



PhpStorm 3.0.0 Web Help

PhpStorm

Previous | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm is a new Integrated Development Environment for the PHP developers, built on top of [IntelliJ IDEA](#) platform. PhpStorm inherits all the web-related functionality of [IntelliJ IDEA](#) for editing PHP, HTML, CSS, JavaScript, XML, working with VCS, SQL, plus adds advanced support for the other tools specific to web development.

PhpStorm brings you the following advanced features:

- Intelligent Editor: for PHP, HTML, CSS, JavaScript, and XML, which includes syntax highlighting, documentation lookup, and refactorings.
- Error-Free Coding: on-the-fly code analysis, error highlighting and quick fixes.
- Project and Code Navigation: specialized project views, file structure views and quick jumping between files, classes, methods and usages.
- VCS Integrations: out-of-the-box support for Subversion, Perforce, Git, and CVS with changelists and merge.
- FTP Sync: update server using FTP or SFTP.
- SQL support: coding assistance, SQL console and Database browser.

JetBrains PhpStorm is cross-platform. It works on Windows, Mac OS X and Linux and brings the whole range of precise developers tools, all tied together to create the convenient development environment.

- [What's New](#)
- [Getting Help](#)
- [Getting Started with PhpStorm](#)
- [Basic Concepts](#)
- [PhpStorm Usage Guidelines](#)
- [Language and Framework-Specific Guidelines](#)
- [Reference](#)

See Also

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

What's New

Previous | [Next](#) | [See Also](#) | [Comments](#)

What's New in Version 3.0

IDE

- [Possibility to preserve temporary files](#)
- [Possibility to reopen a project from the Welcome screen](#)
- [Possibility to drag and drop a project onto the Welcome screen](#)

Code Editing

- [New PHPDoc formatting options in compliance with Zend, PEAR and other standards](#)

Configuring Project and IDE

- [Possibility to copy code style from another language](#)
- [Safe write mode](#)

Tool Windows

- [Tool Windows are now available from the View | Tool Windows menu](#)
- [Dedicated JSTestDriver tool widow for starting the JsTestDriver server for running JavaScript unit tests in the browser](#)

Navigation and Search

- [Navigation commands are available on the Navigate menu](#)
- [Search functionality is in the Edit menu](#)

Version Control

- [Reviewing changes to be checked in \(Digest view\) right in the Commit dialog box](#)
- [Viewing local changes from base revision in a dedicated pane on the Local tab of the Changes tool window](#)
- [Git Fetch is performed silently, without showing the Fetch Settings dialog box](#)
- [Initial support of git gists for sharing code snippets on GitHub](#)
- [Possibility to view change details for a file](#)
- [Revision graphs for Git](#)

Inspections

- [Detecting code duplicates](#)

PHP Support

- [New PHPDoc formatting options in compliance with Zend, PEAR and other standards](#)
- [Debugging single php http requests supported](#)
- [Profiling using integration with XDebug and Zend Debugger](#)
- [PHPUnit 3.6 supported](#)
- [Generating PHP unit tests improved](#)
- [UML class diagrams for PHP code are supported](#)

Debugging

- [XSLT debugger supported](#)

Data Access Support

- [Possibility to change an SQL dialect for an SQL or DDL file open in the editor](#)

JavaScript Support

- [The mark object action is available in the JavaScript debugger](#)
- [JavaScript unit testing](#)
- [NodeJS: coding assistance, running, debugging, and unit testing](#)

Refactoring

- [Search for references option is available for renaming files](#)

Testing Support

- [PHPUnit 3.6 supported](#)
- [Generating PHP unit tests improved](#)
- [Unit Testing for JavaScript. JSTestDriver Assertion, QUnit, and Jasmine frameworks are supported](#)
- [Running JavaScript unit tests in browser](#)

Remote Hosts

- [Synchronizing local and remote folders in the difference viewer](#)

Miscellaneous Improvements

- [Column selection mode](#)
- [Macros functionality is in the Edit menu](#)
- [Quick hide/show tool window buttons](#)
- [Possibility to compare binary files](#)
- [Highlighting level of the current file can be configured from the Code menu](#)
- [Find Action is in Help menu](#)
- [Hierarchies built are in the Navigate menu](#)

What's New in Version 2.1.0

Code Editing

- [Moving lines up and down](#)
- [CamelHumps in code completion are detected without using the Shift key](#)

Version Control

- [Viewing the GitHub version of a file from PhpStorm](#)
- [Setting up connection to Perforce server has become highly fail-proof](#)

PHP Support

- [Viewing parameter info for methods defined through the @method phpDocumentor tag is available](#)
- [Dedicated Command Line Tools tool window](#)
- [PhpStorm recognizes .htaccess files and provides syntax highlighting, formatting, code completion and documentation lookup for common directives](#)
- [Command Line Tools functionality is provided through a dedicated tool window with the possibility to navigate through the list of executed commands and their output and save the output in a text file](#)
- [Phing build tool is supported](#)
- [Workaround for the XDebug on FreeBSD crash can be enabled](#)

What's New in Version 2.0

Running and Debugging

- [Referencing labeled variables, watches, and objects by labels is available](#)

Version Control

- [On-the-fly spellchecking in commit messages is available](#)

Markup languages and style sheets

- [SASS3 support, with syntax highlighting, indentation-based code folding, and color editor](#)
- [LESS support, with compilation and coding assistance](#)

Inspections

- [Running inspections by name](#)
- [Renaming, moving, and deleting files and folders on remote hosts](#)

PHP Support

- [PHP command line tools are supported. You can use integration with Zend Framework or Symfony or define a custom framework](#)
- [Integration with Zend Debugger is supported](#)
- [Zero-configuration debugging](#)

Data Access Support

- [JDBC drivers are downloaded right from Data Sources tool window](#)
- [Operations with database binary large objects \(BLOBs\) are supported](#)
- [Tables can be edited using a graphical Table Editor. The range of records to be displayed can be controlled through filters](#)

JavaScript Support

- [The link to documentation detected automatically upon adding a JavaScript library](#)

Miscellaneous Improvements

- [Detaching editor tabs](#)

See Also

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Getting Help

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

PhpStorm provides a number of features that will help you obtain the necessary information. These features include:

- [Help topics](#)
- [Tips of the Day](#)
- [Default Keymap Reference](#)
- [The Productivity Guide](#)
- [Online resources](#)
- [Sharing your feedback](#)

Using help topics

The contents of the built-in documentation is divided into the following major parts:

Getting Started

- The [Guided Tour](#) and [Tool Windows](#) parts contain essential information about the user interface.
- The part [Familiarize Yourself with PhpStorm Editor](#) helps you master the PhpStorm editor, and describes the basic and advanced editing procedures.
- Besides that, you can find information on PhpStorm registration.

Basic Concepts

The [Basic concepts](#) part contains information about the notions that underlie the PhpStorm functionality.

Product Usage Guidelines

The [PhpStorm Usage Guidelines](#) part provides descriptions of the actions required to fulfil certain common tasks with PhpStorm. Note that the part page contains an alphabetical index of all the available task descriptions.

PHP-Specific Guidelines

The [PHP-Specific Guidelines](#) part outlines a meta procedure of working with PHP and gives some details for performing PHP-specific tasks.

Reference

In the [Reference](#) part you can find descriptions of the tool windows, dialog controls, lists of the keyboard shortcuts, information related to specific version control systems etc.

Built-in documentation enables you to browse through the topics using the table of contents, find occurrences in the **Search** tab, or use the detailed **Index** that contains all keywords from all topics.

To bring up help contents, do one of the following

- On the main menu, choose **Help | Help Topics**.
- Press **F1F1**.
- Click **Help** button, if it is available.

To find a particular piece of information

1. Click the **Search** tab of the help viewer.
2. In the search field, type the search string and press **EnterEnter**.
3. In the list of topics that contain occurrences of the search string, select the desired one. Occurrences are highlighted in the right pane of the help viewer.

To find a keyword in the index tab

1. Click the **Index** tab of the help viewer.
2. In the search field, type the desired keyword and press **EnterEnter**. The caret rests at the first occurrence of the keyword. Every time you press **EnterEnter**, the caret moves to the next occurrence of the keyword. To see the information about a keyword, select one of its sub-entries.

Using tips of the day

Tips of the Day provide a collection of useful and interesting hints. They show up every time you start PhpStorm. If you want to suppress Tips of the Day, clear the check box **Show Tips on Startup**.

To show a tip of the day, choose **Help | Tip of the Day** on the main menu. To navigate through the collection of tips, use the **Previous** and **Next** buttons.

Using the productivity guide

PhpStorm smartly analyzes the features you use most often during your development sessions, and reminds you of the features you might have missed. Actually, the Productivity Guide is your personal adviser that helps you fulfil your tasks in the most efficient way.

The Productivity Guide automatically displays a Tips of the Day window, with the features personally recommended to a particular user, during compilation and other time-consuming operations, such as repository updates, downloads etc.

To view your productivity statistics and enable or disable display of the productivity guide

1. On the main menu, choose **Help | Productivity Guide**. The [Productivity Guide](#) dialog appears.
2. Select or clear the show check boxes to define when the personalized Tips of the Day should display.

Default keymap reference

PhpStorm provides the default platform-specific keymap reference in pdf format, and suggests two ways of viewing it:

- [Help | Default Keymap Reference](#)
- [Keyboard Reference](#) 

Using online resources

If built-in documentation fails to answer your questions, you can find more information on the web. The following resources are available:

- [Official JetBrains home page](#) 
- [WebIDE Community](#) 
- [Blog](#) 

Sharing your feedback

To share your feedback, you have the following means available:

- Choose **Help | Submit Feedback**, which redirects you to [our bug tracking system YouTrack](#) 
- Use the [Feedback page](#).
- Click the **Comments** link on any page of the online version of the PhpStorm documentation. In the **Comments** area, vote for the page, and leave your comment

See Also

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Getting Started with PhpStorm

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm is an advanced IDE, focused on boosting your productivity to enable you to deliver the cutting-edge code in almost no time.

This chapter briefly outlines what you will need to start using PhpStorm right away.

Get quickly accustomed with PhpStorm:

- [System Requirements and Installation](#)
- [Importing PhpStorm Settings](#)
- [Register PhpStorm](#)
- [Guided Tour Around PhpStorm User Interface](#)
- [Familiarize Yourself with PhpStorm Editor](#)
- [PhpStorm Tool Windows](#)
- [Compare Files and Folders](#)
- [Familiarize Yourself with IDE Navigation](#)
- [Copy and Paste Between PhpStorm and Explorer/Finder](#)
- [Configure Your Working Environment](#)
- [Create and Run Your First PHP Project](#)
- [Create and Run Your First Web Project](#)
- [Keyboard Shortcuts You Cannot Miss](#)

See Also

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

System Requirements and Installation

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

In this section:

- [Windows](#)
- [MacOS](#)
- [Linux](#)

- [Switching from OpenJDK to SunJDK](#)

Windows

System Requirements

- Microsoft Windows 7 (incl.64-bit)/Vista/2003/XP/2000
- Intel Pentium III/800 MHz or higher (or compatible)
- 512 MB RAM minimum, 2 GB RAM recommended
- 1024x768 minimum screen resolution

Installation

1. Download PHPStorm at <http://www.jetbrains.com/phpstorm/download/index.html>
2. Run the `PhpStorm-*.exe` file that starts the Installation Wizard.
3. Follow all steps suggested by the wizard. Please pay special attention to the corresponding installation options.

MacOS

System Requirements

- Mac OS X 10.5 or higher, including 10.7 (Lion)
- JDK 6
- 512 MB RAM minimum, 2 GB RAM recommended
- 1024x768 minimum screen resolution

Installation

1. Download the `PhpStorm-*.dmg` Mac OS X Disk Image file at <http://www.jetbrains.com/phpstorm/download/index.html>
2. Mount the `PhpStorm-*.dmg` Mac OS X Disk Image file as another disk in your system.
3. Copy PhpStorm to your **Applications** folder.

Linux

System Requirements

- Intel Pentium III/800 MHz or higher (or compatible)
- 512 MB RAM minimum, 2 GB RAM recommended
- 1024x768 minimum screen resolution
- [Oracle \(Sun\) JDK 1.6](#) or higher. **Open JDK** is not supported.
- [GNOME](#) or [KDE](#) desktop.

Installation

1. Download the `PhpStorm-*.tar.gz` file at <http://www.jetbrains.com/phpstorm/download/index.html>
2. Unpack the `PhpStorm-*.tar.gz` file using the following command:


```
tar xzf PhpStorm-*.tar.gz
```
3. Run `PhpStorm.sh` from the `bin` subdirectory.
4. If necessary, [switch from OpenJDK to Oracle \(Sun\) JDK 1.6](#) or higher.

To switch from openjdk to sunjdk

Running PhpStorm using **OpenJDK** Java runtime is not supported due to numerous **OpenJDK** performance and graphics problems. If you have not previously installed [Oracle \(Sun\) JDK 1.6](#) or higher, it is strongly recommended that you switch to it even if you have started PhpStorm.

OpenJDK is included in a number of popular Linux distributions. The procedure below explains how to switch to **Oracle (Sun) JDK 1.6** in Ubuntu.

1. Enable the Canonical Partner Software through the following steps:
 1. Go to [Ubuntu Software Center](#)
 2. Click **Edit**.
 3. Click **Software Sources**.
 4. Switch to the **Other Software** tab and click **Canonical Partners**. This may take some time.
2. In the **Command** prompt, type the following sequence of commands:
 1. `sudo apt-get update`
 2. `sudo apt-get install sun-java6-jdk sun-java6-jre sun-java6-plugin`
 3. `sudo update-alternatives -config java`

See Also

External Links:

- <http://theaccidentalcoder.com/content/swapping-openjdk-sun-jdk-ubuntu>
- <http://www.jetbrains.com>

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Importing PhpStorm Settings

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

While working with PhpStorm, you've definitely customized the IDE settings (for example, editor behavior, or keyboard shortcuts) to fit your needs. Having installed and launched a newer version of PhpStorm (when the `config` directory doesn't yet exist for the new version), you might want to re-use the settings of the previous versions.

For this purpose, PhpStorm automatically shows a dialog box with the following options:

- **I want to import settings from the previous version:** If this option is selected, PhpStorm automatically detects the home directory of the previous version, and imports settings from this location.
- **I want to import settings from custom location:** If this option is selected, you can specify the desired directory where the settings are stored.

Type the path manually, or click the browse button to select settings from the previous version using the file chooser dialog.

- **I don't have a previous version of PhpStorm, or I don't want to import settings:** If this option is selected, then all the previous settings (if any) will be ignored.

Click the radio-button of the desired option to re-use or ignore the legacy settings.

See Also

Concepts:

- [Project and IDE Settings](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Register PhpStorm

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

To get acquainted with PhpStorm, you can download and install its trial version for free. This trial version is available for 30 days, whereupon you need to obtain and register a license.

PhpStorm provides several types of licenses, to flexibly meet the demands of the customers. In particular, these are: personal, commercial, classroom, and open-source licenses. You can find the detailed information about licensing terms at the [Buy](#) page.

In this section:

- [Registering your license key](#)
- [Getting a permanent license](#)
- [Troubleshooting](#)

To register your license key

1. On the main menu, choose **Help | Register**.
2. In the **Enter License Data** dialog box, select the desired option:
 - o **Enter license key:** specify the user name and license key.
 - o **Enter license server address:** specify the license server. If you don't know the address, click **Discover**.
 - o **Evaluate for free for 30 days:** select this option to get free evaluation license.
3. Click **OK**.

To get a permanent license

You need a **permanent license** to be able to work with PhpStorm offline, without the connection to the license server. After specifying the license server address, the **Obtain Permanent License** command appears on the **Help** menu.

1. On the main menu, choose **Help | Obtain Permanent License**.
2. In the dialog box that opens, type your email address, to which your activation key will be sent.
3. Click **OK**.

Troubleshooting

A license key can be rejected by the software in certain cases. The table below outlines the reasons of license rejections, and the possible ways to solve the problem.

Reason	Solution
Wrong user name	Make sure that you are using the User Name which is specified in your license certificate email. If a license is registered in your company name, it will not work with your personal name.
Misspelled User Name/License Key	To avoid misspellings, we recommend that you copy your User Name and license key from the license certificate e-mail rather than enter them manually in the software. You can do so by using the Ctrl+C Command C or Ctrl+Insert Command Insert and Ctrl+V Command V or Shift+Insert Shift Insert shortcuts.
A license key does not qualify for upgrade to a higher version	If your license key does not work with the newly installed version of the software, please make sure that your current license key allows you to upgrade for free to the latest version. If it does not, please contact sales (sales@jetbrains.com) in order to upgrade your license.
Concurrent use of a license key	Make sure that your license key is not being used by another developer in your company at the same time. Concurrent use of a single license on multiple instances of the software is restricted by our software license agreements (unless you have obtained a floating license) and is prevented by software functionality.

See Also

Reference:

- [Updates](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

```
ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js'; var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s); }());
```



PhpStorm 3.0.0 Web Help

Guided Tour Around PhpStorm User Interface

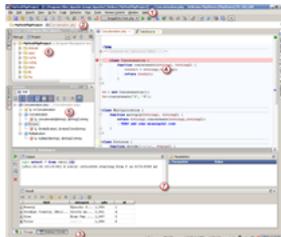
[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

This chapter gives you insight into how PhpStorm user interface is organized to help you find your way through your working environment.

Tip

This chapter outlines the default (out-of-the-box) IDE interface layout. Note that plugins and other add-ons you are installing may change the way your IDE looks and behaves, for example there can be extra command buttons or menu items appearing in uncommon locations.

When you first run PhpStorm, or have no open project, PhpStorm displays the [Welcome screen](#), which enables quick access to the major entry points. When a project is opened, PhpStorm displays the main window. This window is made up of six logical areas, which are shown on the picture below, marked with red number labels.



[Click thumbnail to view larger image.](#)

1. [Menus and toolbars](#) - the main menu and toolbars let you carry out various commands.
2. [Navigation bar](#) that helps navigate through the project and open files for editing.
3. [The status bar](#) - indicates the status of your project, the entire IDE, and shows various warning and information messages.
4. [The editor](#) - here you create and modify the code.
5. [PhpStorm tool windows](#) - secondary windows that provide access to various specific tasks (project management, source code search and navigation, running and debugging, integration with version control systems, etc.).

See Also

Concepts:

- [Basic Concepts](#)

Procedures:

- [PhpStorm Usage Guidelines](#)

Reference:

- [Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

```
ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js'; var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s); })(window);
```



PhpStorm 3.0.0 Web Help

Familiarize Yourself with PhpStorm Editor

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

PhpStorm editor is a powerful tool for creating and modifying source code of the supported file types. You can configure your editing environment in the [Settings dialog](#).

The editor is tab-based. All operations with the editor tabs are available from the context menu of a tab, or from **Window | Editor tabs** node of the main menu.

When you [open a file for editing](#), it opens in its own tab. The editor you are currently working in, is the *active editor*. You can change behavior of the active editor using the commands under **View | Active Editor** node of the main menu.

Tip

You always return the focus to the active editor from any tool window by pressing the `Escape` key.



[Click thumbnail to view larger image.](#)

1. Editor Area

Use this area to type and edit your source code. The editor suggests numerous coding assistance facilities. Refer to the sections under this node, and to [Basic Editing Procedures](#) and [Advanced Editing Procedures](#) for details.

2. Gutter Area

The left gutter provides additional information about your code and displays the various icons that identify the code structure, bookmarks, breakpoints, scope indicators, change markers and the code folding lines that let you hide arbitrary code blocks.

3. Smart Completion Pop-up

This is one of the key editing assistance features that suggests method names, functions, tags and other keywords you are typing.

4. Document Tabs

Enable quick navigation across the multiple documents you are working on. Clicking a tab brings its contents to front and makes it available for editing in the active editor.

To move between the tabs, use the keyboard shortcuts **Alt+RightCommand Right** or **Alt+LeftCommand Left**.

Clicking a tab while **Ctrl+Click** is pressed, allows to navigate to any part of the file path, through opening it in an external browser.

Context menu of a tab provides all commands applicable to a file opened in the editor, for example:

- [Close one or more tabs.](#)
- [Pin active tab.](#)
- [Split and unsplit tabs.](#)
- [Manage groups of tabs.](#)
- [Navigate between tabs.](#)
- [Add to Favorites.](#)
- Move to changelist.
- [Run](#), or [debug](#) in the active editor.
- Perform [local history](#) and [version control](#) commands.
- Perform commands of [your own tools](#).

By default, the tabs appear on top of the editor, but you can change their location in the **Appearance** page of the [Editor](#) dialog box.

5. Validation Side Bar / Marker Bar

This is the bar to the right from the editing area, showing the green, red or yellow box on its top depending on whether your code is okay, or contains errors or warnings. This bar also displays active red, yellow, white and green navigation stripes that let you jump exactly to the erroneous code, changed lines or search results.

In this section:

- [Opening and Reopening Files in the Editor](#)
- [Closing Files in the Editor](#)
- [Saving and Reverting Changes](#)
- [Adding Editors to Favorites](#)
- [Managing Editor Tabs](#)
- [Basic Editing Procedures](#)
- [Advanced Editing Procedures](#)

See Also

Concepts:

- [Supported Languages](#)

Procedures:

- [Finding and Replacing Text in File](#)
- [IDE Settings](#)
- [Running](#)
- [Debugging](#)

Reference:

- [Tool Windows Reference](#)
- [Navigation in Source Code](#)
- [General](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

```
ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js'; var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s); }());
```



PhpStorm 3.0.0 Web Help

1.0+

Opening and Reopening Files in the Editor

Previous | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

In this section:

- [Opening files for editing](#)
- [Opening entire directories](#)
- [Reopening recent files](#)

To open a file for editing

1. Do one of the following:
 - o Double-click the desired file in one of the [Tool Windows](#).
 - o Select the desired file in one of the [Tool Windows](#) and press **F4**.
 - o Select the desired file in one of the [Tool Windows](#) and choose **Jump to Source** on the context menu.
 - o Use the [Navigate](#) command for a Class, File, or Symbol.

- o Click the desired directory in the [Navigation bar](#), and select file from the drop-down list:

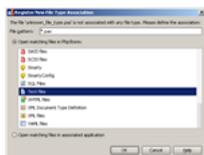


Click thumbnail to view larger image.

2. If the file type is registered, the file opens silently in PhpStorm's editor.

If the file type is registered under the category **Files opened in associated applications**, it will be opened in its associated application, rather than in the PhpStorm editor. By default, PhpStorm suggests a number of such file types, for example `.doc`, `.chm`, or `.pdf`.

If the file type is unknown, PhpStorm suggests you to choose whether you want to register a new file type, or open such file in its associated application. Specify your choice in the [Register New File Type Association](#) dialog box:



Click thumbnail to view larger image.

Note

You can register the required file types on the [File Types](#) page of the **Settings** dialog box.

Tip

To open a file which is external to your project, do one of the following:

- o Choose **File | Open File** on the main menu and select the desired file in the dialog box that opens.
- o [Drag](#) the required file from the Explorer (on Windows), File Browser (Linux), or Finder (on Mac) and [drop](#) it to the editor. The file will be opened for editing in a new tab.

To open a directory

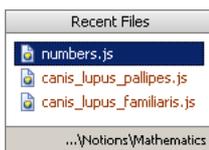
- On the main menu, choose **File | Open Directory**, and select the desired directory in the **Select Path** dialog box that opens.

Tip

After that, you can [open](#) any file from the directory either in the editor or in the associated application. If the [file type is unknown](#), you can also register it in the [Register New File Type Association](#) dialog box.

To reopen a recent file, do one of the following:

- To open a **recently** opened file, choose **View | Recent Files** on the main menu or press `Ctrl+E` (Windows) or `Command E` (Mac). Then select the desired file from the **Recent Files** pop-up window, that opens.



- To open a **recently** updated file, on the main menu, choose **View | Recently Changed Files** or press `Ctrl+Shift+E` (Windows) or `Command Shift E` (Mac). Then select the desired file from the **Recently Edited Files** pop-up window, that opens.

Tip

Use **Recent files limit** text box in the [Editor](#) dialog box to define the maximum number of recent files.

See Also

Procedures:

- [Navigating to Recent File](#)
- [Closing Files in the Editor](#)
- [Opening Language Injections in the Editor](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Closing Files in the Editor

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

PhpStorm suggests several ways to close editor tabs.

To close a file in the editor, do one of the following

- On the main menu point to **Window | Editor Tabs**, choose one of the appropriate closing commands.

Close

Closes the file in the active tab.

Close All

Closes all editor tabs.

Close Others

Closes all tabs except the current one.

Close Unmodified

Closes all files that were not changed.

Close All But Pinned

Closes all files that were not pinned. This command appears, if there are pinned editor tabs.

- Right-click any editor tab, and choose same commands on the context menu.
- Point with your mouse cursor to a tab and click the middle mouse button.
- Point with your mouse cursor to a tab and click **X**.
- Press **Ctrl+F4**/**Command F4**.

Tip

When you close modified files, PhpStorm preserves all changes in the current editing session. After reopening such files, the results of editing are restored.

See Also

Procedures:

- [Opening and Reopening Files in the Editor](#)
- [Saving and Reverting Changes](#)
- [Closing an Editor for a Language Injection](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Saving and Reverting Changes

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

You can save the results of editing yourself, and also configure PhpStorm to enable automatic saving of changes. This section describes how to:

- [Save changes when done with editing.](#)
- [Alert you about unsaved changes](#) by marking a modified file in the editor tab with an asterisk next to its name.
- [Enable auto-saving changes on the regular basis.](#)
- [Enable auto-saving changes on switching to the other applications.](#)
- [Enable preserving temporary files.](#)
- [Revert changes.](#)

To save results of editing, do one of the following

- On the main menu, choose **File | Save All**.
- Press **Ctrl+S**/**Command S**.

To alert you about unsaved changes

1. Open [Settings](#) dialog box, expand the [Editor](#) node, and click **Editor Tabs**.
2. In the [Editor Tabs](#) page, select the **Mark modified tab with asterisk** check box.

To enable auto-saving on the regular basis

1. In the **IDE Settings** section of the [Settings](#) dialog box, click [General](#).
2. Select the **Save files automatically, if the application is idle** check box and specify the number of seconds of idleness required to activate saving.

To enable saving changes when switching to the other applications

1. In the **IDE Settings** section of the [Settings](#) dialog box, click [General](#).
2. Select the **Save files on frame deactivation** check box.

Tip

If the check box is cleared, a conflict may occur between the PhpStorm's version in the memory cache and the version produced in the file system. PhpStorm [prompts you to resolve the conflict](#).

To enable preserving temporary files while saving changes

1. In the IDE Settings section of the [Settings](#) dialog box, click [General](#).
2. Select the Use "safe write" check box.

If this check box is selected, modified file will be first saved as a temporary file. If the save operation is completed successfully, the temporary file will be renamed, and the original file will be deleted.

To revert changes

- On the main menu, choose **File | Reload from Disk**.

Tip

Refer to the [Local History](#) for more powerful way of reverting changes.

See Also

Procedures:

- [Using Local History](#)

Reference:

- [Settings Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

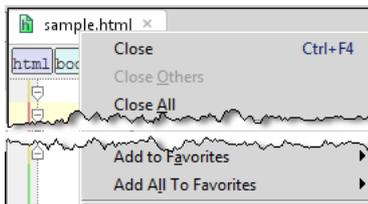
Adding Editors to Favorites

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

You can group most needed items into Favorite lists and get quick access to them through the [Favorites](#) view.

To add one or more items to favorites

1. Do one of the following:
 - o Open the desired files in the Editor.
 - o Select one or more items in the [Project](#) tool window.
2. Right-click the editor tab or the selection in the Project tool window, and choose **Add to Favorites** on the context menu.



3. On the submenu, specify the Favorites list to add the selected items to. Do one of the following:
 - o To add the items to an existing list, select the desired list in the submenu.
 - o To create a new list, choose **Add to New Favorites List**. In the **Add New Favorites List** dialog box that opens enter the desired group name or accept default settings.

See Also

Reference:

- [Favorites View](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Managing Editor Tabs

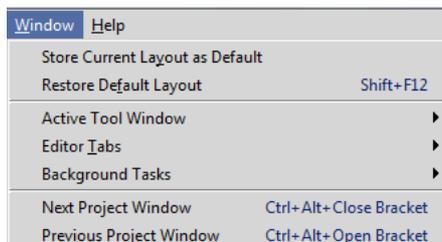
[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Every time you open a file for editing, a dedicated tab is added to the editor window, next to the active editor tab. By default, when the number of tabs reaches its limit, the editor closes the tabs according to the [tab closing policy](#).

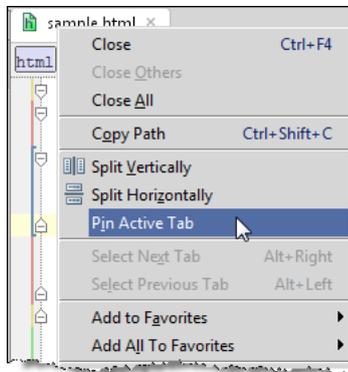
Note

All commands, related to managing editor tabs, are available from:

- **Window | Editor Tabs** menu:



- Context menu of a tab:



In this section:

- [Configuring Behavior of the Editor Tabs](#)
- [Navigating Between Editor Tabs](#)
- [Pinning and Unpinning Tabs](#)
- [Splitting and Unsplitting Editor Window](#)
- [Detaching Editor Tabs](#)
- [Editing Multiple Files Using Groups of Tabs](#)

See Also

Procedures:

- [Accessing the IDE Settings](#)
- [Pinning and Unpinning Tabs](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Configuring Behavior of the Editor Tabs

Previous | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

You can change the way PhpStorm handles the editor tabs in the [Editor tabs](#) page of the Editor settings.

To change the maximum allowed number of tabs

1. Open the Editor settings. To do that, click  on the main toolbar, then click the Editor node in the [Settings](#) dialog box, that opens.
2. In the [Editor Tabs](#) page of the Editor settings, type the desired maximum allowed number of the editor tabs to be opened at a time in the Tab limit field.

Tip

If the tab limit equals to 1, the tabs will be disabled. If you want the editor to never close the tabs, type some unreachable number.

To disable editor tabs

- In the [Editor Tabs](#) page of the Editor settings, select None from the Tab placement drop-down list.

Tip

With the disabled tabs, use the Recent files (`Ctrl+ECommand E`) command to quickly switch between files.

See Also

Reference:

- [Editor. Editor Tabs](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Navigating Between Editor Tabs

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

PhpStorm suggests two approaches to moving between the editor tabs:

- First, you can [jump](#) from one tab to another as your editing session requires. While you move between the editor tabs, PhpStorm remembers the caret position within each opened file.
- The second approach enables you to [move back and forth through the history](#) of your navigation, same way as it is done in a Web browser. As you move from file to file during your editing session, PhpStorm keeps track of the visited locations and enables you to go back, using the **Navigate | Back / Forward** commands.

To navigate from the current tab to the next or previous tab

Do one of the following:

- Right-click the current editor tab and choose **Select Next/Previous Tab** on the context menu.
- Press **Alt+RightCommand Right** or **Alt+LeftCommand Left**. So doing, the focus moves to the editor tab located next to the right or to the left from the active editor tab.
- Press **Ctrl+Tab Command Tab** or **Ctrl+Shift+Tab Command Shift Tab** to use the [Switcher](#).

To go back and forth through the history of visited tabs

Do one of the following:

- On the main toolbar, click the buttons  or .
- On the main menu, choose **Navigate | Back / Forward**.
- Press **Ctrl+Alt+LeftCommand Alt Left** or **Ctrl+Alt+RightCommand Alt Right**.

Note

On a Mac OS X computer, you can also use the three-finger right-to-left and left-to-right swipe gestures.

See Also

Procedures:

- [Navigating to Recent File](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Pinning and Unpinning Tabs

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

When you pin a tab, you make sure that such tab will not be automatically closed, when the editor [starts closing the editor tabs](#) automatically. Besides, when you close the editor tabs, you have an option to preserve pinned tabs opened and close only the unpinned tabs.

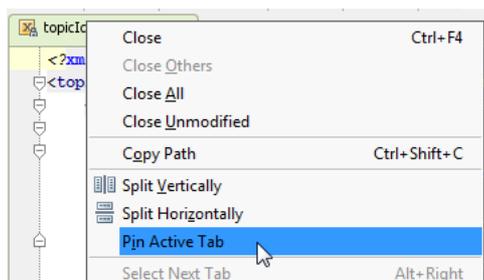
When a tab is pinned, its header is marked with the  icon.

Tip

The **pinned** status of a tab is helpful when you open a file for reference, rather than for editing.

To pin an editor tab

1. Switch to the desired editor tab.
2. Right-click the editor tab, and choose **Pin Active Tab** on the context menu:



To unpin a tab

1. Switch to the desired editor tab.
2. Right-click the editor tab, and choose **Unpin Active Tab** on the context menu.

See Also

Procedures:

- [Closing Files in the Editor](#)

Reference:

- [Editor](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

```
ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js'; var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s); });
```



PhpStorm 3.0.0 Web Help

Splitting and Unsplitting Editor Window

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Splitting the editor window divides it into independent panes. You can split the editor window into as many panes as required, each one containing multiple tabs.

Each pane can be allocated vertically or horizontally. Thus, splitting helps create different editor layouts, organize tabs into groups, and [edit multiple files simultaneously](#). For example, you can scroll through a part of a file, having at the same time the possibility to view the lines in its other part.



[Click thumbnail to view larger image.](#)

To split an editor tab

1. Switch to the desired tab.
2. Right-click the tab header and choose **Split Vertically** or **Split Horizontally** on the context menu.

To change splitter orientation

1. Switch to the desired tab.
2. Right-click the tab header and choose **Change Splitter Orientation** on the context menu.

To remove splitter

1. Switch to the desired tab.
2. Right-click the tab header and choose one of the following commands on the context menu:
 - o To remove splitting in the active tab, choose **Unsplit**.
 - o To remove splitting in all the open editor tabs, choose **Unsplit All**.

See Also

Procedures:

- [Editing Multiple Files Using Groups of Tabs](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

```
ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js'; var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s); });
```



PhpStorm 3.0.0 Web Help

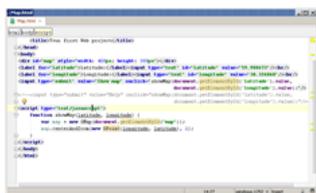
2.0+

Detaching Editor Tabs

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm makes it possible to detach editor tabs, and move them to separate frames.

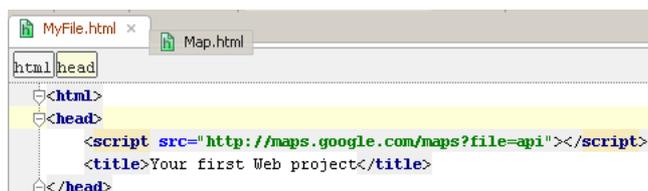
- To detach an editor tab, drag it until a preview thumbnail appears:



Click thumbnail to view larger image.

The contents of the editor tab is opened in a separate frame.

- To attach an editor tab, drag it from its frame and drop to the main PhpStorm frame until tab name appears:



Tip

- The detached editor tabs can be [split or unsplit](#).
- You can move editor tabs between split panes.

See Also

External Links:

- [Demo](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Editing Multiple Files Using Groups of Tabs

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

You can arrange tabs into groups to facilitate working with multiple files at a time. PhpStorm allows to have an unlimited number of tab groups, thus enabling you to view several files or several place within the same file.

To create a new group of tabs

- Just [split](#) the desired editor tab.

To move a tab from one group to another

1. Switch to the desired tab.
2. Right-click the desired editor tab and choose **Move Tab to Opposite Tab Group** on the context menu.

See Also

Procedures:

- [Splitting and Unsplitting Editor Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Basic Editing Procedures

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

This section describes how to perform the most common editing tasks:

- [Adding, Deleting and Moving Lines](#)
- [Cutting, Copying and Pasting](#)
- [Selecting Text in the Editor](#)
- [Undoing and Redoing Changes](#)

- [Using Drag-and-Drop in the Editor](#)
- [Commenting and Uncommenting Blocks of Code](#)

See Also

Procedures:

- [Configuring IDE Settings](#)

Reference:

- [Editor](#)

Getting Started:

- [Familiarize Yourself with PhpStorm Editor](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

2.1+

Adding, Deleting and Moving Lines

Previous | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

In this section:

- [Adding lines](#)
- [Duplicating lines](#)
- [Deleting lines](#)
- [Moving lines](#)
- [Moving statements](#)

To add a line

- Press Shift+EnterShift Enter to add a new line after the one where the caret is currently located and move the caret to the beginning of this new line.

For instance, you have typed some text:

```
function multiplication (a,b){
  c = sum (a,b);
  return d;
}
```

Press Shift+EnterShift Enter to start the next line immediately:

```
function multiplication (a,b){
  c = sum (a,b);
  return d;
}
```

To duplicate a line or fragment

1. Place the caret at the line to be duplicated, or [select](#) the desired fragment of text.
2. Press Ctrl+DCommand D.

To remove a line

- Press Ctrl+YCommand Y to delete the line at caret.

To move a line

1. Place the caret at the line to be moved.
2. Do one of the following:
 - o On the main menu, choose Code | Move Line Up or Code | Move Line Down.
 - o Press Alt+Shift+UpAlt Shift Up or Alt+Shift+DownAlt Shift Down.

PhpStorm moves the selected line one line up or down, performing the syntax check. For example:

```
var s = "The number is ";
document.write("<br/>");
document.write(s + i);
```

After moving line at caret:

```
var s = "The number is ";
document.write(s + i);
document.write("<br/>");
```

To move a statement up or down

1. [Select](#) a statement to be moved, or just place the caret at the first or last lines of a multi-line statement. Note that if moving a statement is not allowed in the current context, the commands

will be disabled.

- 2. Do one of the following:
 - o On the main menu, choose **Code | Move Statement Up/Move Statement Down**.
 - o Press **Ctrl+Shift+UpCommand Shift Up** or **Ctrl+Shift+DownCommand Shift Down**.

Note

If you apply the same commands to a line at caret, or a to a selection, it will be moved one line up or down.

PhpStorm moves the selected statement above the previous one, or directly underneath the next one, performing the syntax check. For example, place the caret at the method declaration:

```
var i=0;
for (i=0;i<=10;i++){
    if(i%2==0) {
        document.write("The number is even.<br/>");
    }
    else {
        document.write("The number is odd.<br/>");
    }
    document.write("The number is ", i, ". ");
}
```

After moving the statement:

```
var i=0;
for (i=0;i<=10;i++){
    document.write("The number is ", i, ". ");
    if(i%2==0) {
        document.write("The number is even.<br/>");
    }
    else {
        document.write("The number is odd.<br/>");
    }
}
```

Tip

Make sure that keyboard shortcuts are not in conflict. You can do that in the [Keymap](#) page of the IDE Settings.

See Also

Procedures:

- [Selecting Text in the Editor](#)
- [Configuring Keyboard Shortcuts](#)

Reference:

- [Keyboard Shortcuts and Mouse Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Cutting, Copying and Pasting

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

PhpStorm provides a number of handy Clipboard operations. You can copy, cut, and paste selected text, a path to a file, or a reference to a symbol.

Because PhpStorm uses the system Clipboard, you can copy and paste between applications. So doing, when pasting Clipboard entries, PhpStorm removes any formatting from the text and any special symbols from the `String` values.

The `Paste` command smartly understands what is being inserted. If you paste a reference to a symbol, it is analyzed for possible imports, references, etc. So doing, PhpStorm provides the necessary brackets and places the caret at the appropriate insertion point. The `Paste Simple` command helps paste any Clipboard entry as a plain text, without any analysis.

PhpStorm enables Clipboard stacking, which means that you can store multiple Clipboard entries and access them with a single shortcut. The number of entries that can be kept in the Clipboard stack is customizable on the [Editor](#) page of the IDE Settings.

To copy selected text, do one of the following

- On the main menu, choose **Edit | Copy**.
- Press **Ctrl+C Command C** or **Ctrl+Insert Command Insert**.
- Click the **Copy** button  on the toolbar.

Tip

The **Ctrl+DCommand D** keyboard shortcut clones a line at the caret or a selected arbitrary fragment of text.

To copy path to a file, do one of the following

- Open the desired file in the editor, then choose **Edit | Copy Path** on the main menu or press **Ctrl+Shift+C** **Command Shift C**.
- Select the desired file in the [Project](#) tool window and choose **Copy Path** on the context menu of the selection.

To cut the selected text

1. [Select](#) the desired fragment in the editor.
2. Do one of the following:
 - On the main menu, choose **Edit | Cut**.
 - Press **Ctrl+X** **Command X** or **Shift+Delete** **Shift Delete**.
 - Click the Cut button  on the toolbar.

To paste the last entry from the clipboard

1. Place the caret in the location where you want to paste content.
2. Do one of the following:
 - On the main menu, choose **Edit | Paste**.
 - Press **Ctrl+V** **Command V** or **Shift+Insert** **Shift Insert**.
 - Click the Paste button  on the toolbar.

To paste the last entry from the clipboard as plain text, do one of the following

- On the main menu, choose **Edit | Paste Simple**.
- Press **Ctrl+Alt+Shift+V** **Command Alt Shift V**.

To paste a specific entry from the clipboard

1. On the main menu, choose **Edit | Paste from History** or press **Ctrl+Shift+V** **Command Shift V** or **Ctrl+Shift+Insert** **Command Shift Insert**.
2. In the **Choose Content to Paste** dialog box select the desired entry from the list of recent Clipboard entries, and click **OK**.

Note

The depth of the Clipboard stack is configured in the Limits section on the [Editor](#) page of the [Settings](#) dialog box. When the specified number is exceeded, the oldest entry is removed from the list.

See Also

Procedures:

- [Configuring IDE Settings](#)

Reference:

- [Editor](#)
- [Keyboard Shortcuts and Mouse Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Selecting Text in the Editor

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts:  

The basic way to select a piece of text is to extend the selection with the mouse cursor. PhpStorm, as a keyboard-centric IDE, suggests to use navigation keys to make selections. You can opt to select pieces of text by lines, or select rectangular fragments in the column mode, extend and shrink the selection.

In this section:

- [Selecting all text in the active editor tab](#)
- [Selecting with navigation keys](#)
- [Extending selection](#)
- [Toqqlimh between selection modes](#)
- [Using the column selection mode](#)
- [Using smart selection](#)

To select the entire text in the current editor tab, do one of the following

- On the main menu, choose **Edit | Select All**.
- Press **Ctrl+A** **Command A**.

To select text with navigation keys, use one of the following shortcuts

- **Ctrl+Shift+LeftAlt** **Shift Left**, **Ctrl+Shift+RightAlt** **Shift Right** to select text from the caret position to the beginning/end of the current word.
- **Ctrl+Shift+PageUp** **Command Shift PageUp**, **Ctrl+Shift+PageDown** **Command Shift PageDown** to select text from the caret position to the top/bottom of the screen.

To extend selection from the word at the caret to the piece of code the caret is contained in, use the following shortcuts

- Press **Ctrl+W** to select the word where the caret is currently located.
- Press **Ctrl+W** successively to extend the selection to the next containing node (for example, an expression, a paired tag, or a whole conditional block, a method body, a class, etc).

Note

Pressing **Ctrl+W** successively in `plain text` or `comments` extends the selection first to the current sentence, then to the current paragraph.

- Press **Ctrl+Shift+W** to shrink selection in the reverse order (from the outermost container to the word where the caret currently resides).

Tip

The selection extends or shrinks according to capitalization, if [CamelHumps mode](#) is enabled (File | Settings | IDE Settings | Editor | Smart Keys | Use "CamelHumps" words for Windows and Linux, PhpStorm | Preferences | IDE Settings | Editor | Smart Keys | Use "CamelHumps" words for Mac OS).

If you want to make selection according to capitalization, using double-click, make sure that the option [Honor CamelHumps words...](#) is also turned on (File | Settings | IDE Settings | Editor | Honor CamelHumps words... for Windows and Linux, PhpStorm | Preferences | IDE Settings | Editor | Honor CamelHumps words... for Mac OS).

To toggle between the line and the column selection modes, do one of the following

- On the main menu, choose **Edit | Column Selection Mode**.
- On the context menu of the editor, choose **Column Selection Mode**.
- Press **Alt+Shift+Insert** to multiply.
- Keeping the middle mouse button pressed, drag the mouse.

To make selection in the column selection mode

- Keeping the **Alt** key pressed, drag your mouse pointer to select the desired area.
- Keeping the middle mouse button pressed, drag your mouse pointer to select the desired area.

To use smart expression selection

When you perform various [code refactorings](#) that involve selecting an expression, PhpStorm can help you select the expression of interest. This feature is known as [smart expression selection](#).

The procedure for the Introduce Variable refactoring is just an example. The smart expression selection feature, in fact, is available in all the refactorings that start with selecting an expression.

1. Place the cursor before or within the expression.

```

numbers.js x
var i=0;
for (i=0; i<=10; i++)
{
  if (i==3)
  {
    continue;
  }
  document.write("The number is " + i);
  document.write("<br />");
}
  
```

2. Choose **Refactor | Introduce Variable** from the main or the context menu, or press **Ctrl+Alt+V**.
3. In the **Expressions** pop-up menu, select the expression. To do that, click the required expression. Alternatively, use the **Up** and **Down** arrow keys to navigate to the expression of interest, and then press **Enter** to select it.

```

numbers.js x
var i=0;
for (i=0; i<=10; i++)
{
  if (i==3)
  {
    continue;
  }
  document.write("The number is " + i);
  document.write("<br />");
}
  
```

Expressions

- "The number is"
- "The number is " + i
- document.write("The number is " + i)

See Also

Procedures:

- [Configuring IDE Settings](#)
- [Introduce Variable](#)

Reference:

- [Editor](#)
- [Keyboard Shortcuts and Mouse Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Undoing and Redoing Changes

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: 

The `Undo` command discards the last changes to the file in the editor. The `Redo` command discards the results of the last `Undo` command.

You can undo or redo your changes as many times as required. However, when you exit PhpStorm, the undo history is lost.

PhpStorm smartly defines the logical steps that can be undone and redone. The following events signal about the end of a logical step:

- Pressing `Enter`.
- Repositioning the mouse cursor.
- Using navigation keyboard shortcuts.
- Cutting or pasting.
- Pressing `Tab`.

To undo an action, do one of the following

- On the main menu, choose `Edit | Undo`.
- Press `Ctrl+Z` `Meta Z` or `Alt+BackSpace`.

To redo an action, do one of the following

- On the main menu, choose `Edit | Redo`.
- Press `Ctrl+Shift+Z` `Command Shift Z` or `Alt+Shift+BackSpace` `Alt Shift BackSpace`.

Tip

PhpStorm expands the undo and redo mechanism to complex operations, such as reformatting or refactoring source code, creating or deleting files. When you undo or redo a complex operation, PhpStorm requests on your confirmation.

See Also

Reference:

- [Keyboard Shortcuts and Mouse Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Using Drag-and-Drop in the Editor

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: 

PhpStorm allows copying and moving code fragments within the active editor tab, by means of drag-and drop.

To move or copy code fragment

1. Make sure that using drag-and-drop is enabled in the [Editor](#) page of the IDE Settings.
2. [Select](#) the desired fragment in the editor.
3. Move or copy selection:
 - Move: Drag the selected fragment to the target location.
 - Copy: Keeping the `ControlCommand` key pressed, drag selection to the target location.

See Also

Procedures:

- [Editor](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Commenting and Uncommenting Blocks of Code

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: 

You can comment or uncomment the current line or selected block of source code.

Commenting feature extends to all supported file types. For the custom file types, you can define line and block comments characters, as described in the section [Creating and Registering File Types](#).

To comment or uncomment the current line of code, do one of the following

- On the main menu, choose **Code | Comment with Line Comment**.
- Press **Ctrl+Slash** `Command Slash` or **Ctrl+Divide** `Command Divide`.

To add or remove a block comment, do one of the following

- On the main menu, choose **Code | Comment with Block Comment**.
- Press **Ctrl+Shift+Slash** `Command Shift Slash` or **Ctrl+Shift+Divide** `Command Shift Divide` .

See Also

Procedures:

- [Creating and Registering File Types](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Advanced Editing Procedures

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

This part describes more sophisticated editing techniques provided by PhpStorm:

- [Code Folding](#)
- [Improving Visibility of the Source Code](#)
- [Spellchecking](#)
- [Using TODO Lists](#)
- [Using Macros in the Editor](#) 

See Also

Reference:

- [Advanced Editing](#)

External Links:

- [Configuring IDE Settings](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Code Folding

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

The `Code Folding` feature enables you to collapse code blocks, reducing them to a single visible line. Folded code is displayed as shaded ellipsis. Folded blocks retain their opening and closing delimiters.

With PhpStorm, you can display source code in a folded way both automatically, and on demand.

In this section:

- [Configuring Autofolding Behavior](#)
- [Folding and Expanding Code Blocks](#)
- [Folding and Expanding Custom Blocks](#)
- [Viewing Contents of a Folded Block](#)

See Also

Reference:

- [Editor. Colors and Fonts](#)
- [Code Folding](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Configuring Autofolding Behavior

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm allows [displaying source code in a folded form](#), making it really concise, and hiding from view the details you might consider less important. When a file is first opened for editing, certain code constructs can be shown folded. It is up to you to define the desired set of code constructs to be folded by default. So doing, when you hover your mouse pointer over the collapsed code fragment, its preview is displayed in a pop-up window.

To configure autofolding behavior in the editor

1. [Open the IDE settings](#).
2. Under the **Editor** node, click **Code Folding**. [Code Folding](#) page is displayed.
3. In the **Collapse by default** list, select the check boxes to the left of the code constructs you want to be displayed collapsed.
4. Apply changes.

Now, when you first open files for editing, the selected code constructs will be shown as shaded brief information, with the preview available at the mouse pointer.

See Also

Reference:

- [Editor. Code Folding](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Folding and Expanding Code Blocks

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Code blocks have folding toggles and outline at the right edge of the gutter next to each block. When a block is folded, the toggle  appears to the left from the first line of the block. When a block is expanded, the code folding outline encompasses all the lines in the block between two fold toggles  .

Folding and expanding code blocks works for the entire classes, method bodies, import lists, comments, HTML and XML tags .

To fold or expand a block of code, do one of the following

- Click the toggles  or .
- Place the caret within the desired block, and choose **View | Folding | Collapse or View | Folding | Expand** on the main menu, or just press **Ctrl+Plus/Minus** **Ctrl Plus/Minus**.

To fold or expand all blocks in a file, do one of the following

- On the main menu, choose **View | Folding | Collapse All**, or **View | Folding | Expand All**.
- Press **Ctrl+Shift+Plus/Minus** **Ctrl Shift Plus/Minus**.

Note

Pressing **Ctrl+Shift+Plus** **Ctrl Shift Plus** once expands all blocks except the imports list. Repeating the keystroke expands the imports.

See Also

Reference:

- [Editor. Code Folding](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Folding and Expanding Custom Blocks

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Custom code blocks can be represented either by an arbitrary selection, or code constructs not marked with the folding toggles and outline (for example, `if`, or `while`).

To fold an arbitrary selected block

1. Select contiguous fragment of code in the editor.
2. On the main menu, choose **View | Folding | Fold Selection / Remove Region**, or press **Ctrl+Period** **Command Period**

See Also

Procedures:

- [Folding and Expanding Code Blocks](#)
- [Viewing Contents of a Folded Block](#)

Getting Started:

- [Selecting Text in the Editor](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Viewing Contents of a Folded Block

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

To view the contents of a folded block

- Hover the mouse cursor over the fold marker, and see the contents in a pop-up window.

See Also

Procedures:

- [Folding and Expanding Code Blocks](#)
- [Folding and Expanding Custom Blocks](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Improving Visibility of the Source Code

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

This section describes how to make your source code more clear and readable:

- [Highlighting Braces](#)
- [Reformatting Source Code](#)
- [Changing Indentation](#)
- [Toggling Case](#)

See Also

Procedures:

- [Code Folding](#)

Reference:

- [Advanced Editing](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Highlighting Braces

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

This editor feature significantly improves readability of the code, and simplifies search for unclosed blocks or tags.

To highlight block borders

- Place the caret immediately after the block closing brace/bracket or before block opening brace/bracket.

If the editor can find the block border, its braces, brackets or tags are highlighted with blue and a blue outline appears in the gutter area.

If the opening brace, bracket or tag is currently out of sight, you don't need to scroll to the beginning of the block. The editor shows a pop-up window on top that displays the beginning of the block.

If the editor cannot find the pair brace, bracket or tag, the unmatched one is highlighted with pink when the caret is placed next to it, and is underlined with a red curly line.

```

43 }
44 )
45

```

See Also

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Reformatting Source Code

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

You can reformat source code to meet the requirements of your code style. PhpStorm will lay out spacing, indents, keywords etc.

Reformatting can apply to the selected text, entire file, or entire project.

To reformat source code

1. On the main menu, choose **Code | Reformat Code**, or press `Ctrl+Alt+L`/`Command Alt L`.
2. In the **Reformat Code** dialog box, specify the reformatting scope:
 - o The current file.
 - o Selected text.
 - o All files in the current directory, including or omitting subdirectories.
3. Click **Run**.

See Also

Reference:

- [Code Style](#)
- [Reformat Code Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Changing Indentation

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts:  

PhpStorm makes it possible to:

- [Indent or unindent](#) text. This action applies to a selection, or to a line at caret.
- Choose [tabs or spaces](#) for indentation. This action applies to a selection, or the whole current file in the active editor.

To change indentation of a text fragment, do one of the following

- On the main menu, choose **Edit | Indent Selection / Edit | Unindent Selection**.
- Press `TabTab` / `Shift+TabShift Tab`.

To toggle between tabs and spaces

- On the main menu, choose **Edit | Convert Indents**, and then choose **To Spaces** or **To Tabs** respectively.

See Also

Getting Started:

- [Selecting Text in the Editor](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Toggling Case

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts:  

To toggle between upper case and lower case

1. Select text fragment, or just place the caret at the line you want to change case in.
2. On the main menu, choose **Edit | Toggle Case**, or press `Ctrl+Shift+U`/`Command Shift U`.

See Also

Getting Started:

- [Selecting Text in the Editor](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Spellchecking

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts:  

PhpStorm helps you make sure that all your source code, including textual strings, comments, and literals, and commit messages, is spelt correctly. For this purpose, PhpStorm suggests a dedicated `Type` inspection, which is supported by the corresponding bundled plugin and is enabled by default.

Correctness of spelling is checked against pre-defined dictionaries (as of now, `jetbrains.dic` and `english.dic`), and any number of user-defined custom dictionaries.

A `user dictionary` is a textual file with the `dic` extension, containing the words you want to be accepted by the `Typo` inspection as correct. The words in such dictionaries are delimited with the newline.

Besides that, you can define your own list of words that will be skipped by the inspection. You can add words to this list "on-the-fly", or intentionally while setting up your spellchecker options.

With the `Typo` inspection enabled, PhpStorm detects and highlights words not included in dictionaries and user's words list. It up to the user to provide correct spelling, accept word as is, or disable inspection.

If a word is accepted, it will be added to the user's words list, and skipped by the spellchecker in future. If inspection is disabled, all typos will be ignored.

In the textual strings and comments, spelling of a word at caret can be changed to a correct one. In the contexts that enable [Rename](#) refactoring, the inspection suggests to rename all occurrences of a symbol.

In this section you will learn how to [perform spellchecking](#), and [configure spellchecker behavior](#).

To spellcheck a word

1. Place the caret on a word highlighted by the `Typo` inspection.
2. Press `Alt+Enter` `Alt Enter` to show the available intention actions.
3. Choose one of the following actions:
 - **Change to:** select the desired spelling of a textual string or comment from the suggestion list.
 - **Save to dictionary:** add word to the user's list and skip it in future.

To configure spellchecking options

1. [Open the Project Settings](#) dialog box
2. Click [Spelling](#).
3. Make up a user's words list. To do that, click **Accepted Words** tab, and use **Add** and **Delete** buttons to create the list of words to be skipped by the `Typo` inspection.
4. Define the set of dictionaries to be used for spellchecking in the **Dictionaries** tab. In the **User dictionaries paths** section, click **Add**, and in the **Select Path** dialog box, point to the director where the desired dictionaries are located. All dictionaries detected in the specified directories, are listed in the **Dictionaries** section below. By default, all dictionaries are enabled. If you want to omit some of them, clear their check boxes.
5. Define the type of contents to be inspected, and inspection severity. To do that, click the **Manage spelling inspection settings** hyperlink to open the `Typo` page in [Inspections](#). Use check boxes to define the pieces of code to be inspected (code, literals, and comments), and select severity level from the drop-down list.

See Also

Procedures:

- [Applying Intention Actions](#)
- [Rename Refactorings](#)

Reference:

- [Spelling](#)
- [Inspections](#)
- [Plugins](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Using TODO Lists

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: 

Working on a large project, you often need to create the lists of tasks, and keep your team mates informed about the issues that require their attention. Such issues can include the questions that should be answered, certain changes that should be done later, areas of optimization and improvement etc.

PhpStorm suggests to use special `TODO` comments in the source code. Such comments can be used in all supported file types, and should match a certain `TODO` pattern. PhpStorm comes with one pre-defined pattern, but you can define as many `TODO` patterns as required. When PhpStorm a matching occurrence is encountered, it is interpreted as a `TODO` item. PhpStorm highlights such comments in accordance with the [Colors and Fonts](#) settings.

Whenever a pattern is changed, or a new pattern is added, PhpStorm scans the whole project and rebuilds the index of `TODO` items. Results display in the [TODO tool window](#).

You might want to view the `TODO` comments of certain a type, and hide the others. For this purpose, PhpStorm suggests to use `filters`. This way you can show those items that match certain patterns only.

In this section:

- [Defining TODO patterns](#)
- [Defining filters](#)
- [Creating TODO items](#)
- [Example](#)

To define a todo pattern, follow these general steps

1. Open the [TODO](#) page of the **Settings** dialog.
2. In the **Patterns** section, click **Add** button to create a new pattern, or **Edit** to change an existing one. [Add/Edit Pattern dialog](#) opens.
3. In the **Pattern** field, enter the regular expression that describes the desired pattern.
4. From the **Icon** drop-down list, select the desired icon that will denote the matching `TODO` items in the [TODO tool window](#).
5. Specify the colors and fonts properties, which PhpStorm will use to highlight the matching comments in the source code.
6. Check the option **Case sensitive**, if you want the pattern match to take into consideration the case of the characters.

To define a filter that will be used to show specific types of todo items

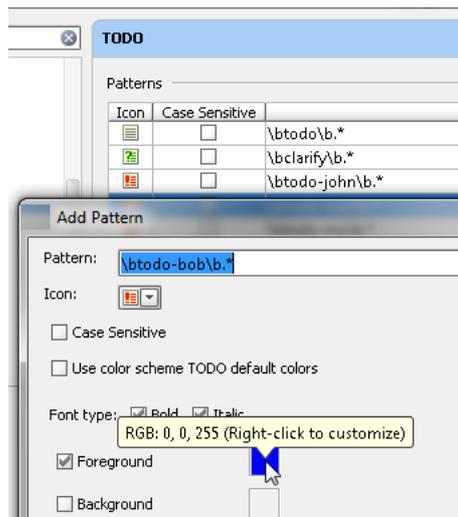
1. Open the [TODO](#) page of the Settings dialog.
2. In the Filters section, click **Add** to create a new filter, or **Edit** to change an existing one.
3. In the [Add/Edit Filter](#) dialog, specify the filter name, and check the patterns to be included in the filter.

To create todo items

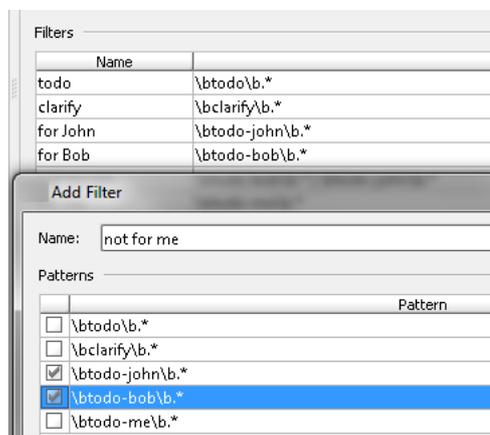
1. Open the desired file in the editor and position the caret at the place where a TODO item should be created.
2. [Create a comment](#). For example, you can use the `Ctrl+Slash` *Command Slash* or `Ctrl+Divide` *Command Divide* keyboard shortcut.
3. In the comment, type the string that matches one of your TODO patterns. By default, any strings that start with `TODO` (regardless of the case) are interpreted as TODO items and highlighted accordingly.
4. View the list of TODO items in the TODO tool window.

Example

In the TODO page of the Settings dialog, click **Add**, and create an additional TODO patterns, for example, `todo-John`, `todo-Bob``todo-me`, with new icons, and custom color schemes:



Next, create several filters, which you will use to show the TODO items, say for each of the developers, and not for your good self. For this purpose, in the **Filters** section, click **Add**, and specify the filter names, for example, `for John`, `for Bob`, and `not for me`. Associate these filters with the patterns:



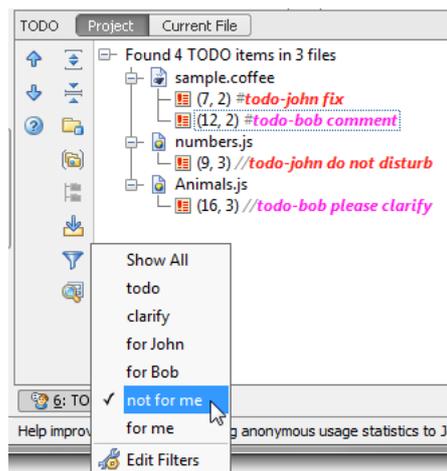
Now, in the source code, create TODO items: in the line of code, where you want to add a note, press `Ctrl+Slash` *Command Slash* or `Ctrl+Divide` *Command Divide*, or `Ctrl+Shift+Slash` *Command Shift Slash* or `Ctrl+Shift+Divide` *Command Shift Divide*, and type `TODO` that matches one of the patterns, followed by some meaningful description:

```

move: (meters) ->
    alert @name + " moved #{meters}m."
#todo-john fix
class Snake extends Animal
move: ->
    alert "Slithering..."
    super 5
#todo-bob comment
class Horse extends Animal
move: ->
    alert "Galloping..."
    super 45
#todo-me take a break
sam = new Snake "Sammy the Green Mamba"
tom = new Horse "Tommy the Palomino"
    
```

Having produced a number of TODO items across the whole project, review them in the TODO tool window: click the button in the side bar to expand the tool window. By default, all the encountered TODO items are displayed.

Let's now show the TODO items for Bob and John, and hide the other items: click the filter icon  on the toolbar of the TODO tool window, and select **not for me** on the submenu:



See Also

Reference:

- [TODO Tool Window](#)
- [TODO](#)
- [Regular Expression Syntax Reference](#)
- [Commit Changes Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

3.0+

Using Macros in the Editor

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Macros provide a convenient way to automate repetitive procedures you do frequently while writing code. You can record, edit and playback macros, assign them a shortcut, and share them. Generally speaking, macros are designed for rather simple operations, and as such have the following limitations:

- Macros can be used for editor-related actions within a file.
- You cannot record such actions as button clicks, navigating to pop-up dialog boxes, and accessing tool windows or menus.

If a macro is intended for temporary use only, it is unnamed; permanent macros have unique names.

This section describes how to:

- [Record Macros](#)
- [Bind Macros With the Keyboard Shortcuts](#)
- [Play Back Macros](#)
- [Edit Macros](#)

See Also

Concepts:

- [Project and IDE Settings](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Binding Macros with Keyboard Shortcuts

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

To bind a macro with a keyboard shortcut

1. On the main menu, choose **File | Settings | IDE Settings | Keymap** for Windows and Linux or **PhpStorm | Preferences | IDE Settings | Keymap** for Mac OS.
2. [Create a new keymap](#) or select an editable keymap from the list of keymaps.
3. Expand the **Macros** node and select the macro for which a keyboard shortcut should be created.
4. Click the **Add Keyboard Shortcut** button.
5. In the **Enter Keyboard Shortcut** dialog, press the keys to be used as a shortcut. The keystrokes are immediately reflected in the **First Stroke** field. Optionally, select the **Enable** check box and specify the **Second Stroke**. As you press the keys, the **Preview** field displays the suggested combination of keystrokes, and the **Conflicts** field displays warnings, if some of the keystrokes are already in use.
6. Click **OK** using the mouse pointer to create a shortcut and bind it with the macro.

Tip

It is important that you use the mouse pointer, because any keystroke is interpreted as a shortcut.

See Also

Procedures:

- [Configuring IDE Settings](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
 - <http://youtrack.jetbrains.net/issues/WI> 
-

Editing Macros

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

After recording, you can remove or rename any or all of the macros from the list of available macros. The actions list of each macro is also editable - you can remove unnecessary actions.

To edit macros

1. On the main menu, choose **Edit | Macros | Edit macros**.
2. In the **Edit Macros** dialog, select the macro to be edited.
3. To change the macro name, click the **Rename** button, and specify the new name in the **Rename Macro** dialog.
4. To change the list of actions for a macro, select an action in the action list, and click **Remove Action**.

See Also

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
 - <http://youtrack.jetbrains.net/issues/WI> 
-

Playing Back Macros

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

You can play back the recorded macros using the **Tools** menu commands, or custom shortcuts.

To play back a temporary macro

- On the main menu, choose **Edit | Macros | Playback Last Macro**.

To play back a named macro

- On the main menu, choose **Edit | Macros | <Macro name>**.

To play back a macro with a keyboard shortcut

1. Select the desired keymap.
2. Press keyboard shortcut that corresponds to the desired macro.

See Also

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
 - <http://youtrack.jetbrains.net/issues/WI> 
-

Recording Macros

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

To record a macro

1. On the main menu, choose **Edit | Macros | Start Macro Recording**. From that moment on, all your recordable actions are recorded.
2. When you are done with the procedure, choose **Edit | Macros | Stop Macro Recording**.
3. In the **Enter Macro Name** dialog, specify the name of the new macro, and click **OK**. If the macro is intended for temporary use only, you can leave the name blank.

See Also

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 

- <http://youtrack.jetbrains.net/issues/WI>

```
ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js'; var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s); }());
```



PhpStorm 3.0.0 Web Help

PhpStorm Tool Windows

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Attached to the bottom and sides of the workspace are PhpStorm `tool windows`. These secondary windows let you look at your project from different perspectives and provide access to typical development tasks. These include project management, source code search and navigation, running and debugging, integration with version control systems, and many other specific tasks.



Click thumbnail to view larger image.

Certain tool windows are available always, that is, in any `project` irrespective of the project nature, contents, and configuration. Other tool windows are available only if the corresponding `plugins` and/or `facets` are enabled. There are also tool windows that require certain specific actions to be made to become available. (For example, to make the `Data Sources tool window` available you have to create a `data source`.)

In this section:

- [Tool window bars and buttons](#)
- [General tool window layout](#)
- [Active Tool Window](#)
- [Accessing Tool Windows Menus](#)

Tool window bars and buttons

Around the tool windows (or the `editor` area if the tool windows are hidden) are the tool window button bars (or just `tool window bars`). These bars contain the buttons for showing or hiding the tool windows (`tool window buttons`). The tool window buttons also provide access to tool window context menus which are shown when the buttons are right-clicked.

Initially, there are three button bars, two at the sides of the main window and one at the bottom. You can show or hide all the button bars at once by clicking in the bottom-left corner of the workspace.

Each tool window button has the name of the corresponding tool window on it. On certain buttons, the window name may be preceded by a number, for example, `1: Project`. This means that the keyboard shortcut `Alt+<number>Alt <number>` is available for showing or hiding the window. You can, for example, show or hide the `Project` tool window by pressing `Alt+1Alt 1`.

You can turn showing the window access numbers on the buttons on and off in the [Appearance settings](#).

The buttons for visible tool windows and for hidden ones have different colors.



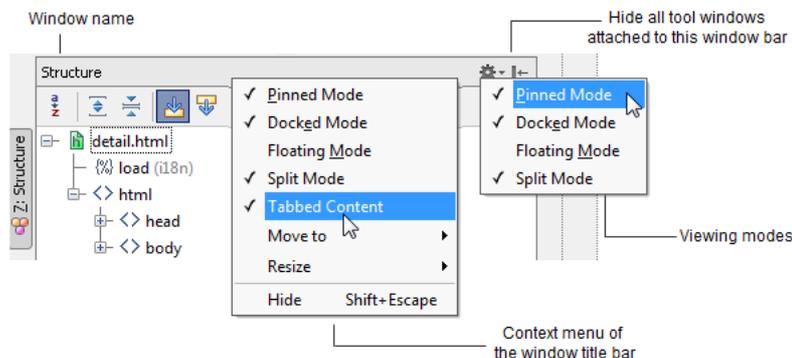
You can rearrange the tool windows by dragging-and-dropping the tool window buttons onto a different tool window bar (or to a different corner of the same bar). As a result, the tool window becomes attached to the bar you've moved the window button to.



General tool window layout

Generally, all the tool windows are organized in similar way.

At the top of the window is a title bar. The title bar contains the buttons that control the tool window [viewing modes](#) and also the ones for hiding individual windows or all the windows attached to the same tool window bar.



The title bar also provides access to the tool window context menu. To access the context menu, right-click the title bar.

Underneath the title bar are the toolbar and content pane. Depending on the window, the toolbar may be above or to the left of the content pane.

The toolbar buttons, generally, are window-specific. However, the windows with similar purposes may contain similar controls on their toolbars.

In most cases, the function associated with a toolbar button may also be accessed from the main or context menu or may have a keyboard equivalent.

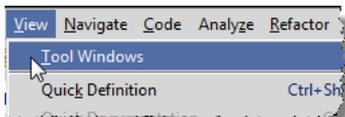
The content panes may be plain or contain two or more "layers" represented, for example, by tabs. There are also tool windows with the content pane part shown on a separate tab in the editor area.

Active tool window

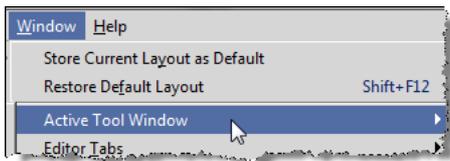
The tool window that currently has the focus, is active.

Accessing tool windows menus

- Use the View | Tool Windows menu to show or hide tool windows:



- Use Window | Active Tool Window menu to show or hide, choose viewing mode, change size of the active tool window:



See Also

Procedures:

- [Manipulating the Tool Windows](#)
- [Viewing Modes](#)
- [Speed Search in the Tool Windows](#)
- [Specifying the Appearance Settings for Tool Windows](#)
- [Configuring Keyboard Shortcuts](#)

Reference:

- [Tool Windows Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Manipulating the Tool Windows

Previous | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

You can manipulate the [tool windows](#) in various ways to adjust the PhpStorm workspace to your needs.

- [Showing a tool window](#)
- [Hiding an individual tool window](#)
- [Hiding all tool windows attached to the same tool window bar](#)
- [Hiding all tool windows](#)
- [Switching to the last active tool window](#)
- [Hiding or showing the tool window bars](#)
- [Attaching a tool window to a different tool window bar](#)
- [Resizing a tool window](#)
- [Increasing the maximum number of tool windows to be shown at a time](#)
- [Saving and restoring the arrangement of the tool windows](#)

To show a tool window, do one of the following

- Click the corresponding tool window button on the [tool window bar](#).
- Choose **View | Tool Windows | <tool window>** on the main menu.
- If the tool window has an associated quick access number, press **Alt+<number>Alt <number>** (for example, **Alt+1Alt 1** for the **Project** tool window).

To hide an individual tool window, do one of the following

- Click the corresponding tool window button on the [tool window bar](#).
- Right-click the corresponding tool window button and select **Hide** from the context menu.
- Right-click the tool window title bar and select **Hide** from the context menu.
- If the tool window has an associated quick access number, press **Alt+<number>Alt <number>** (for example, **Alt+1Alt 1** for the **Project** tool window).

To hide all tool windows attached to the same tool window bar, do one of the following

- On the title bar of any of the tool windows attached to the corresponding tool window bar, click **[-]**.
- Choose **View | Hide Side Tool Windows** in the main menu. This command hides all the tool windows attached to same tool window bar as the active tool window or the last of the active tool windows.

To hide all the tool windows, do one of the following

- Choose **Window | Active Tool Window | Hide All Windows** in the main menu.
- Press **Ctrl+Shift+F12Command Shift F12**.

To switch to the last active tool window, do one of the following

- Choose **Window | Active Tool Window | Jump to Last Tool Window** in the main menu.
- Press **F12F12**.

If all the tool windows are currently hidden, the last active tool window will be shown and made active.

To hide or show the tool window bars, do one of the following

You can hide all the tool window bars if you need more space in the PhpStorm window:

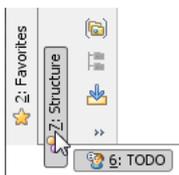
- In the lower left corner of the workspace, click **[☰]**.

If the tool window bars are hidden, you can bring them back onto the screen either permanently or for a short period of time:

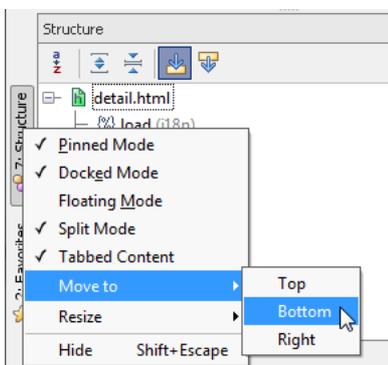
- To restore the tool window bars, click **[☰]** in the lower left corner of the workspace.
- To see the tool window bars for a short period of time, double-press and hold the **AltCommand** key. The tool window bars appear on the screen making the tool window buttons accessible. The tool window bars are hidden again when you release the key.

To attach a tool window to a different tool window bar, do one of the following

- Drag-and-drop the corresponding tool window button onto the desired tool window bar.



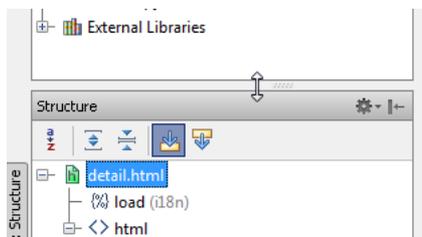
- Right-click the corresponding tool window button or the tool window title bar to open the context menu. Choose **Move to** and then select the destination tool window bar (**Top**, **Left**, **Bottom** or **Right**).



To resize a tool window

The obvious way to resize a tool window is this:

- Hover the mouse pointer over the tool window border. When the pointer becomes a double-headed arrow, drag the border in the required direction.



You can also resize a tool window by moving its border to left or right, or up or down in steps. The following alternatives are available for doing that:

- Right-click the corresponding tool window button or title bar and select **Resize**. Then select one of the available **Stretch to** options.
- Make the tool window of interest active and do one of the following:
 - Choose **Window | Active Tool Window | Resize**, and then the necessary **Stretch to** option.
 - Press **Ctrl+Shift+Ctrl Shift** in combination with the corresponding arrow key.

To increase the maximum number of tool windows to be shown at a time

To increase the maximum number of tool windows to be shown at a time, you should appropriately set the [viewing modes](#) for different tool windows. Consider the following:

- Generally, for a tool window to be visible always (i.e. even when inactive), the tool window should be [pinned](#).
- There are no limiting factors for the number of [floating](#) windows shown simultaneously.
- To be able to see two windows [docked](#) to the same tool window bar at a time (rather than one), one of the windows should have the [split mode](#) off and the other one on.
- Initially, three (out of four) tool window bars are used. You can "activate" the fourth tool window bar (the top one) by [moving](#) certain tool windows to it.

To save or restore the way the tool windows are arranged

You can save the way the tool window are currently arranged by choosing **Window | Store Current Layout as Default** in the main menu.

At a later time, you can return to the saved workspace layout by choosing **Window | Restore Default Layout**.

See Also

Procedures:

- [Viewing Modes](#)
- [Speed Search in the Tool Windows](#)
- [Specifying the Appearance Settings for Tool Windows](#)

Reference:

- [Keyboard Shortcuts and Mouse Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Specifying the Appearance Settings for Tool Windows

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

You can change certain tool window appearance properties by specifying the corresponding Appearance settings.

To change the appearance properties for tool windows

1. [Open the IDE Settings](#) and click **Appearance**.
2. If necessary, change the settings related to tool window appearance. These are mainly in the **Transparency** and the **Window Options** sections. For more information, see the description of the [Appearance page](#).

See Also

Procedures:

- [Manipulating the Tool Windows](#)
- [Viewing Modes](#)
- [Speed Search in the Tool Windows](#)

Reference:

- [Tool Windows Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Speed Search in the Tool Windows

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts:

Speed search in the tool windows helps you find and navigate to a file or folder in the **Project** tool window, a member in the **Structure** tool window, a changelist in the **Changes** tool window, an item in the **TODO** list, and more.

To search through a tool window

1. Select the desired tool window.
2. Start typing the item name (for instance, file, class, field, etc.). As you type, the **Search for** field appears over the tool window toolbar showing the entered characters, and the element selection moves to the first item that matches the specified string.
3. Press **Enter** when ready. As a result, the matching item is selected in the tool window. Pressing **Escape** hides the **Search for** field.

See Also

Procedures:

- [Manipulating the Tool Windows](#)
- [Viewing Modes](#)
- [Specifying the Appearance Settings for Tool Windows](#)

Reference:

- [Tool Windows Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Viewing Modes

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm provides various viewing modes that let you control the way the tool windows are shown and behave. These modes help you keep a proper balance between quick, easy access to tool windows and maximum screen space for editing your code.

The viewing modes are set separately for each of the tool windows.

- [Ways to control the viewing modes](#)
- [Fixed / Floating Mode](#)
- [Docked / Undocked Mode](#)
- [Pinned / Unpinned Mode](#)
- [Split Mode](#)
- [Tabbed Content](#)

Ways to control the viewing modes

PhpStorm suggests the following means to control the viewing modes:

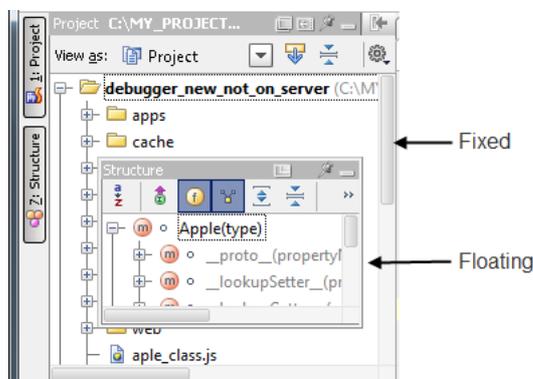
- Buttons on the title bar of a tool window. Such buttons are available only for the most important viewing modes.
- Context menu options. (The context menus are accessed by right-clicking the tool window buttons or the tool window title bars).
- Options in the **Window | Active Tool Window** menu. These are available only if the corresponding tool window is currently active.

Note

All the corresponding controls are toggles.

Fixed / floating mode

A tool window may be **fixed** or **floating**.



When in the fixed mode, one side of the tool window is attached to one of the tool window bars. The behavior of the opposite side depends on whether the window is [docked or undocked](#).

Initially, all the tool windows are in the fixed mode (i.e. the floating mode is off).

In the floating mode, a tool window becomes independent of all other windows and may be moved around the screen to any place.

When a floating tool window becomes inactive, it may turn semi-transparent under the following conditions:

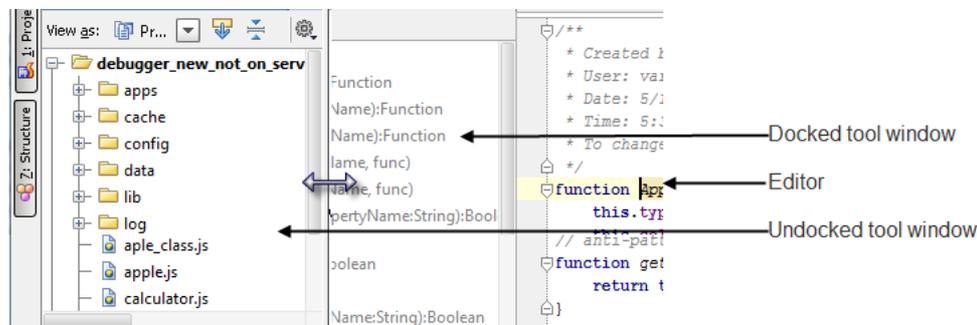
- The window is [pinned](#).
- The corresponding transparency option is on in the [Appearance settings](#).

To switch between the fixed and floating mode, do one of the following

- Click  on the tool window title bar, and then select or clear the **Floating Mode** toggle on the drop-down menu.
- Select or clear the **Floating Mode** toggle on the context menu of the tool window button or tool window title bar.
- On the main menu, choose **Window | Active Tool Window**, and then select or clear the **Floating Mode** toggle.

Docked / undocked mode

A tool window in the **fixed mode** may be docked or undocked.



In the **docked mode**, all the sides of a tool window are attached to surrounding elements (the editor, other tool windows, etc.) Thus, the tool window and the adjacent elements share the space available in the main window.

When a docked tool window becomes inactive, it stays visible or is hidden depending on whether the window is [pinned or unpinned](#).

Initially, all the tool windows are in the docked mode (i.e. the docked mode is on).

When **undocked**, all the sides of a tool window (except the one at the tool window bar) are detached from surrounding elements. The window moves to the "upper layer" covering the elements it use to share the space with. In one of the directions (along the tool window bar), the window stretches and takes all the available space. In the other direction, one of the window borders becomes loose and can be moved without affecting the sizes of other, underlying elements.

When an undocked tool window becomes inactive, it is automatically hidden.

To switch between the docked and undocked mode, do one of the following

- Click  on the tool window title bar, and then select or clear the **Docked Mode** toggle on the drop-down menu.
- Select or clear the **Docked Mode** toggle on the context menu of the tool window button or tool window title bar.
- On the main menu, choose **Window | Active Tool Window**, and then select or clear the **Docked Mode** toggle.

Pinned / unpinned mode

Pinned tool windows, generally, stay visible when becoming inactive. **Unpinned** tool windows in such cases are automatically hidden.

Initially, all the tool windows are pinned (i.e. the pinned mode is on).

There may be slight differences in behavior depending on the other viewing modes:

- [Undocked](#) tool windows are always hidden when inactive. (In the undocked mode, the tool windows are effectively unpinned.)
- [Floating](#) pinned tool windows, when inactive, may become semi-transparent.

To switch between the pinned and unpinned mode, do one of the following

- Click  on the tool window title bar, and then select or clear the **Pinned Mode** toggle on the drop-down menu.
- Select or clear the **Pinned Mode** toggle on the context menu of the tool window button or tool window title bar.
- On the main menu, choose **Window | Active Tool Window**, and then select or clear the **Pinned Mode** toggle.

Split mode

This mode has to do with how many windows [docked](#) to the same tool window bar may be shown at a time (one or two).

Generally, the space along a tool window bar is shared between two groups of docked tool windows.

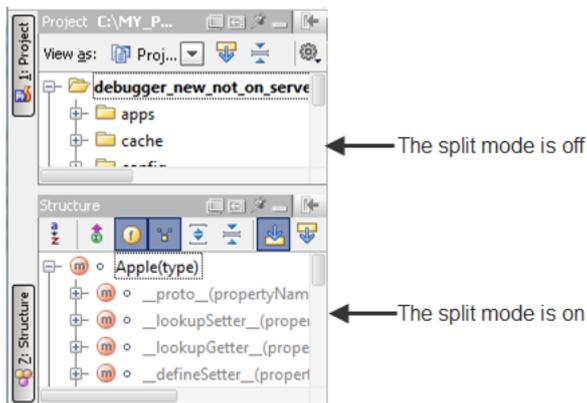
In one of the groups are the tool windows for which the split mode is off; in the other group are the ones with this mode on.

At each moment of time, only one window from each of the groups may be visible.

Thus, if all the tool windows docked to a tool window bar have the split mode off, only one tool window may be shown at a time. In this case, the tool window which is visible takes all the space available along the tool window bar. So when you make a certain window visible, the previous visible window is automatically hidden.

The same behavior is observed if the split mode is on for all the windows docked to the same tool window bar.

To be able to see two windows simultaneously, the corresponding windows should belong to different groups, that is, one of the windows should have the split mode off and the other one on.



The tool window buttons for the tool windows with different settings of the split mode are grouped and shown at different corners of the corresponding tool window bar. For vertical window bars, the windows with the split mode off have the buttons at the top corner; for the horizontal bars, the buttons for such windows are at the left corner.

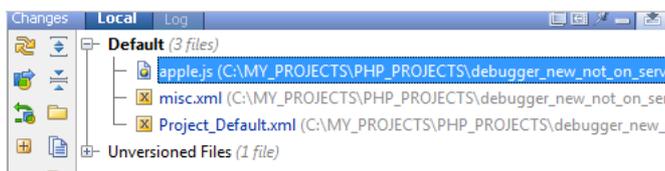
To turn split mode on or off, do one of the following

- Click on the tool window title bar, and then select or clear the **Split Mode** toggle on the drop-down menu.
- Select or clear the **Split Mode** toggle on the context menu of the tool window button or tool window title bar.
- On the main menu, choose **Window | Active Tool Window**, and then select or clear the **Split Mode** toggle.

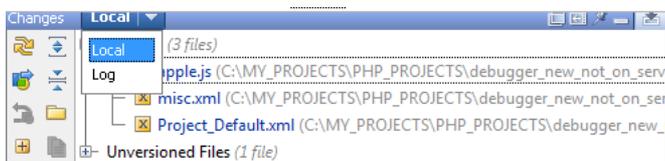
Tabbed content

This mode controls the way how different layers are shown and selected in "multilayer" tool windows:

- If the tabbed content mode is on, the layers are shown as tabs and you can switch between the layers by selecting the necessary tab.



- If the tabbed content mode is off, the layers are shown as panes. The necessary layer (pane) is selected from the corresponding list.



To turn the tabbed content mode on or off, use the **Tabbed Content** toggle in the context or the **Window** menu.

See Also

Procedures:

- [Manipulating the Tool Windows](#)
- [Speed Search in the Tool Windows](#)
- [Specifying the Appearance Settings for Tool Windows](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Compare Files and Folders

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm helps explore differences in the various situations: differences between files, directories, revisions of the same file under version control or in the local history, database objects, local and remote files.

All these operations are performed in a similar way. In this section we'll consider the most basic operations:

- [Comparing Files](#)
- [Comparing Folders](#)

See Also

Procedures:

- [Comparing Data Sources](#)
- [Comparing Deployed Files and Folders with Their Local Versions](#)
- [Viewing Local History of a File or Folder](#)
- [Handling Differences](#)

Reference:

- [Differences Viewer](#)
- [Differences Viewer for Folders and DB Objects](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Comparing Files

Previous | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

PhpStorm enables you to compare [two arbitrary files](#) in a project (including image files), or [compare a file in the editor with the Clipboard contents](#), using the [Differences viewer](#).

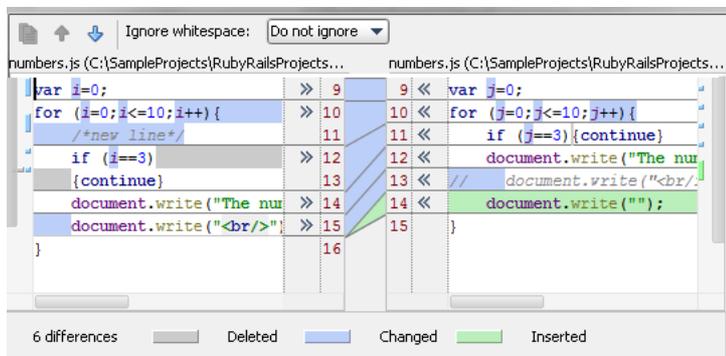
To compare two files

1. Keeping the `Ctrl+Ctrl` key pressed, click two files in the Project tool window.
2. On the context menu of the selection, choose **Compare Two Files**. The [Differences Viewer for Files](#) opens, with the differences being color-highlighted.

Tip

It is enough to select a single file in the Project tool window. In this case the context menu command is **Compare File with Editor**, and the Differences Viewer shows the contents of the selected file on the left pane, and the contents of the active editor tab on the right pane.

3. View the differences and apply them, if necessary, using the buttons `>>` and `<<`.



To compare a file with the clipboard contents

1. Open the desired file in the editor.
2. Right-click the editor pane and choose **Compare with Clipboard** on the context menu.
3. View and manage differences in the [Differences Viewer for Files](#).

See Also

Procedures:

- [Comparing File Versions](#)
- [Comparing Deployed Files and Folders with Their Local Versions](#)

Reference:

- [Differences Viewer](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Comparing Folders

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

PhpStorm provides a dedicated [Differences Viewer for Folders](#) for comparing files in two folders against the file size, content, or timestamp. The Differences Viewer shows the contents of the selected directories in the left and right panes of the Item List. The contents of the selected file are shown in the lower pane, with the differences being color-highlighted.

Besides exploring differences, the tool also provides interface for synchronizing the contents of folders.

In this section:

- [Opening the Difference Viewer](#)
- [Comparing two folders in the Difference Viewer](#)
- [Synchronizing contents of folders](#)

To open the difference viewer, do one of the following:

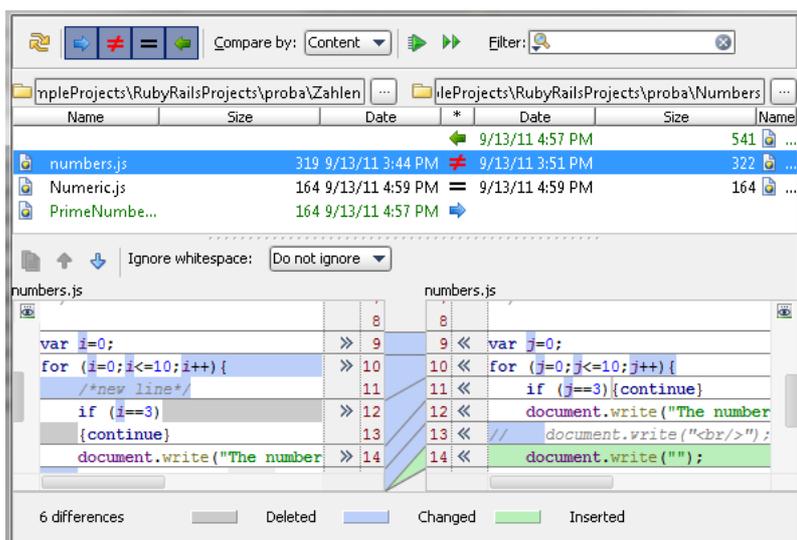
- Keeping the **Ctrl+Ctrl** key pressed, click two directories in the Project tool window, and choose **Compare Directories** on the context menu of the selection, or press **Ctrl+D**/**Command D**.
- Select a directory in the **Project** tool window, choose **Compare with** on the context menu of the selection, and then select the second directory in the **Select Path** dialog box, that opens.

Tip

You can also open the difference viewer without running PhpStorm. This is done through the following command: `<path to PhpStorm executable file> diff <path_1> <path_2>` where `path_1` and `path_2` are paths to the folders in question.

To compare two folders in the difference viewer, perform these general steps:

- Configure the layout of the **Items List**. Use the [toolbar buttons](#) to narrow down or widen the set of items to show. For example, show or hide files that exist in just one of the directories, equal files, or different files, etc.
- Specify the parameter for comparison. In the **Compare by** drop-down list, select one of the possible options (contents, size, or time stamp).
- Filter the folders' contents. To do that, type filtering string in the **Filter** text field, and press **Enter/Enter** to apply it. Using the asterisk `*` wildcard to represent any number of characters is welcome.
- To switch to another pair of folders to compare, update the fully qualified paths to them. Click the **Browse** button  next to the **Paths** read-only fields and choose the required folders in the **Select Path** dialog box, that opens.
- Explore the detected differences between files in the **Differences Pane**.



To synchronize the contents of folders

1. For each pair of items, in the `*` field specify the action to apply. Click the icon in the field until the required action is set.

Icon Action

 -  Copy the item in the left side to the right side, possibly overwriting the contents of the corresponding target item, if it already exists.
 -  Copy the item in the right side to the left side, possibly overwriting the contents of the corresponding target item, if it already exists.
 -  The items are treated identical with regard to the selected criterion of comparison. No action will be performed by default.
 -  The items differ with regard to the selected criterion of comparison. No action will be performed by default. Explore the differences in the [Differences Pane](#) and change the intended action by clicking the icon.
 -  The item is present only in one of the folders and will be removed.
2. Do one of the following:
 - To synchronize the currently selected item, click the **Synchronize Selected** button  on the toolbar.
 - To synchronize all the items, click the **Synchronize All** button  on the toolbar.

See Also

Procedures:

- [Viewing Local History of a File or Folder](#)
- [Comparing Data Sources](#)
- [Comparing Deployed Files and Folders with Their Local Versions](#)

Reference:

- [Differences Viewer for Folders and DB Objects](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

```
ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js'; var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s); }());
```

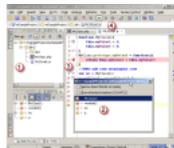


PhpStorm 3.0.0 Web Help

Familiarize Yourself with IDE Navigation

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm lets you walk through your projects and files in a wide variety of ways. Dedicated views and tool windows let you examine your code at any level - from details to the top-level structure.



[Click thumbnail to view larger image.](#)

This sample displays some of the navigation elements:

1. Project View
[Project view](#) shows the top-level project structure.
2. Structure
[Structure](#) showing the drill down of document elements or class members.
3. Gutter Icons
Active **Gutter Icons** you can use to [jump between an element's source and usage.](#)
4. Navigation Bar
[Navigation bar](#) lets you quickly navigate through the project file structure.

Refer to the section [Navigating Through the Source Code](#) for more information.

See Also

Procedures:

- [Navigating Through the Source Code](#)
- [Navigating with Navigation Bar](#)
- [Navigating with Structure Views](#)

Reference:

- [Keyboard Shortcuts by Category](#)
- [Tool Windows Reference](#)

Getting Started:

- [Guided Tour Around PhpStorm User Interface](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Copy and Paste Between PhpStorm and Explorer/Finder

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

PhpStorm enables tight interaction with the native file managers (Explorer on Windows, or Finder on Mac), and allows you exchange files and directories via the system clipboard, using numerous techniques:

- Cut, copy and paste files and directories from the Project tool window of PhpStorm to a directory in the file manager, and vice versa, using menu commands and keyboard shortcuts:

Keyboard shortcut	Function	Use this shortcut to...
Ctrl+C Command C or Ctrl+Insert Command Insert	Copy	Copy selected text to the Clipboard.
Ctrl+X Command X or Shift+Delete Shift Delete	Cut	Cut to the Clipboard.
Ctrl+V Command V or Shift+Insert Shift Insert	Paste	Paste from the Clipboard.

- Move (drag) or copy (Ctrl+drag) a file or directory from the file manager to a directory in the Project tool window.
- Move (drag) or copy (Ctrl+drag) a file from the Project tool window to a directory in the file manager.
- Open any file for editing, by dragging it from a file manager to the editor.

See Also

Procedures:

- [Opening and Reopening Files in the Editor](#)

- [Copy/Clone](#)
- [Move Refactorings](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Configure Your Working Environment

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm IDE can be fully customized and fine-tuned to match your every specific habit. Syntax coloring and error highlighting, dialog and tool window styles, keyboard shortcuts, menu and toolbar commands - everything can be configured to ensure that you always have what you need in just few keystrokes away. For your convenience, the entire set of IDE configuration options is available from the [Settings dialog](#).

The part [Configuring IDE Settings](#), describes:

- [Accessing the IDE Settings](#)
- [Configuring Colors and Fonts](#)
- [Configuring Keyboard Shortcuts](#)
- [Configuring Menus and Toolbars](#)
- [Configuring Quick Lists](#)
- [Configuring Third-Party Tools](#)
- [Creating and Editing File Templates](#)
- [Creating and Registering File Types](#)
- [Creating and Editing Live Templates](#)
- [Creating and Editing Template Variables](#)
- [Exporting and Importing Settings](#)
- [Switching Between Schemes](#)

See Also

Procedures:

- [Configuring IDE Settings](#)

Reference:

- [Settings Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

```
ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js'; var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s); }());
```



PhpStorm 3.0.0 Web Help

Create and Run Your First PHP Project

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

To get familiar with PhpStorm it is recommended that you create your first project from scratch and implement the very basic functionality in it.

Creating a living PHP application that enables you to view results in a browser of your choice requires the following preconditions to be fulfilled:

- A local [Web server](#) is installed, configured, and running on your computer or access to a remote server is configured.

Note

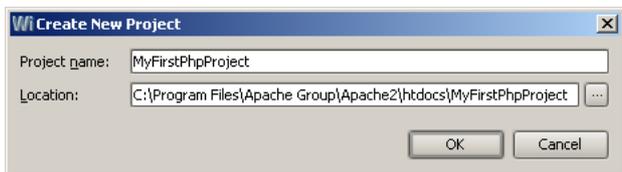
This topic supposes that the local [Apache HTTP server](#) is used and its root directory is `.\htdocs`.

- The [PHP engine](#) is installed and configured.

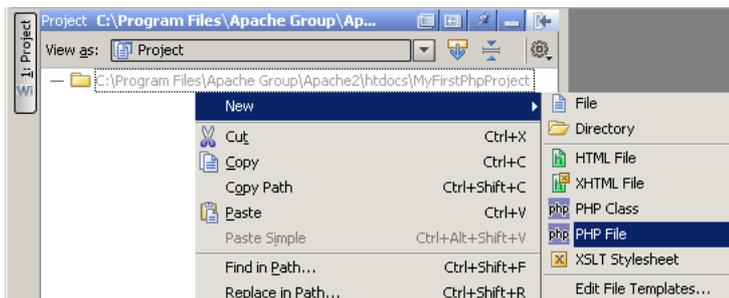
You can install each component separately and then configure the environment. Alternatively, install an AMP package compatible with the operating system you use to have the components and the entire environment configured without any steps from your side.

To create and run your first php project

1. Create a project. For that, choose **File | Create New Project** on the main menu.
2. In the **Create New Project** dialog box that opens specify the name of the project in the **Name** text box, for example, type `MyFirstPhpProject`.
3. Click the **Browse** button next to the **Location** text box.
4. In the **Select Path** dialog box that opens select the path to the `.\htdocs` folder and click **OK**. PhpStorm composes the path to the project folder as follows:



5. Create a PHP file. To do that, right-click the project directory in the [Project tool window](#), point to New on the context menu, and choose PHP File.



6. In the **New PHP File** dialog box that opens type **MyFile** and click **OK**. PhpStorm will create the stub file for you and open it in the dedicated editor tab.

7. Type this sample code:

```
echo phpinfo();
```

8. To save the file, choose **File | Save All** or press **Ctrl+S** or **Command S**.

9. Open a Web browser and type the following URL address:

```
http://localhost:<port>/MyFirstPhpProject/MyFile.php
```

The PHP Information page opens displaying the PHP engine configuration settings on your computer.



[Click thumbnail to view larger image.](#)

See Also

Procedures:

- [PHP-Specific Guidelines](#)

Reference:

- [Create New Project](#)

Getting Started:

- [Guided Tour Around PhpStorm User Interface](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

```
ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js'; var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s); });
```



PhpStorm 3.0.0 Web Help

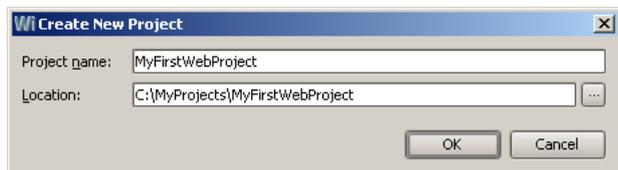
Create and Run Your First Web Project

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

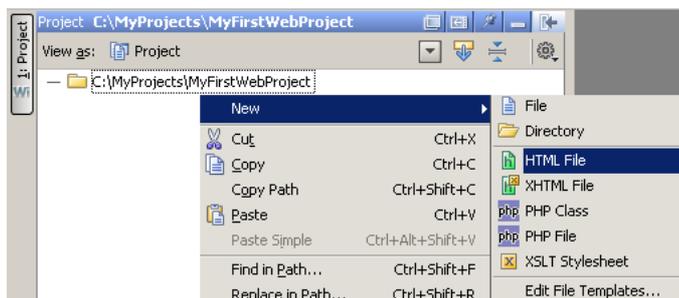
To get familiar with PhpStorm it is recommended that you create your first project from scratch and implement the very basic functionality in it.

To create and run your first web project

1. Create a project. For that, choose **File | Create New Project** on the main menu.
2. In the [Create New Project](#) dialog box that opens specify the name of the project in the Name text box, for example, type `MyFirstWebProject`.
3. In the **Location** text box, specify the folder to create the new project folder in. Type the path manually or click the **Browse** button  next to the text box. Then select the desired folder in the **Select Path** dialog box that opens. PhpStorm composes the path to the project folder as follows:



4. Create an HTML file. To do that, right-click the project directory in the [Project tool window](#), point to New on the context menu, and choose HTML File.



5. In the New HTML dialog box that opens type MyFile and click OK. PhpStorm will create the stub file for you and open it in the dedicated editor tab.
6. Type this sample code inside the <html /> tag:

```
<html>
<head>
<script src="http://maps.google.com/maps?file=api"></script>
<title>

    Your first Web project
</title>
</head>
<body>
<div id="map" style="width: 400px; height: 300px"></div>
<label for="latitude">Latitude:</label>
<input type="text" id="latitude" value="59.980473"/><br/>
<label for="longitude">Longitude:</label>
<input type="text" id="longitude" value="30.324068"/><br/>
<input type="submit" value="Show map" onclick="showMap(document.getElementById('latitude').value,
document.getElementById('longitude').value)"/>
    <script type="text/javascript">
functionshowMap (latitude, longitude)

    {
var map = new GMap(document.getElementById("map"));
map.centerAndZoom(new GPoint(longitude, latitude), 1);

    }
</script>
</body>
</html>
```

Tip

It is recommended that you type this code manually to experience coding assistance provided by PhpStorm.

7. Save the file by choosing File | Save All or pressing Ctrl+S/Command S.
8. Run your application. Do one of the following:
- o Choose View | Web Preview, then select the desired browser from the list.
 - o Click the desired browser on the browser toolbar:



9. In the page that opens in the browser, click the Show Map button. The Google map, that is displayed, shows the location of the IntelliJLabs office in Saint-Petersburg.



[Click thumbnail to view larger image.](#)

See Also

Procedures:

- [Previewing Pages with Web Contents in a Browser](#)

Reference:

- [Create New Project](#)

Getting Started:

- [Guided Tour Around PhpStorm User Interface](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Keyboard Shortcuts You Cannot Miss

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

PhpStorm as a keyboard-centric IDE suggests a wide variety of keyboard shortcuts. In this topic you can find a short list of the most indispensable of them, to make your first steps with PhpStorm easy.

See the detailed list of default keyboard shortcuts in the [Keyboard Shortcuts Reference](#) and learn how to customize your preferred keymap in the section [Configuring Keyboard Shortcuts](#).

Shortcut	Description
Alt+F1Alt F1	Switch between views (Project, Structure, etc.).
Ctrl+TabAlt Tab	Switch between the tool windows and files opened in the editor.
Alt+HomeAlt Home	Show the Navigation bar.
Ctrl+JCommand J	Insert a live template.
Ctrl+Alt+JCommand Alt J	Surround with a live template.
F4F4	Edit an item from the Project or another tree view.
Alt+EnterAlt Enter	Use the suggested quick fix.
Ctrl+Slash Command Slash Or Ctrl+Divide Command Divide	Comment or uncomment a line or fragment of code with the line or block comment.
Ctrl+Shift+Slash Command Shift Slash Or Ctrl+Shift+Divide Command Shift Divide	
Ctrl+NCommand N	Find class or file by name.
Ctrl+Shift+NCommand Shift N	
Ctrl+DCommand D	Duplicate the current line or selection.
Ctrl+WCommand W and Ctrl+Shift+WCommand Shift W	Incremental expression selection.
Ctrl+F Command F Or Alt+F3 Alt F3	Find text string in the current file.
Ctrl+Shift+FCommand Shift F	Find in the current folder.
Ctrl+Shift+F7Command Shift F7	Quick view the usages of the selected symbol.
Ctrl+Add Command Add Or Ctrl+Equals Command Equals	Expand or collapse a code block.
Ctrl+Subtract Command Subtract Or Ctrl+Minus Command Minus	
Ctrl+SpaceCommand Space	Invoke code completion.

See Also

Procedures:

- [Configuring IDE Settings](#)
- [Navigating Through the Source Code](#)

Reference:

- [Keymap](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Basic Concepts

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm is an advanced IDE. To get the most out of its capabilities and features, you should be familiar with its concepts. Concepts describe the basic notions of the IDE.

In this part:

- [Project](#)
- [Project and IDE Settings](#)
- [Contents](#)
- [Scope](#)

- [Supported Languages](#)
- [Live Templates](#)
- [File Templates](#)
- [Running and Debugging](#)
- [Encoding](#)
- [External Tools](#)
- [Plugins](#)
- [Version Control with PhpStorm](#)

See Also

Procedures:

- [PhpStorm Usage Guidelines](#)

Reference:

- [Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>
-

Project

Previous | [Next](#) | [See Also](#) | [Comments](#)

Everything you do in PhpStorm, is done within the context of a project. A project represents a complete software solution, and defines project-wide settings.

In this section:

A project in PhpStorm is represented in the `Directory Based Format`. A project directory is marked with a  icon.

Such project directory contains a `.idea` directory, with the following files:

- `*.iml` file that describes the project structure.
- `workspace.xml` that contains your workspace preferences.
- A number of `.xml` files. Each `.xml` file is responsible for its own set of settings, that can be recognized by its name: `projectCodeStyle.xml`, `encodings.xml`, `vcs.xml` etc.

Thus, for example, adding a new run/debug configuration and changing encoding will affect different `.xml` files. This helps avoid merge conflicts when the project settings are stored in a version control system and modified by the different team members.

Note

The file in `.idea` directory that stores your local preferences (`workspace.xml`) file should not be placed under version control. All the other settings files should be shared.

Note

`.idea` directory is not visible in the Project view.

See Also

Procedures:

- [Creating and Managing Projects](#)

Reference:

- [Project Settings](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>
-

Project and IDE Settings

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

There are two types of settings that define your preferred environment:

- **Project Settings**, which apply to a specific project.
- **IDE Settings**, which are common to all projects and refer to the project-independent aspects.

Project settings are stored with each specific project; **IDE settings** are stored in the dedicated directories, depending on the platform. For example:

Windows

- `<User home>\.<product name>\config` that contains user-specific settings.
- `<User home>\.<product name>\system` that stores PhpStorm data caches.

`<User home>` in WindowsXP, is `C:\Documents and Settings\<User name>`; in Windows Vista it is `C:\Users\<User name>`

Linux

- `~/.<product name>/config` that contains user-specific settings.
- `~/.<product name>/system` that stores PhpStorm data caches.

Mac OS

- `~/Library/Application Support/.<product name>` contains the catalog with plugins.
- `~/Library/Preferences/.<product name>` contains the rest of the configuration settings.
- `~/Library/Caches/.<product name>` contains data caches, logs, local history, etc. These files can be quite significant in size.

The `config` directory has several subfolders that contain xml files with your personal settings. You can easily share your preferred keymaps, color schemes, etc. by copying these files into the corresponding folders on another PhpStorm installation. Prior to copying, make sure that PhpStorm is not running, because it can erase the newly transferred files before shutting down.

The following is the list of some of the subfolders under the `config` folder, and the settings contained therein.

Folder name	User Settings
<code>codestyles</code>	Contains code style schemes.
<code>colors</code>	Contains editor colors and fonts customization schemes.
<code>filetypes</code>	Contains user-defined file types.
<code>inspection</code>	Contains code inspection profiles.
<code>keymaps</code>	Contains IDE keyboard customizations.
<code>options</code>	Contains various options, for example, feature usage statistics and macros.
<code>templates</code>	Contains user-defined live templates.
<code>tools</code>	Contains configuration files for the user-defined external tools.

See Also

Procedures:

- [Configuring IDE Settings](#)
- [Configuring Project Settings](#)
- [Importing PhpStorm Settings](#)

Reference:

- [Settings Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Contents

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

`Content` is a set of roots that can belong to one of the following types:

- `Content roots` contain the actual source files within a project. PhpStorm treats the files under content roots according to their type: they are involved in indexing, they can be parsed, inspected, and interpreted.
- `Excluded roots` contain files and folders ignored by PhpStorm when indexing, searching, parsing, watching etc. These roots are marked with the  icon. Excluded roots are not visible to PhpStorm. Usually, one would like to exclude temporary build folders, generated output, logs, and other project output. Excluding the unnecessary paths is a good way to significantly improve performance.
- `Resource roots` contain folders with Web resources. These roots are marked with the  icon. Marking a folder as a resource root provides proper code completion and analysis.

See Also

Procedures:

- [Configuring Content Roots](#)

Reference:

- [Directories](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Scope

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

A scope is a certain area, where you inspect, search or analyze source code, apply file coloring . A scope can consist of a single file, one or more classes or files, folders, or a whole project.

Scopes can be shared or local:

- `Shared scopes` are accessible for the team members and are stored on the project level in the `misc.xml` file.
- `Local scopes` are intended for personal use only and are stored in your workspace `workspace.xml`

PhpStorm provides a special language that enables you to flexibly define the sets of entities, included in a scope. See section [Scope Language Syntax Reference](#) for details. Scopes are editable in the

[Scopes](#) dialog.

Moreover there are several pre-defined scopes that you can use to find usages, for example:

- Project Files: includes source files and tests. Project libraries are not included in this scope.
- Project Production Files: includes project sources. Libraries and tests are not included.
- Project Test Files: includes project test files. Source files and libraries are not included.
- Project and Libraries: includes source files, tests and libraries.
- Current File.
- Changed Files.

See Also

Concepts:

- [Code Inspection](#)

Reference:

- [Scopes](#)
- [File Colors](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Supported Languages

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Development of a modern application involves using multiple languages, that is why PhpStorm is an IDE for polyglot programming. With the deep understanding of all the subtleties of the source code structure and syntax, PhpStorm extends its support to:

- [JavaScript](#). Refer to the section [JavaScript-Specific Guidelines](#).
- [CoffeeScript](#). Refer to the section [CoffeeScript Support](#).
- HTML/XHTML. Refer to the section [Markup Languages and Style Sheets](#).
- XML. Refer to the section [Markup Languages and Style Sheets](#)
- XSLT. Refer to the section [XPath and XSLT Support](#)
- 2.0+ CSS: coding assistance and compilation for [LESS](#); basic support of [SASS 3](#).

Refer to [Markup Languages and Style Sheets](#).

- [PHP](#). Refer to the section [PHP-Specific Guidelines](#).
- [SQL](#). Refer to [Relational Databases](#).

Coding assistance in PhpStorm includes:

- Syntax and error highlighting. The color attributes are configurable in the [Colors and Fonts | <language>](#) page of the [Settings](#) dialog.
- [File templates](#) for the supported languages that enable creating stub classes, scripts etc.
- [Live templates](#) for creating complicated code constructs.
- [Code completion](#).
- [Code generation](#).
- [Code folding, formatting, and highlighting](#).
- [Intention actions and quick fixes](#).
- [Possibility to view code hierarchy](#).
- [Quick access to the API documentation](#).
- [Using macros in the editor](#).
- [Advanced search and replace facilities](#).
- [Advanced means of navigation](#).
- [Refactoring](#).
- [Import Assistance](#)

Besides editing assistance, PhpStorm features a JavaScript [debugger](#). For PHP projects, [debugging with XDebug](#) is supported.

Warning

Debugging for JavaScript applications is supported only in the [Firefox](#) and [Chrome](#) browsers.

See Also

Reference:

- [Code Style](#)
- [File Types](#)

External Links:

- [PHP](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

PHP Support

Previous | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

PHP support includes:

- Possibility to [create PHP files and classes from templates](#).
- Full [PHP 5.3](#) syntax support.
- [Syntax highlighting](#).
- [Error highlighting](#).
- [Basic on-the-fly code completion](#).
- Resolution of `include` statements and file references.
- [Class Completion](#).
- [Intention actions and quick fixes](#).
- [Surrounding](#) with code constructs `Ctrl+Alt+J` and `Ctrl+Alt+T`.
- [Code inspections](#).
- [Jump to declaration](#) (`Ctrl+B`).
- [Refactoring](#):
 - [Rename](#) (`Shift+F6`).
 - [Move](#) (`F6`).
 - [Copy](#) (`F5`).

Note

PHP coding assistance is provided via a bundled PHP plugin that is enabled by default. If not, activate it on the [Plugins](#) page of the [Settings](#) dialog box.

Syntax highlighting

```
class Concatenation {
    public function concatenate($string1, $string2) {
        $result = $string1.$string2;
        return $result;
    }
}
```

Error highlighting

```
class Concatenation {
    public function concatenate($string1, $string2) {
        $result = $string1.$string2;
        return $result;
    }
}
```

Expected: semicolon

Basic on-the-fly code completion

```
class Concatenation {
    public function concatenate($string1, $string2) {
        $result = $string1.$string2;
        $
    }
}
```

Completion list:

- \$result string
- \$string1
- \$string2
- Concatenation hierarchy.php
- \$name string calligra.php
- \$_ADODB_ACTIVE_DB\$ array adodb-active-record.inc.php

Class completion

```
class CompleteClass extends
```

Completion list:

- CompleteClass hierarchy.php
- Concatenation hierarchy.php
- UnitTest2 phpunit_test2.php
- _dummyConverter Converter.inc
- ... nup Template Class ...

Intention actions and quick fixes

```
class Concatenation {
    public function concatenate($string1, $string2) {
        $result = $string1.$string2;
        return $result;
    }
}
```

Intention actions:

- Resolve using PHPDOC annotation
- Typo: Accept 'result' as correct
- Typo: Rename to...

See Also

Procedures:

- [Creating Files from Templates](#)
- [Opening and Reopening Files in the Editor](#)
- [Advanced Editing Procedures](#)
- [Programming by Intention](#)

Reference:

- [Refactoring Dialogs](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

HTML, XHTML, XML and CSS

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm leverages support for `HTML/XHTML`, `XML` and `CSS`, making up the best tool kit for editing Web pages.

- Support for `CSS` is implemented as a plugin that comes bundled with PhpStorm. If you do not use PhpStorm as a Web development tool kit, you can turn it off.
- `HTML/XHTML`, `XML` and `CSS` [code completion](#): tags (including auto-insertion of closing tags), attributes, styles, and even file references in hyperlinks.
- [Find/highlight](#) usages of tags, IDs, files, images, styles, etc.
- `HTML/XHTML` [code formatting](#).
- Matching tag highlighting, which lets highlight tag pairs and quickly navigate between them.
- `HTML`, `XML` and `CSS` [code inspections](#):
 - Wrong or missing closing tag.
 - Missing, duplicate or incorrect attributes.
 - Wrong references to files in hyperlinks, and more.
- `HTML/XHTML`, `XML` and `CSS`-aware:
 - [Quick-fixes](#) for various code problems.
 - Syntax highlighting with a number of [customizable schemes](#).
 - Numerous [refactorings](#):
 - Safe delete for files.
 - Rename or move for files, anchors and other elements and more.
- [Quick view](#) of code documentation.
- Smart [Structure](#) tool window, aware of the tags and their hierarchy.
- Miscellaneous helpful features, available for both `HTML/XHTML`, `XML` and `CSS` code:
 - [Navigate to declaration](#) for tags, images and references.
 - [Show applied styles for tag](#) opens a tree view of all styles that are applied to it via the document stylesheet.
 - [Open in browser](#) quickly opens the current file in the specified Web browser.
 - [Auto-comment](#) quickly [toggles comments](#) for arbitrary blocks of code.
 - [Navigate to declaration](#) performs advanced navigation to the underlying code of a selector or ID.
 - [Show Content](#) is a Goto declaration complimentary that only shows the code in a pop-up window, without letting you modify it.
 - [Breadcrumb navigation tabs](#) is a toolbar that dynamically shows the current file hierarchy and provides one-click navigation, and can be disabled if necessary.
- Support for [Zen Coding templates](#).

Note

PhpStorm parses Web contents files according to the following specifications:

- `HTML`: specification `HTML 4.01` from W3C.
- `CSS`: specification `CSS 2.1`. The most common selectors are supported: universal selector `*`, type selectors `.a`, descendant selectors `.a.b`, child selectors `.a .b`, ID selectors `#b`, pseudo-classes and class selectors `DIV.warning`.
- PhpStorm uses `Xerces 2.6`, an XML parser developed by Apache Software Foundation Group.

See Also

Procedures:

- [Markup Languages and Style Sheets](#)
- [Reformatting Source Code](#)
- [Updating, Installing and Uninstalling Plugins from a Repository](#)

Reference:

- [Editor](#)
- [Plugins](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

JavaScript and AJAX

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm enables creating rich Internet applications and Web applications by providing elaborate support of [JavaScript](#) and tight integration with [AJAX](#) and other adjacent frameworks and technologies.

See Also

Procedures:

- [Advanced Editing Procedures](#)
- [Creating Documentation Comments](#)

- [Viewing Inline Documentation](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Using Language Injections

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm makes it possible to work with islands of different languages embedded in the source code. Any string literal can be treated as a source code in one of the [supported languages](#), rather than plain text.

You can **inject** other languages into **string literals**. This can be done within the source code written in most (but not all) of the [supported languages](#) (PHP, JavaScript, XML, and CSS). The typical examples are HTML fragments injected into JavaScript code, SQL statements in PHP or XML, and so on.

When the editor recognizes a string as a language injection:

- The syntax and error highlighting and coding assistance are extended to this string.
- You can open and modify it in a separate tab in the [editor](#), as if you were working with the source code in the corresponding language.

To open an injection in the editor, use the `Edit <Language> Fragment` [intention action](#).

To explain the PhpStorm editor that certain text should be always treated as an embedded source code, you can use:

- The `Inject Language` [intention action](#), which is applied to a particular string. Note that if you use this method, it's possible that the string literal will stay marked as a language injection only within a limited period of time. That is, PhpStorm, at a certain moment, may "forget" that the corresponding literal is a language injection. The period of the injection "persistence" will depend on the language, context and the modifications that you make in other parts of your source code.
- The [Language Injection](#) page of the [Settings](#) dialog box. Using this page, you can tell PhpStorm that a certain method parameter, text in an XML tag, or XML attribute should always be treated as embedded source code in another language by creating injection configurations.

Warning

PhpStorm comes with a set of predefined injection configurations which is quite sufficient to ensure high productivity and comfortable environment. Therefore it is strongly recommended that you use the predefined injection configurations and avoid creating new ones.

PhpStorm distinguishes the **project** and **global** states of injection configurations.

- Character strings configured as **project** injections are treated as source code only within the current project.
- Character strings configured as **global** injections are treated as source code at the PhpStorm level, that is, within any PhpStorm project.

To toggle between the **project** and **global** states, use the `Move to Project/Make Global` toolbar button  on the [Language Injection](#) page of the [Settings](#) dialog box.

PhpStorm supports full coding assistance for:

- CSS and JavaScript in an HTML or XML file.
- CSS, JavaScript, and SQL outside PHP code blocks and inside PHP string literals.

See Also

Concepts:

- [Code Inspection](#)
- [Intention Actions](#)

Reference:

- [Language Injections](#)
- [Inspections](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

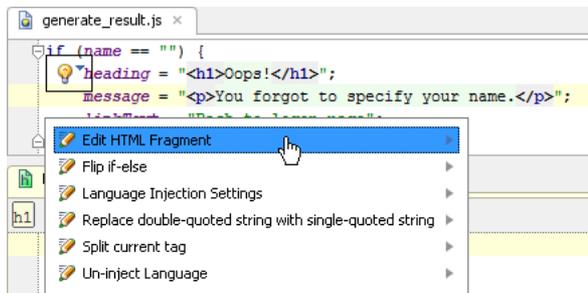
Opening Language Injections in the Editor

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac 

[Language injections](#) are opened in the editor by means of [a dedicated intention action](#) `Open <Injected Language> Fragment`.

To open a language injection in the editor:

1. In the editor, place the cursor within the string literal that represents the language injection of interest.
2. To show the available intention actions, do one of the following:
 - Click the yellow bulb .
 - Press `Alt+Enter` `Alt Enter`.
3. To select the `Edit <Injected Language> Fragment` option, do one of the following:
 - Click this option.
 - Use the `Up` `Up` and `Down` `Down` arrow keys to navigate to the option, and then press `Enter` `Enter` to select it.



Tip

You can open a number of language injections in separate editors, and copy code fragments from one editor to another. However, as soon as you start modifying the code in one of the editors, all the rest of the editors close automatically.

See Also

Concepts:

- [Using Language Injections](#)
- [Intention Actions](#)

Procedures:

- [Closing an Editor for a Language Injection](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Closing an Editor for a Language Injection

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

To close an editor for a [language injection](#), you can use the same options as for [closing files](#). Additionally, you can use the `EscapeEscape` key.

To close the editor for a language injection:

- While in the corresponding editor, press `EscapeEscape`.

Tip

You can have a number of language injections opened in separate editors at the same time. However, as soon as you start modifying the code for one of the injections, the editors for the rest of the injections all close automatically.

See Also

Concepts:

- [Using Language Injections](#)

Procedures:

- [Opening Language Injections in the Editor](#)
- [Closing Files in the Editor](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Live Templates

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Live templates contain predefined code fragments. You can use them to insert frequently-used or custom code constructs into your source code file quickly, efficiently, and accurately.

Live templates are stored in the following location:

- Windows: `<your home directory>\.<product name><version number>\config\templates`
- Linux: `~\.<product name><version number>\config\templates`
- MacOS: `~/Library/Preferences/<product name><version number>/templates`

In this section:

- [Simple, Parameterized and Surround Live Templates](#)
- [Live Template Abbreviation](#)

- [Live Template Variables](#)
- [Groups of Live Templates](#)

See Also

Procedures:

- [Creating and Editing Live Templates](#)
- [Creating and Editing Template Variables](#)
- [Surrounding Blocks of Code with Language Constructs](#)
- [Creating Code Constructs by Live Templates](#)
- [Zen Coding Support](#)

Reference:

- [Live Templates](#)
- [Edit Template Variables Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Simple, Parameterized and Surround Live Templates

Previous | [Next](#) | [See Also](#) | [Comments](#)

- [Simple live templates](#)
- [Parameterized live templates](#)
- [Surround live templates](#)

`Simple template` contains some fixed code that expands into plain text. When invoked and expanded in the editor, the code specified in the template is automatically inserted into your source code replacing the abbreviation.

`Parameterized template` contains plain text and [variables](#) that enable user input.

Tip

If you need a dollar sign (\$) in the template text, escape it by duplicating this character (\$\$).

After a template is expanded, variables appear in the editor as `input fields` whose values can be either filled in by the user or calculated by PhpStorm automatically.

`Surround templates` work only with the blocks of selected text. Such templates place code before and after the selected block.

See Also

Procedures:

- [Creating and Editing Live Templates](#)
- [Creating and Editing Template Variables](#)
- [Surrounding Blocks of Code with Language Constructs](#)

Reference:

- [Live Templates](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Live Template Abbreviation

Previous | [Next](#) | [See Also](#) | [Comments](#)

Each live template is identified by a `template abbreviation`.

The template abbreviations work like shortcuts and are expanded into fragments of source code, depending on the surrounding `context`. So doing, PhpStorm can format the generated code fragments according to the code style settings.

Note

An abbreviation may contain alphanumeric characters, dots, and hyphens, and must be unique within its [group](#). However, you can use the same abbreviation for different templates provided that they are in different groups.

See Also

Procedures:

- [Creating and Editing Live Templates](#)
- [Creating and Editing Template Variables](#)
- [Surrounding Blocks of Code with Language Constructs](#)
- [Creating Code Constructs by Live Templates](#)
- [Zen Coding Support](#)

Reference:

- [Live Templates](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Live Template Variables

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Template variables in live templates enable user input. After a template is expanded, variables appear in the editor as input fields.

Variables within templates are declared in the following format:

```
$<variable_name>$
```

Variables are defined by expressions, and can accept some default values.

This expression may contain constructs of the following basic types:

- Predefined functions with possible arguments.
- String constants in double quotes.
- The name of another variable defined in a live template.

Template variables are editable in the [Edit Template Variables Dialog](#), which contains a complete list of available functions.

PhpStorm supports two predefined live template variables: `END` and `$SELECTION$`.

- `END` indicates the position of the cursor after the template is expanded. For example, the template `return END` will be expanded into

```
return ;
```

with the cursor positioned right before the semicolon.

- `$SELECTION$` is used in surround templates and stands for the code fragment to be wrapped. After the template is expanded, the selected text is wrapped as specified in the template.

For example, if you select `EXAMPLE` in your code and invoke the "`$SELECTION$`" template via the assigned abbreviation or by pressing `Ctrl+Alt+T`/`Command Alt T` and selecting the desired template from the list, PhpStorm will wrap the selection in double quotes as follows:

```
"EXAMPLE".
```

Note

You cannot edit the predefined variables `END` and `$SELECTION$`.

See Also

Procedures:

- [Creating and Editing Template Variables](#)

Reference:

- [Edit Template Variables Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Groups of Live Templates

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

All templates are stored in groups for your convenience. Usually, templates are arranged into groups in accordance with the context they are sensitive to.

Each group is represented by an `.xml` file with the name `<group>.xml`; for example, `css.xml`. These files are stored in the `config/templates` directory that resides in the PhpStorm's [user settings directory](#).

PhpStorm comes with a set of pre-defined groups of live templates. The `user` group, represented by `user.xml` file, is intended for storing custom live templates.

Tip

It is strongly recommended to avoid storing custom templates in the pre-defined PhpStorm groups. For this purpose, use the `user` group, or a new group.

See Also

Procedures:

- [Creating and Editing Live Templates](#)
- [Zen Coding Support](#)

Reference:

- [Live Templates](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wiki/>
- <http://youtrack.jetbrains.com/issues/WI>

File Templates

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

File templates are specifications of the contents to be generated when creating a new file. They let you create the source files that already contain some initial code.

You can view, edit and create the templates on the [File Templates page](#) of the [Settings dialog](#) (File | Settings | File Templates for Windows and Linux or PhpStorm | Preferences | File Templates for Mac OS).

- [Overview](#)
- [Predefined, internal and custom file templates](#)
- [When are the file templates used?](#)

Overview

The file templates are written in the [Velocity Template Language](#) (VTL). So they may include:

- Fixed text (markup, code, comments, etc.). In a file based on a template, the fixed text is used literally, as-is.
- [File template variables](#). When creating a file, the variables are replaced with their values.
- [#parse directives](#) to include other templates.
- Other VTL constructs.

Here is a typical template example. (This template is used for creating a JavaScript file.)

```
/**
 * Created by ${PRODUCT_NAME}.
 * User: ${USER}
 * Date: ${DATE}
 * Time: ${TIME}
 * To change this template use File | Settings | File Templates.
 */
```

In this template, `${PRODUCT_NAME}`, `${USER}`, `${DATE}` and `${TIME}` are template variables.

Applying this template leads to generating a file whose contents look similar to this:

```
/**
 * Created by PhpStorm.
 * User: John.Smith
 * Date: 6/7/11
 * Time: 4:17 PM
 * To change this template use File | Settings | File Templates.
 */
```

Predefined, internal and custom file templates

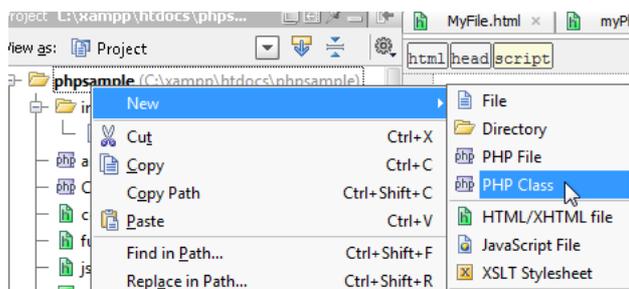
PhpStorm comes with a set of predefined file templates. You can use these templates as-is or modify them as necessary. You can as well create your own file templates (custom file templates).

Internal file templates are a subset of the predefined templates. These templates differ from all the other templates in that they cannot be deleted.

On the [File Templates page](#) of the [Settings dialog](#), the names of internal templates are shown in bold. The names of the custom templates and the predefined templates that you have modified are shown in blue.

When are the file templates used?

Whenever you create a new file, you can choose to create an empty file (e.g. File | New | File) or use a file template. In the latter case, the initial contents of the new file will be generated according to the template you have selected. (Basically, all the options in the New menu except File and Directory correspond to using a template.)



See Also

Concepts:

- [File Template Variables](#)

- [#Parse Directive](#)

Procedures:

- [Creating Files from Templates](#)
- [Creating and Editing File Templates](#)

Reference:

- [File Templates](#)

External Links:

- [Velocity Template Language User Guide](#) 

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

#Parse Directive

Previous | [Next](#) | [See Also](#) | [Comments](#)

Using the `#parse` directive, you can include other templates in [file templates](#). This is useful for inserting reusable contents (e.g. standard headers, copyright statements, etc.) into multiple file templates.

The syntax for the `#parse` directive is:

```
#parse("<template_name.extension>")
```

For example: `#parse("ActionScript File Header.as")`.

The templates that can be referenced like this in other templates, are shown on the [Includes](#) tab of the [File Templates page](#).

See Also

Concepts:

- [File Template Variables](#)

Procedures:

- [Creating Files from Templates](#)
- [Creating and Editing File Templates](#)

Reference:

- [File Templates](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

File Template Variables

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

A [file template](#) can contain `template variables`. When a template is applied the variables are replaced with their values.

A file template variable is a string that starts with a dollar sign which is followed by the variable name. The variable name may be enclosed in braces. For example: `$MyVariable` or `${MyVariable}`

PhpStorm comes with a set of `predefined template variables`.

The available predefined file template variables are:

- `${USER}` - login name of the current user.
- `${NAME}` - the name of the file that will be created.
- `${DATE}` - the current system date.
- `${TIME}` - the current system time.
- `${YEAR}` - the current year.
- `${MONTH}` - the current month.
- `${DAY}` - the current day of the month.
- `${HOUR}` - the current hour.
- `${MINUTE}` - the current minute.

In addition to the predefined template variables, it is possible to specify `custom variables`. If necessary, you can define the values of custom variables right in a template using the `#set` VTL directive (e.g. `#set($MyName = "John")`).

If when applying a template, the values of certain template variable are not known, PhpStorm will ask you to specify those values.

See Also

Concepts:

- [#Parse Directive](#)

Procedures:

- [Creating Files from Templates](#)
- [Creating and Editing File Templates](#)

Reference:

- [File Templates](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Running and Debugging

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

In PhpStorm, you can run and debug various types of applications without leaving the IDE.

Each script, console, server, or test that you wish to run or debug from within PhpStorm, needs a special profile that specifies script name, working directory, and other important data required for running or debugging in different modes. PhpStorm comes with a number of such pre-defined profiles, or [run/debug configurations](#).

Using the PhpStorm's debugger, you can find out the origin of the run-time errors and exceptions. The debugger enables you to execute your application step by step, examine program information related to variables, watches, or threads, and change your program without leaving the IDE.

Prior to launching the debugger session, you have to set [breakpoints](#) that cause the debugger to suspend application (or take some other actions), when such breakpoint is reached.

Generated output and console information during the running or debugging sessions are displayed in the [Run tool window](#) and [Debug tool window](#) respectively.

In this part you can find basic information about:

- [Run/Debug Configuration](#)
- [Temporary and Permanent Run/Debug Configurations](#)
- [Breakpoints](#)

Refer to the sections [Running](#), [Debugging](#) and [Testing](#) for the detailed descriptions of procedures.

See Also

Procedures:

- [Running](#)
- [Debugging](#)
- [Testing](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Run/Debug Configuration

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm enables using numerous run/debug configurations. Each `run/debug configuration` represents a named set of run/debug startup properties. When you perform run, debug, or test operations with PhpStorm, you always start a process based on one of the existing configurations using its parameters.

PhpStorm comes with a number of pre-defined default run/debug configuration types for the various running, debugging and testing issues.

Whenever a new [temporary or permanent](#) run/debug configuration is created, it is based on these default settings applicable to any new configuration of the respective type.

See Also

Concepts:

- [Temporary and Permanent Run/Debug Configurations](#)

Procedures:

- [Creating and Editing Run/Debug Configurations](#)
- [Creating and Saving Temporary Run/Debug Configurations](#)
- [Code Coverage](#)

Reference:

- [Run/Debug Configurations](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Temporary and Permanent Run/Debug Configurations

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Run/debug configurations are categorized into `temporary` and `permanent`.

Temporary Configuration

A temporary run/debug configuration is automatically created every time you choose **Run <item_name>** or **Debug <item_name>** for an item without a permanent configuration. Temporary configurations can be saved as permanent.

Temporary configurations are marked with semi-transparent icons and are managed same way as the permanent configurations.

By default, 5 temporary configurations are allowed per project. You can change this limit via the [Edit Configurations](#) dialog.

Permanent Configuration

A configuration of this type [is explicitly created](#) for a particular class or method. If there is no permanent configuration for an item, PhpStorm automatically creates a temporary configuration for it, when you choose **Run <item_name>** or **Debug <item_name>** on the context menu of this class or method.

See Also

Procedures:

- [Creating and Saving Temporary Run/Debug Configurations](#)

Reference:

- [Run/Debug Configurations](#)

Web Resources:

- <http://www.jetbrains.com/idea/faq/faq-run-debug.html>
- <http://youtrack.jetbrains.com/issue/WI-10000>

Breakpoints

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Breakpoints are source code markers used to trigger actions during a debugging session.

In this part:

- [Types of Breakpoints](#)
- [Breakpoints Icons and Statuses](#)

Typically, the purpose behind setting a breakpoint is to suspend program execution to allow you to examine program data. However, PhpStorm can use breakpoints as triggers for a variety of different actions. Breakpoints can be set at any time during the debugging process. Your breakpoints don't affect your source files directly, but the breakpoints and their settings are saved with your PhpStorm project so you can reuse them across debugging sessions.

See Also

Procedures:

- [Using Breakpoints](#)

Reference:

- [Breakpoints](#)

Web Resources:

- <http://www.jetbrains.com/idea/faq/faq-run-debug.html>
- <http://youtrack.jetbrains.com/issue/WI-10000>

Types of Breakpoints

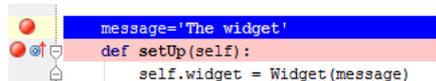
[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

You can create **line breakpoints** that are assigned to the lines of source code and point at particular sections for debugging.

Depending on the context, PhpStorm distinguishes between the following types of breakpoints:

- **PHP breakpoints** can be set in the PHP context inside *.php, *.html, and files of other types.
- **JavaScript breakpoints** can be set in the JavaScript context inside *.js or *.html files.

Breakpoints are triggered when the program reaches the specified line of source code, before it is executed. The line of code that contains a set breakpoint, is marked with a red stripe; once such line of code is reached, the marking stripe changes to blue:



Once set, a breakpoint remains in project until removed. Breakpoints can only be set on executable lines of code. Comments, declarations of methods, and empty lines are not valid locations for breakpoints.

Tip

If a file with breakpoints has been modified externally, for example, updated from a version control repository, or changed in an external editor, so that line numbers are changed, then the breakpoints will be moved accordingly.

It is important to note that PhpStorm should be running at the moment of such modification; otherwise, such changes will pass unnoticed.

In this section:

See Also

Procedures:

- [Using Breakpoints](#)

Reference:

- [Breakpoints](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Breakpoints Icons and Statuses

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

When a breakpoint is set, the editor displays a breakpoint icon in the gutter area to the left of the affected source code. A breakpoint icon denotes status of a breakpoint, and provides useful information about its type, location, and action.

The icons serve as convenient shortcuts for managing breakpoints. Clicking an icon removes the breakpoint. Successive use of **Alt+Click** on an icon toggles its state between enabled and disabled. The settings of a breakpoint are shown at the tooltip when a mouse pointer hovers over a breakpoint icon in the gutter area of the editor.

The table below summarizes the possible breakpoint states:

Status	Icon	Description
Enabled		Shows at design-time or during the debugging session.
Disabled		Indicates that nothing happens when the breakpoint is hit.
Conditionally disabled		This state is assigned to breakpoints when they depend on another breakpoint to be activated.

See Also

Procedures:

- [Using Breakpoints](#)

Reference:

- [Breakpoints](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Encoding

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Different types of files use different ways to define encoding. PhpStorm recognizes encoding of files based on their contents.

Encoding has influence on the way PhpStorm reads or writes files. If a file has been modified but not yet saved, any changes in encoding affect file writing; if a file has not been modified, then reading is affected. PhpStorm suggests specific ways to change encoding of a file according to its type, using [File Encodings dialog](#), the [Status bar, or the editor](#).

Encoding	Can be changed in
File encoding is specified within the file, for example, in XML.	If a file contains explicit encoding declaration, you can change it in the Editor . In this case PhpStorm provides code completion.
File encoding is defined by BOM.	In this case, you can't change encoding with which PhpStorm reads the file, but it is still possible to change encoding for writing such file.
UTF characters are detected in the file contents.	PhpStorm provides an option that automatically changes file encoding to UTF , if the file contents can be reasonably interpreted as UTF. This option only works for reading; a file can be saved with any encoding.
Encoding cannot be found out from the file content.	In this case, the default encoding is the one defined by the IDE encoding in the File Encodings page of the Settings dialog. You can change it for multiple files and directories , or for a single file .

Note

Encoding applies to directories and individual files. So doing, the encoding information contained in a file overrides the default encoding; encoding of a file or subdirectory overrides that on the higher levels.

See Also

Procedures:

- [Configuring Individual File Encoding](#)

Reference:

- [File Encodings](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

External Tools

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Support for external tools enables integration with the third-party applications without creating plugins.

Configuration options for the external tools enable passing contextual information (like the current file name or project source path) to the application via command-line arguments.

The list of external tools is preserved as a part of the IDE setting, and can be shared among the development team. For managing the list of external tools, PhpStorm provides the [External Tools](#) dialog.

Once added, the external tools appear as the new menu commands.

See Also

Procedures:

- [Configuring Third-Party Tools](#)

Reference:

- [External Tools](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Plugins

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

On this page:

- [Types of Plugins](#)
- [Plugin Repositories](#)

Types of plugins

PhpStorm distinguishes between the following three types of plugins:

- Plugins bundled with PhpStorm. These plugins are installed and enabled by default.
- [Plugins available from the PhpStorm Repository](#). These plugins need installation and enabling.
- [Custom plugins available from enterprise repositories](#). These plugins also need installation and enabling.

The set of installed and enabled plugins determines the set of supported features, module types, VCS integrations, application servers, and development technologies. Once added, plugins appear in the product UI after PhpStorm restart.

Plugin repositories

JetBrains Plugins Repository resides at <http://plugins.jetbrains.net/> and stores numerous plugins, created by the community members. You can use the [Plugin Manager](#), represented by the [Plugins](#) dialog, to download, install, update, and remove plugins.

For custom plugins you want to preserve for internal use only, PhpStorm helps [maintain your own enterprise repositories](#).

Note

All plugins available from enterprise repositories are registered in the `updatePlugins.xml` file. Each plugin entry in this file should contain plugin identifier and URL of the plugin repository. Additionally, plugin version can be specified here. Though, if it is not pointed out in `updatePlugins.xml`, PhpStorm will fish it out from the plugin `*.jar`.

The DTD of such file is:

```
<!DOCTYPE plugins [
  <!ELEMENT plugins (plugin)*>
  <!ELEMENT plugin (#PCDATA)>
  <!ATTLIST plugin
    id CDATA #REQUIRED url DATA #REQUIRED version CDATA #REQUIRED>]>
```

See Also

Procedures:

- [Managing Plugins](#)

Reference:

- [Plugins](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Version Control with PhpStorm

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm provides integration with several Version Control Systems (referred to as VCS in documentation). This includes both support of features specific for each VCS as well as unified interface and management for common VCS tasks.

In this part:

- [Supported Version Control Systems](#)
- [Directory-Based Versioning Model](#)
- [Unified Version Control Functionality](#)
- [Changelist](#)
- [Local, Committed and Incoming Changes](#)
- [Local History](#)
- [Patches](#)
- [Shelved Changes](#)

See Also

Procedures:

- [Version Control with PhpStorm](#)

Reference:

- [Changes Tool Window](#)
- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Supported Version Control Systems

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

The list of supported integrations of version control systems is determined by the set of currently enabled [plugins](#).

The basic principles of getting started with the supported version control systems (VCS) in PhpStorm are rather similar, though some commands and settings are specific and conform to the version control system conventions. In addition to the common information, you can find VCS-specific procedures in the following sections:

- [Using CVS Integration](#)
- [Using Git Integration](#)
- [Using Subversion Integration](#)
- [Using Mercurial Integration](#)
- [Using Perforce Integration](#)
- [Using TFS Integration](#)
- [Using Visual SourceSafe Integration](#)

See Also

Procedures:

- [Version Control with PhpStorm](#)

Reference:

- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Directory-Based Versioning Model

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm adopts a directory based model for version control association. A version control system is assigned to a project directory and/or additional directories that are part of or related to the project. Directories under version control are not required to be located under the project root. They can reside in any accessible location.

Note

The file in `.idea` directory that stores your local preferences (`workspace.xml`) file should not be placed under version control. All the other settings files should be shared.

See Also

Procedures:

- [Enabling Version Control](#)

Reference:

- [Version Control](#)
- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Unified Version Control Functionality

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

In addition to support for general and individual VCS commands, PhpStorm provides several unique features that simplify and speed up the work with any version control system.

- For the projects with VCS support enabled, the standard VCS actions (commit, update, revert, show differences and show history) are added to the main toolbar.
- Commit and update an entire project.
- Uniform interface for configuring common version control system settings.
- Changelists support for all integrated version control systems.
- `Next`, `Previous`, `Rollback`, `Show Difference` actions are available from the dedicated gutter bar in changed locations.
- View revision history for file/directory.
- Automatic checkout of all affected files when refactoring.
- Advanced Version Control tool window, with multiple dedicated tabs: History, Status, Update Info, etc.

Note

Mind the difference in terminology in the different version control systems. For example, to denote the check-in functionality, `Git` uses the term `push`, `Subversion` uses `submit`, etc.

See Also

Procedures:

- [Common Version Control Procedures](#)

Reference:

- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Changelist

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

A changelist is a set of changes in files that represents a logical change in source code. The changes specified in a changelist are not stored in the repository until committed (pushed).

Any changes made to the source files, are automatically included in the `active changelist`. Initially, the Default changelist is active, but you can make any other changelist active. The active changelist is displayed on top of the Changes tool window, with the name being highlighted in bold font.

In addition to the Default changelist, you can create new changelists, delete existing ones (except for the Default changelist), and move files between changelists.

All modified, deleted, unversioned and other files are managed in the [Changes tool window](#). From this window you can:

- Commit (push) changelists.
- Create new changelists (if you want to keep an eye on certain files and changes).
- Remove existing changelists and set the default changelists.
- Rollback modified files in changelists.
- Add the unversioned files and directories to the version control.
- Move files between changelists.
- Show differences on selected files.
- Refresh the list of VCS changes.
- Jump to the source code from within a changelist.
- Shelve (stash) and unshelve (unstash) changes.

See Also

Procedures:

- [Managing Changelists](#)

Reference:

- [Changes Tool Window](#)
- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Local, Committed and Incoming Changes

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm distinguishes among three categories of changes: **Local**, **Repository**, and **Incoming**.

Local Changes

The changes that you have introduced to your local working copy but have not yet checked in to the repository.

Repository Changes

The changes that you and other users have checked in to the repository.

Incoming Changes

The changes that have been checked in to the repository but that you have not yet checked out locally.

Note

Repository and incoming changes are not supported by Git and Mercurial integrations.

All these types of changes are displayed in the respective tabs of the [Changes](#) tool window. Information about changes is stored in the history cache. Configure the size of this cache and frequency of refreshing information in the [General VCS Settings](#).

See Also

Concepts:

- [Shelved Changes](#)

Procedures:

- [Version Control with PhpStorm](#)

Reference:

- [Changes Tool Window](#)
- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
 - <http://youtrack.jetbrains.com/issues/WI>
-

Local History

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Your source code constantly changes as you edit, test, or compile. Any version control system tracks the differences between the committed versions, but the local changes between commits pass unnoticed. **Local History** is your personal version control system that tracks changes to your source code on your computer and enables you to compare versions and roll changes back, if necessary. Local History is always at your disposal, no steps are required to enable it.

Local History is independent of external version control systems and works with the directories of your project even when they are not under any VCS control. It applies to any structural artifacts: a project, a directory or package, a file, a class, class members, tags, or selected fragment of text.

Unlike usual version control systems, Local History is intended for your personal use, it does not support shared access.

With Local History, PhpStorm automatically tracks changes you make to the source code, results of refactoring, and state of the source code based on a set of predefined events (testing, deployment, commit or update).

Local History revisions are marked with labels, which are similar to versions in traditional version control systems. Labels based on predefined events are added to the local revisions automatically; besides that, you can put your own labels to the project artifacts to mark your changes. Reverting or viewing differences are performed against these labels.

Note

Local History has certain limitations:

- Tracking local changes is only possible for textual files. Binary files do not have Local History.
- For files larger than 1 Mbyte, Local History tracks only the very fact of changes, but does not preserve the respective contents.

See Also

Procedures:

- [Using Local History](#)

Reference:

- [Version Control Reference](#)
- [Show History for File / Selection Dialog](#)
- [Show History for Folder Dialog](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
 - <http://youtrack.jetbrains.com/issues/WI>
-

Patches

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm helps you create and apply patches to the source code. A patch is a file in the standard text format that has the *.patch extension and contains a list of differences between the two sets of source files.

Patches contain only changes to textual files. Changes to binary files cannot be patched.

This concept is tightly related with the concept of [Shelved Changes](#).

See Also

Procedures:

- [Creating Patches](#)
- [Applying Patches](#)

Reference:

- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Shelved Changes

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

You can run into a situation when you are short of time to bring your source code to a certain required condition or you need to work on an urgent high priority task. In this case you might want to put some changes aside and continue working on a stable version.

With PhpStorm, you can use shelves for storing postponed changes temporarily. In due time, the desired changes can be taken back from the shelf (unshelved).

PhpStorm enables shelving both separate files and entire changelists. Accordingly, you can unshelve entire shelves or specific files.

See Also

Concepts:

- [Patches](#)

Procedures:

- [Shelving Changes](#)
- [Unshelving Changes](#)
- [Stashing and Unstashing Changes](#)

Reference:

- [Version Control Reference](#)
- [Shelve Changes Dialog](#)
- [Unshelve Changes Dialog](#)
- [Changes Tool Window](#)
- [Stash Dialog](#)
- [Unstash Changes Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

PhpStorm Usage Guidelines

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

This part provides descriptions of the actions required to fulfil certain common tasks with PhpStorm.

A

B

C

D

E

F

G

H

I

- [J](#)
- [L](#)
- [M](#)
- [N](#)
- [O](#)
- [P](#)
- [Q](#)
- [R](#)
- [S](#)
- [T](#)
- [U](#)
- [V](#)
- [W](#)
- [Z](#)

See Also

Reference:

- [Reference](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Configuring IDE Settings

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

This part suggests descriptions of some basic procedures related to customizing your working environment:

- [Accessing the IDE Settings](#)
- [Configuring Colors and Fonts](#)
- [Configuring Keyboard Shortcuts](#)
- [Configuring Menus and Toolbars](#)
- [Configuring Quick Lists](#)
- [Configuring Third-Party Tools](#)
- [Creating and Editing File Templates](#)
- [Creating and Registering File Types](#)
- [Creating and Editing Live Templates](#)
- [Creating and Editing Template Variables](#)
- [Exporting and Importing Settings](#)
- [Switching Between Schemes](#)

Find descriptions of the other specific configuration procedures in the relevant procedural parts.

See Also

Concepts:

- [Project and IDE Settings](#)

Procedures:

- [Improving Visibility of the Source Code](#)
- [Exporting and Importing Settings](#)

Reference:

- [IDE Settings](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Accessing the IDE Settings

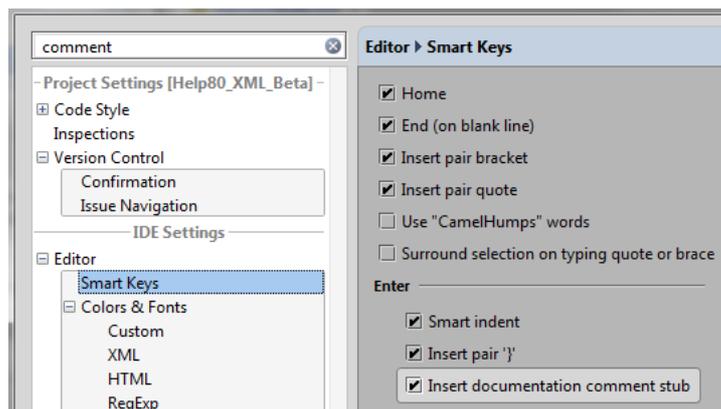
Previous | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

To gain access to the IDE Settings

- Do one of the following:
 - Press `Ctrl+Alt+SMeta Comma`.
 - On the main toolbar, click the **Settings** button 
 - On the main menu, choose:
 - File | Settings** for Windows and Linux.
 - PhpStorm | Preferences** for Mac OS.
- In the [Settings](#) dialog box that opens, select the desired page under the **IDE Settings** node.

To quickly find an option or setting

- [Open the IDE settings](#).
- In the **Search** field, start typing the name of the setting or a part of its name. As you type, the first matching node in the list is highlighted and the corresponding page is displayed.



See Also

Concepts:

- [Project and IDE Settings](#)

Reference:

- [Settings Dialog](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Configuring Colors and Fonts

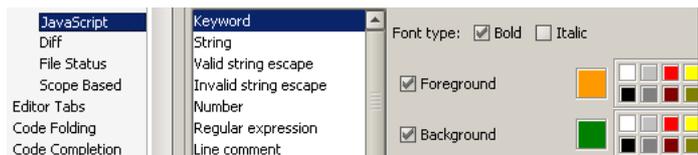
[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

With PhpStorm, you can maintain your preferable colors and fonts layout for syntax and error highlighting in the editor, search results, Debugger and consoles via font and color schemes.

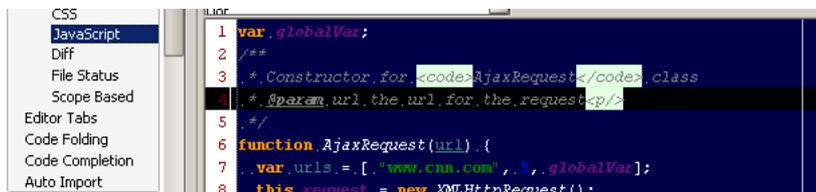
PhpStorm comes with a number of pre-defined color schemes. You can select one of them, or create your own one, and configure its settings to your taste. Note that pre-defined schemes are not editable. You have to create a copy of the scheme, and then change it as required.

To configure color and font scheme

- [Open the IDE Settings](#), and under the **Editor** node, click [Colors&Fonts](#).
- Select the desired scheme from the **Scheme name** drop-down list.
- If you need to change certain settings of the selected scheme, create its copy. To do that, click **Save as** button, and type the new scheme name in the dialog box.
- Open pages to configure specific color preferences and font types for the different supported languages and PhpStorm components.



- Observe results in the preview pane:



You can also view how the new scheme looks in the editor. To do that, just click **Apply** without closing the **Settings** dialog box.

See Also

Reference:

- [Editor. Colors and Fonts](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wiki/>
- <http://youtrack.jetbrains.com/issues/WI>

Configuring Keyboard Shortcuts

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

PhpStorm is a keyboard-centric IDE. Most of the actions (navigation, refactoring, debugging, etc.) can be carried out without using a mouse, which lets dramatically increase coding speed. If you had used another IDE for a while and have memorized your favorite keyboard shortcuts, you can use them all in PhpStorm.

PhpStorm completely suits your *shortcut habits* by supporting customizable keymaps. A keymap is a set of keyboard and mouse shortcuts that invoke different actions - menu commands, editor operations, etc. PhpStorm comes with a set of preconfigured keymaps.

Tip

Preconfigured keymaps are not editable. If you need to change some shortcuts, create a copy of the desired keymap with a new name, and then modify as required.

All user-defined keymaps are stored in separate configuration files under the `config/keymaps` subdirectory in the PhpStorm profile directory:

- Windows and *NIX systems: `<User home>/.PhpStorm<xx>/config/keymaps`
- Macintosh systems: `~/Library/Preferences/.PhpStorm<xx>/keymaps/`

Each keymap file contains only differences between the current and the parent keymaps.

To configure keyboard shortcuts and mouse shortcuts

1. [Open Settings dialog](#) and click [Keymap](#).
2. Select one of the preconfigured **Keymaps**, which you want to use a base for the new one, and click **Copy**. Accept the default name, or change it as required.
3. In the **All Actions** list, select the desired action.
4. **Configure keyboard shortcuts**. To do that, follow these steps:
 1. Click **Add Keyboard Shortcut**. [Enter Keyboard Shortcut](#) dialog box appears.
 2. Press the keys to be used as shortcuts. The keystrokes are immediately reflected in the **First Stroke** field. Optionally, check the **Enable** option and press keys for the **Second Stroke**. As you press the keys, **Preview** field displays the suggested combination of keystrokes, and the **Conflicts** field displays warnings, if some of the keystrokes are already assigned to the other actions.
 3. Click **OK** with the mouse pointer to create a shortcut and bind it with an action.

Warning

It is important to use the mouse pointer, because any keystroke is interpreted as a shortcut.

5. **Configure mouse shortcuts**. To do that, follow these steps:
 1. Click **Add Mouse Shortcut**, if you need to bind an action to a mouse click. [Enter Mouse Shortcut](#) dialog box appears.
 2. In the **Click Count** section, click a radio button to choose a **Single Click** or **Double Click**.
 3. Hover your mouse pointer over the section **Click Pad** and click the desired mouse button. Use **AltAlt**, **CtrlCtrl** and **ShiftShift** modifiers for diversity. As you click, the **Shortcut Preview** field displays the suggested shortcut, and the **Conflicts** field displays warnings, if some of the shortcuts are already assigned to the other actions.
 4. Click **OK** or **Press EnterEnter** to create a shortcut and bind it with an action.

Warning

If a conflict is reported, a warning message appears. You can choose one of the following options:

- **Remove** to remove all other bindings and preserve the new one.
- **Leave** to preserve all bindings including the new one.
- **Cancel** to return to the keymap definition.

Although you can ignore conflict and bind a shortcut with several actions, it is strictly recommended to avoid binding two actions with the same shortcut, because the order of performing such actions is not defined.

See Also

Reference:

- [Keyboard Shortcuts and Mouse Reference](#)
- [Keymap](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Configuring Menus and Toolbars

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

You can customize menu and toolbar command lists to regroup features or make your favorites easier to access.

To customize menus and toolbars

1. [Open the IDE Settings](#) and click [Menus and Toolbars](#). Alternatively, right-click the main toolbar, and choose **Customize Menus and Toolbars** on the context menu.
2. In the list of available menus and bars, expand the node you want to customize and select the desired item.
3. Customize the list of items in the selected menu or bar using the buttons on the right from the list:
 - o To add a new command, select the desired location in the list and click the **Add After** button. In the [Choose Action To Add](#) dialog box that opens, select the desired action.

Optionally associate the action with an icon using the **Icon Path** text box. In this text box, specify the location of the file with the icon you want to assign to the selected action. If necessary, use the **Browse** button  to open the **Select Path** dialog box.

Tip

The image file should have .png extension.

- o To change the icon associated with a command, select the desired command in the list and click the **Edit Action Icon** button. In the **Choose Actions Icon Path** dialog box that opens, specify the location of the desired image. If necessary, use the **Browse** button  to open the **Select Path** dialog box and select an image there.
 - o To delete an item from the list, select it and click the **Remove** button.
 - o To have logical groups of commands separated from each other by a separator, select the desired location in the list and click the **Add Separator** button.
 - o To change the order in which commands appear in the selected menu or on the selected bar, use the **Move Up** and **Move Down** buttons.
4. To abandon the changes and return to the default settings, click the **Restore Default** button.

See Also

Procedures:

- [Menus and Toolbars](#)

Reference:

- [Menus and Toolbars](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Configuring Quick Lists

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

A **Quick List** is a pop-up menu of PhpStorm commands, configured by the user and associated with a keyboard or mouse shortcut. You can create as many quick lists, as necessary. Each command, included in a quick list, is identified by a sequential number. Numbering starts from the numerals (0 to 9), and then proceeds with the letters in alphabetical order.

To invoke a command from a quick list

1. Invoke quick list by its keyboard shortcut.
2. Select the desired command, using its number, the mouse cursor, or navigation keys and the **Enter** key:



To configure a quick list

1. [Open the IDE Settings](#), and select **Quick List**.
2. Click add button  to create a new unnamed quick list.
3. In the **Display name** field, specify the name of the quick list, or accept default. Optionally, provide description.
4. Using the **->**, **<-** and **Separator** buttons, make up the list of actions to be included in the quick list. Using the **Move Up/Move Down** buttons, provide the desired order of actions.
5. Apply changes. The new quick list appears in the Quick Lists node of All Actions list.
6. Bind the new quick list with one or more shortcuts:

- In the [Keymaps](#) page of the Settings dialog, expand the Quick Lists node and select the new quick list.
 - Perform the [key binding procedure](#). Note that you can only modify a custom keymap.
7. Apply changes and close the dialog.

See Also

Procedures:

- [Configuring Keyboard Shortcuts](#)

Reference:

- [Quick Lists](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi> 
 - <http://youtrack.jetbrains.com/issues/WI> 
-

Configuring Third-Party Tools

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

This section describes how to add a new tool to PhpStorm, and define the menus to which the tool command will be added.

To add a new tool to the product, or change an existing one

1. Open the [External Tools](#) dialog.
2. Click **Add** to create a new entry, or select an existing tool, and click **Edit**. The [Edit Tool](#) dialog box appears.
3. Specify the tool name, the group where the tool should belong, and optional description.
4. In the **Menu** section, check the menus, where you want the tool to appear. You can opt to add menu items for the new tool to the main menu, context menus of the Project views and the editor, and the search results.
5. In the **Program** field, specify the name of the executable, which defines the program to run.
6. In the **Parameters** field, specify the list of arguments or parameters, as if you were entering them on the command line.
7. In the **Working directory** field, specify the directory where the tool should run.
8. Apply changes, and close the dialog. The new tools can now be accessed via **Tools | <Group Name>**:

Note

If you want to use a keyboard shortcut to invoke your external tool, [bind this tool with a shortcut](#).

See Also

Concepts:

- [Plugins](#)

Reference:

- [External Tools](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi> 
 - <http://youtrack.jetbrains.com/issues/WI> 
-

Creating and Editing File Templates

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm enables the following ways of creating [file templates](#):

- [From scratch](#).
- [On the base of existing templates](#).
- [From existing source files](#).
- Using [template includes](#).

To create a file template from scratch

1. On the main menu, choose **File | Settings** for Windows and Linux or **PhpStorm | Preferences |** for Mac OS. Then click **File Templates**.
2. In the [File Templates](#) dialog box that opens switch to the **Templates** tab.
3. Click the **Add** button  on the toolbar and specify the template name, the file extension, and the body of the template, which can contain:
 - Plain text.
 - `#parse` directives to work with [template includes](#).
 - Predefined variables to be expanded into corresponding values in the format `${<variable_name>}`.

The available predefined file template variables are:

- `${USER}` - login name of the current user.

- `{NAME}` - the name of the file that will be created.
 - `{DATE}` - the current system date.
 - `{TIME}` - the current system time.
 - `{YEAR}` - the current year.
 - `{MONTH}` - the current month.
 - `{DAY}` - the current day of the month.
 - `{HOUR}` - the current hour.
 - `{MINUTE}` - the current minute.
- Custom variables to define their names during the file creation.
4. To have a variable or directive rendered "as is" upon template expansion, [escape](#) the `{}` or `#` character in reposition.
 5. Apply the changes and close the dialog box.

To create a file template from an existing one

1. Open the [File Templates](#) dialog box and switch to the **Templates** tab.
2. Click the **Copy** button  on the toolbar and change the template name, extension, and source code as required.
3. Apply the changes and close the dialog box.

To create a file template from a file

1. Open the desired file in the editor.
2. On the main menu, choose **Tools | Save File as Template**.
3. In the [File Templates](#) dialog box that opens specify the new template name and edit the source code, if necessary.
4. Apply the changes and close the dialog box.

To create and reference an include template

1. In the [File Templates](#) dialog box, switch to the **Includes** tab.
2. Click the **Add** button  on the toolbar and specify include template name, extension, and the source code.
3. In the **Templates** tab, select the desired template and click the **Edit** button.
4. To include a template, insert the `#parse` directive in the source code.

See Also

Concepts:

- [File Templates](#)

Reference:

- [File Templates](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Escaping Characters in Velocity Variables and Directives

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

You can include a variable or a directive in your template but suppress expanding it by PhpStorm on template applying. Just prepend the `{}` or `#` symbol for variable or directive respectively with a backslash `'\'` character. The [Velocity](#)  engine will exclude the symbol from processing according to [Velocity escaping rules for file templates](#).

No backslash is required if a variable or directive itself contains any symbols that prevent the Velocity engine from treating it as a Velocity variable or directive. Such characters are, for example, dots, interrogation marks, quotes, double quotes, etc. see the [Velocity notation](#)  for details.

Examples of applying [Velocity escaping rules for file templates](#):

- To use a `{SINGLE_VARIABLE_IDENTIFIER_WITH_NO_DOTS}`, prepend it with a backslash: `\SINGLE_VARIABLE_IDENTIFIER_WITH_NO_DOTS`
- To use a `{SINGLE_DIRECTIVE_WITH_SINGLE_SCRIPT_ELEMENT}`, prepend it with a backslash: `\#SINGLE_DIRECTIVE_WITH_SINGLE_SCRIPT_ELEMENT`.

Note

For details on escaping directives with multiple script elements inside, see [Velocity guide](#) .

- To use a `{SINGLE.VARIABLE.IDENTIFIER.WITH.DOTS}` no backslash is required. The engine will not treat such expression as a variable to be processed because a Velocity variable cannot contain dots according to the Velocity variable notation.
- To use some version control keywords (such as `{Revision$}`, `{Date$}`, etc.) in your default class template, prepend them with a backslash: `\$Revision$`.

See Also

Concepts:

- [File Templates](#)

Procedures:

- [Creating and Editing File Templates](#)

Reference:

- [File Templates](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Creating and Registering File Types

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

You can [create custom file types](#) to enable parsing these files in the editor by defining highlighting schemes for keywords, comments, numbers, etc. To enable PhpStorm to decide how to treat a file, you need to [associate each file type with relevant extensions](#).

To create a new file type

1. [Open the Settings](#) dialog box and click [File Types](#).
2. On the [File Types](#) page that opens click the **Add** button .
3. In the [New File Type](#) dialog box that opens specify the name of the new type and optionally provide a description.
4. In the **Syntax Highlighting** section, specify the characters for line and block comments, hex prefixes, and number postfixes.
5. In the **Keywords** section, specify sets of keywords using the tabs from 1 to 4. To do so, select the desired tab, click the **Add** button, and enter the keyword name in the **Add New Keyword** dialog box that opens.

Tip

Each set of keywords has its own highlighting. You can change the highlighting color scheme for each set, using the [Colors and Fonts](#) dialog box. Click the **Custom** tab and edit the **Keyword1**, **Keyword2**, **Keyword3**, and **Keyword4** properties.

To associate a file type with extensions

1. [Open the File Types](#) dialog box.
2. From the **Recognized File Types** list, select the desired type.
3. In the **Registered Patterns** area, complete the list of patterns that define the file extensions to indicate that the corresponding files belong to the selected type. Do one of the following:
 - To register a new pattern, click the **Add** button and enter the desired extension pattern in the **Add Wildcard** dialog box that opens.
 - To update a pattern, select it in the list, click the **Edit** button, and make the necessary changes in the **Edit Wildcard** dialog box that opens.
 - To remove a pattern from the list, select it and click the **Remove** button.

See Also

Reference:

- [File Types](#)
- [New File Type](#)
- [Register New File Type Association Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Creating and Editing Live Templates

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm comes with a set of the predefined [Live Templates](#). You can use them as is, or [modify](#) them to suit your needs.

If you want to create a new live template, you can do it [from scratch](#), on the base of the copy of an existing template, or from a [fragment of source code](#).

To modify an existing template

1. In the [IDE settings](#), open the [Live Templates](#) page.
2. Expand the desired template group, and select the template you want to change.
3. In the [Template Text](#) area, change the [template abbreviation](#) as required.
4. In the [Template Text](#) field, edit the template body which may contain plain text and variables in the format `<variable name>$.`
 - If you need a dollar sign (\$) in the template text, escape it by duplicating this character (`$$`).
 - To [change variables](#) in a template, click the **Edit Variables** button.

PhpStorm supports two predefined live template variables: `END` and `$SELECTION$`.

- `END` indicates the position of the cursor after the template is expanded. For example, the template `return END` will be expanded into

```
return ;
```

with the cursor positioned **right before** the semicolon.

- `$SELECTION$` is used in `surround` templates and stands for the code fragment to be wrapped. After the template is expanded, the selected text is wrapped as specified in

the template.

For example, if you select `EXAMPLE` in your code and invoke the "\$SELECTION\$" template via the assigned abbreviation or by pressing `Ctrl+Alt+T`/`Command Alt T` and selecting the desired template from the list, PhpStorm will wrap the selection in double quotes as follows:

```
"EXAMPLE".
```

Note

You cannot edit the predefined variables `END` and `$SELECTION$`.

Note

The **Edit Variables** button is enabled only if the template body contains at least one user-defined variable, that is, a variable different from `END` or `$SELECTION$`.

5. In the **Options** section, specify how the template will be expanded and reformatted.
6. In the **Available in** section, specify the languages and places of code where the editor should be sensitive to the template abbreviation.
7. Click **OK** when ready.

To create a new template from scratch

1. In the [IDE settings](#), open the [Live Templates](#) page, and expand the template group where you want to create a new template.
2. Click the **Add** button . A new template item is added to the group and the focus moves to the [Template Text](#) area.
3. Specify the new template abbreviation, type the template body, define the variables and the template group, configure the options, as described in the [template modification](#) procedure.
4. Click **OK** when ready.

To create a live template from a text fragment

1. In the editor, select the text fragment to create a live template from.
2. On the main menu, choose **Tools | Save as Live Template**. The [Live Templates](#) page of the **Settings** dialog box opens, with the [Template Text](#) area in focus.
3. In the **Abbreviation** field, type abbreviation to identify your new live template.
4. Specify the new template abbreviation, type the template body, define the variables and the template group, configure the options, as described in the [template modification](#) procedure.
5. Click **OK** when ready.

See Also

Concepts:

- [Live Templates](#)

Procedures:

- [Creating Code Constructs by Live Templates](#)
- [Expanding Zen Coding Templates with User Defined Templates](#)
- [Zen Coding Support](#)

Reference:

- [Live Templates](#)
- [Edit Template Variables Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Creating and Editing Template Variables

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

After a [template](#) is expanded, its variables are presented in the editor as input fields. The values of these fields can be either filled in by the user or calculated by PhpStorm.

To have it done automatically, for each variable you need to specify the following:

- Expression to be calculated in association with the variable.
- Default value to be entered in the input field if the calculation fails.

The order in which PhpStorm will process input fields after the template expansion, is determined by the order of variables in the list.

To configure variables used in a template

1. [Open the template settings](#), and in the [Template Text](#) area click the **Edit Variables** button.

Note

The **Edit Variables** button is enabled only if the template body contains at least one user-defined variable, that is, a variable different from `END` or `$SELECTION$`.

The [Edit Template Variables](#) dialog box opens, where you can define how the variables will be processed when the template is used.

2. In the **Name** text box, specify the variable name to be used in the template body.

3. In the **Expression** drop-down list, specify the expression to be calculated by PhpStorm when the template is expanded. Do one of the following:
 - o Type a string constant in double quotes.
 - o Type a predefined function with possible arguments or select one from the drop-down list.

Tip

An argument of a function can be either a line constant or another predefined function.

4. To enable PhpStorm to proceed with the next input field, if an input field associated with the current variable is already defined, select the **Skip if defined** check box.
5. To arrange variables in the order you want PhpStorm to switch between associated input fields, use the **Move Up** and **Move Down** buttons.

See Also

Concepts:

- [Live Templates](#)

Procedures:

- [Creating Code Constructs by Live Templates](#)
- [Creating and Editing Live Templates](#)

Reference:

- [Live Templates](#)
- [Edit Template Variables Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Exporting and Importing Settings

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm enables you to preserve and share your working environment. You can archive and store your preferred IDE settings, put the settings file under version control and thus make it available to your colleagues. On the other hand, you can use the settings, defined by the other team members, or your own ones intended for a different usage. This section describes how to:

- [Export settings to a JAR archive](#)
- [Import settings from a JAR archive](#)

To export ide settings to a jar archive

1. On the main menu, choose **File | Export Settings**.
2. In the **Select Components to Export** dialog box that opens specify the settings to export by selecting the check boxes next to them. By default, all settings are selected.
3. In the **Export settings** to text box, specify the fully qualified name of the target archive. Type the path manually or click the **Browse** button  and select the target file in the **Select Path** dialog box that opens.

To import settings from a jar archive

1. On the main menu, choose **File | Import Settings**.
2. In the **Import File Location** dialog box that opens select the desired archive.
3. In the **Select Components to Import** dialog box that opens specify the settings to be imported, and click **OK**.

Note

By default, all settings are selected.

See Also

Concepts:

- [Project and IDE Settings](#)

Procedures:

- [Configuring IDE Settings](#)

Reference:

- [Settings Dialog](#)
- [Code Style](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Switching Between Schemes

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

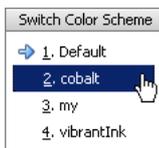
You can quickly switch between various schemes, keyboard layouts, and look&feels without actually invoking the corresponding page of the [Settings](#) dialog box.

To switch between schemes

1. Choose **View | Quick Switch Scheme** on the main menu or press `Ctrl+BackQuote`/`Command BackQuote`.
2. In the pop-up window that opens select the desired scheme (Colors and Fonts, Code Style etc.)



3. In the suggestion list, click the desired option.



See Also

Reference:

- [Settings Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Configuring Project Settings

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

`Project settings` refer to a set of preferences related to resources, file colors, version control options, code styles, etc.

You can configure project settings on the two possible levels:

- The level of a `template project`. The settings defined for a template project, apply to any project you create.
- The project level. The settings defined on this level apply to the current project only.

In this part:

- [Accessing Project Settings](#)
- [Configuring Individual File Encoding](#)
- [Configuring Code Style](#)
- [Exporting Project Code Style Settings](#)

Find descriptions of the other specific configuration procedures in the relevant procedural parts.

See Also

Concepts:

- [Project and IDE Settings](#)

Procedures:

- [Configuring IDE Settings](#)

Reference:

- [Settings Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Accessing Project Settings

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

This section describes simple steps required to access the project settings.

To access the project settings

1. On the main menu, choose:
 - o **File | Settings** for Windows and Linux.
 - o **PhpStorm | Preferences** for Mac OS.

Alternatively, do one of the following:

- o Press **Ctrl+Alt+S**Meta **Comma**.
 - o Click  on the main toolbar.
2. In the **Settings** dialog box that opens, select the desired page below the **Project Settings** section.

See Also

Concepts:

- [Project and IDE Settings](#)

Procedures:

- [Configuring Project Settings](#)
- [Configuring IDE Settings](#)

Reference:

- [Project Settings](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

```
ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js'; var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s);})();
```



PhpStorm 3.0.0 Web Help

Configuring Individual File Encoding

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

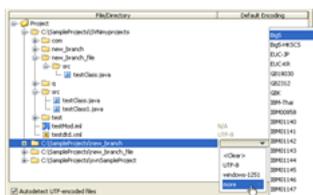
PhpStorm provides [individual approach to encoding](#) and suggests the following major ways to change encoding:

- [Using the File Encodings](#) page of the Settings dialog, for directories and for the files that do not contain encoding information.
- [Using the Status bar or menu command](#), for individual files that do not contain encoding information.
- [Using the editor](#), for individual files that contain encoding information.

PhpStorm also supports configuring [encoding for properties files](#).

To configure encoding for a directory or file without embedded encoding information

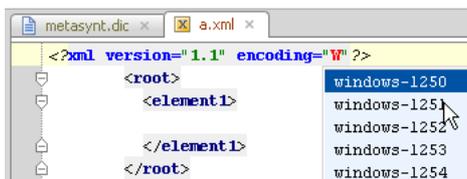
1. Open [Project Settings](#) dialog and select [File Encodings](#).
2. The **File/Directory** column shows the tree view of your project. The **Default Encoding** column shows encoding for directories or files. Click the **Default encoding** column for a directory or file you want to define encoding for, and then choose the desired encoding from the drop-down list:



Click thumbnail to view larger image.

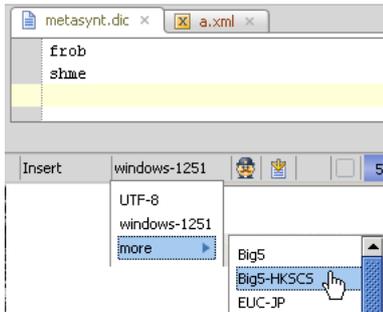
To change encoding of a file that contains explicit encoding

1. [Open the desired file](#) in the editor.
2. Change explicit encoding information. Use error highlighting to recognize wrong encoding and press **Ctrl+Space**Command **Space** to have a list of available encodings displayed:



To change encoding of a single file that doesn't contain explicit encoding

1. [Open the desired file](#) for editing editor.
2. Do one of the following:
 - o On the main menu, choose **File | Reload <current encoding> in another encoding**.
 - o Click file encoding on the [Status bar](#) to show the list of available encodings.
3. Select the desired target encoding from the submenu.



See Also

Concepts:

- [Encoding](#)
- [Project and IDE Settings](#)

Reference:

- [File Encodings](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Configuring Code Style

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

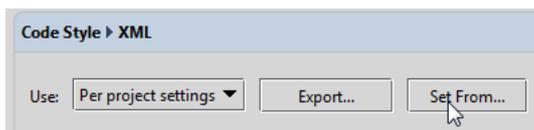
Obviously, you have a number of standards and guidelines that you have to obey when creating source code. PhpStorm can automatically maintain the required code style.

To configure code style on the global level

1. [Open the Project Settings](#) dialog box and click [Code Style](#).
2. Select the level to configure the code style settings on. If you select **Use per-project settings** from the drop-down list, your own code style will be used. The per-project code style settings are configurable at the level of the current project.
If you select **Use global settings**, the global code style will be applied to the current project. Note that project code style overrides the global one.
3. Browse through the nested pages to set code style preferences for the supported languages.

Tip

Note that for each of the supported languages, you can copy code style settings from any other language. To do that, just click the **Set From** button on top, and choose the language to copy code style from:



See Also

Procedures:

- [Improving Visibility of the Source Code](#)

Reference:

- [Code Style](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Exporting Project Code Style Settings

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

On the project level, you can define the code style that differs from the global one. If necessary, these code style schemes can be stored in XML files, in the `config\codeStyles` folder under the PhpStorm home directory, and shared.

To export project code style settings

1. [Open the Settings dialog box](#), and click **Project Code Style**.
2. In the [Project Code Style](#) dialog box that opens select the **Use per-project settings** option.

3. Click the **Export** button.
4. In the **Copy Project Scheme to Global List** dialog box, specify the name to save the code style scheme under. Then click **OK**.

See Also

Procedures:

- [Configuring Project Settings](#)
- [Configuring IDE Settings](#)

Reference:

- [Code Style](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
 - <http://youtrack.jetbrains.net/issues/WI> 
-

Creating and Managing Projects

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

In this part:

- [Creating New Project from Scratch](#)
- [Creating New Project from Existing Source Code](#)
- [Opening, Reopening and Closing Projects](#)
- [Opening Multiple Projects](#)
- [Cleaning System Cache](#)

See Also

Concepts:

- [Project](#)

Reference:

- [Project Structure](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
 - <http://youtrack.jetbrains.net/issues/WI> 
-

Creating New Project from Scratch

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

To create a new project from scratch

1. On the main menu, choose **File | New Project**.

Tip

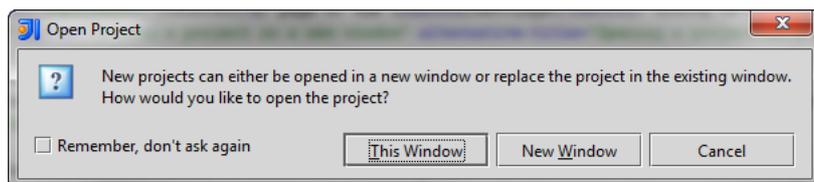
Alternatively, click the **New Project** button on the **Welcome** screen.

2. In the **Create New Project** dialog box that opens, specify the project name and location.

Tip

In a PHP project, the project root should be located under a directory which is "visible" for the Web server according to the settings in the Web server configuration file.

3. Click **OK**, when ready.
4. Specify whether you want the new project to be opened in a separate window or close the current project and reuse the existing window.



Refer to the section [Opening Multiple Projects](#) for details.

See Also

Concepts:

- [Project](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Creating New Project from Existing Source Code

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Tip

In a PHP project, the content roots should be located under a directory which is "visible" for the Web server according to the settings in the Web server configuration file.

You can set up a project around the existing source code created externally. PhpStorm analyzes the code base, adds the `.idea` directory with settings, and marks the project with the special icon .

Tip

In a PHP project, the project root should be located under a directory which is "visible" for the Web server according to the settings in the Web server configuration file.

To create a new project from existing source code

1. On the main menu, choose **File | Open Directory**.
2. In the **Select Path** dialog box, locate the directory that contains the desired source code. Note that applications created externally are marked with the regular directory icon .
3. Click **OK**.
4. Specify whether you want the new project to be opened in a separate window or close the current project and reuse the existing one. Refer to the section [Opening Multiple Projects](#) for details

See Also

Concepts:

- [Project](#)

Procedures:

- [Configuring Project Settings](#)
- [Completing Path](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Opening, Reopening and Closing Projects

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

When PhpStorm starts, the most recent [project](#) reopens by default (unless this option is disabled in [General settings](#)).

PhpStorm keeps the history list of the recent projects, from which you can select the desired one.

When the only open project is closed, the [Welcome screen](#) is displayed. In case of multiple projects, each one is closed with its frame.

To open an existing project

1. Do one of the following:
 - On the [Welcome screen](#), click **Open Directory** link. Alternatively, locate the desired project in your file chooser, and then drag and drop it onto the Welcome screen.
 - On the main menu, choose **File | Open Directory**.
2. In the **Select Path** dialog box, select the directory that contains the desired project.
3. Specify whether you want to open the project in a new frame, close the current project and reuse the existing frame, or open the new project in the same frame with the current one. Refer to the section [Opening Multiple Projects](#) for details.

Note

Alternatively, you can drag the desired project from your file chooser right to the Open Project dialog without locating it there. The respective file in the dialog will be found automatically.

To reopen a recent project

1. On the main menu, choose **File | Reopen**.
2. Select the desired project from the list of the recent ones.
3. Specify whether you want to open the project in a new frame, or reuse the current frame.

Tip

You can:

- Reopen a recent project from the [Welcome screen](#). To do that, just click the desired link under **Recent Projects**, or press **Alt+numberAlt+number**

- Terminate the project opening process, without waiting for its completion, by clicking **Cancel** in the progress window.
- Open projects and individual files from the [command line](#), by specifying the project or file name as a command line argument.

To close a project

- On the main menu, choose **File | Close Project**.

See Also

Concepts:

- [Project](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Opening Multiple Projects

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm allows you to work with several projects simultaneously, each one in a separate window. The projects are independent, and cannot share information, except for the Clipboard operations. All the projects run in the same instance of PhpStorm and use the same memory space.

Tip

This behavior is controlled by the check box **Confirm window to open project in** in the [General](#) page of the **Settings** dialog.

In this section:

- [Prerequisite](#)
- [Opening a project in a new window](#)

Prerequisite

Make sure that the check box **Confirm window to open project in** in the [General](#) page of the **Settings** dialog is selected.

To open a project in a new window

1. [Open a project](#), while another one is already open. PhpStorm prompts you to select whether you want to open the project in a new window, or close the current project and reuse the existing window.
2. Click the **New Window** button to leave the current project in its window and open the new one in a separate window.

Tip

Clicking **This Window** button results in closing the current project and opening the new one in the same window.

PhpStorm suggests to memorize your choice by selecting the check box **Remember, don't ask again**.

See Also

Concepts:

- [Project](#)

Reference:

- [General](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Cleaning System Cache

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm caches a great number of files, therefore the system cache may one day become overloaded. In certain situations the caches will never be needed again, for example, if you work with frequent short-term projects. Also, the only way to solve some conflicts is to clean out the cache.

To clean out the system caches

- On the main menu, choose **File | Invalidate Caches**. The **Invalidate Caches** message appears informing you that the caches have been invalidated and will be rebuilt on the next startup. To restart PhpStorm, click **Yes**.

Warning

Cleaning out the system caches causes a complete rebuild of all the projects ever run in the current version of PhpStorm.

See Also

Concepts:

- [Project](#)

Procedures:

- [Configuring IDE Settings](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Configuring Project Structure

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

To access project structure, use the [Directories](#) page of the Settings dialog, which is invoked by  icon on the main toolbar, or keyboard shortcut `Ctrl+Alt+SMeta Comma`.

In this section:

- [Configuring Content Roots](#)
- [Configuring Folders Within a Content Root](#)
- [Excluding Files from Project](#)

See Also

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

```
ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js'; var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s); });
```



PhpStorm 3.0.0 Web Help

Configuring Content Roots

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

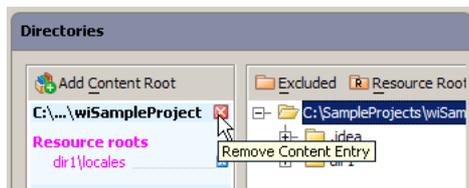
Any project contains at least one [content root](#) created together with the project. You can create additional content roots as well as remove the unnecessary ones.

To create a new content root

1. [Open the Project Settings](#).
2. In the **Projects** pane of the [Project Structure page](#), click the project you want to configure content roots for.
3. On the **Directories** page, click the **Add Content Root** button .
4. In the **Select Path** dialog box that opens, locate the desired directory and click **OK**.

To remove a content root

1. [Open the Project Settings](#).
2. In the **Projects** pane of the [Project Structure page](#), click the project you want to configure content roots for.
3. On the **Directories** page, select the content root to be deleted.
4. Click the **Remove Content Entry** button .



5. Confirm deletion.

See Also

Concepts:

- [Contents](#)

Reference:

- [Directories](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

```
ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js'; var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s); });
```



PhpStorm 3.0.0 Web Help

Configuring Folders Within a Content Root

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

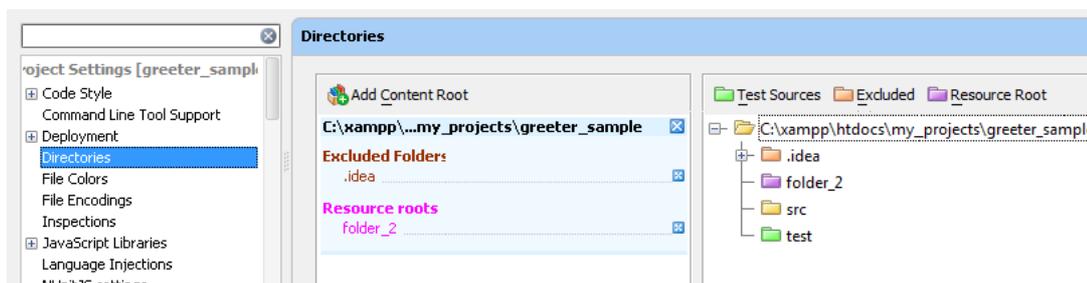
Within a content root, PhpStorm can distinguish between the folders that contain production source code, **Test Source** code, **Resource Root** folders, which contain with Web resources, **Excluded** folders, which are invisible for PhpStorm and therefore are ignored while searching, parsing, watching etc. To invoke this distinction, you can mark any folder below a content root as **Resource** or as **Excluded**.

In this section:

- [Marking directories under content root.](#)
- [Returning directories to their regular status.](#)

To mark a folder under a content root as non-source

1. [Open the Project Structure](#) settings.
2. In the **Projects** pane of the [Project Structure page](#), click the project you want to configure content roots for.
3. In the [Directories](#) page click the desired content root. The directories under this content root are displayed as a tree view in the right-hand pane.
4. Select the directory you want to mark and do one of the following:
 - o To have PhpStorm consider the contents of the selected folder as unit tests, click the **Test Sources** toolbar button or choose **Test Sources** on the context menu of the selection.
 - o To have PhpStorm ignore the selected directory during indexing, parsing, code completion, etc., click the **Excluded** toolbar button or choose **Excluded** on the context menu of the selection.
 - o To have PhpStorm "see" the selected directory without involving it in indexing, parsing, code completion, etc., click the **Resource Root** toolbar button or choose **Resource Root** on the context menu of the selection.



Tip

Alternatively, invoke the context menu command from the [Project Tool Window](#). Right-click the directory in question, choose **Mark Directory As node**, and choose **Mark as <directory status>**.

To return a folder to its regular status, do one of the following:

- Select the directory in question in the list of folders under the content root, and click :



- Click the folder's status icon once more.
- Choose the corresponding command on the context menu of the directory.

Tip

Alternatively, invoke the context menu command from the [Project Tool Window](#). Right-click the directory in question, choose **Mark Directory As node**, and choose **Unmark as <directory status>**.

See Also

Concepts:

- [Contents](#)

Procedures:

- [Configuring Content Roots](#)

Reference:

- [Directories](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

2.1+

Excluding Files from Project

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Sometimes you might need to exclude a single file from your project, so that it will be ignored by inspections, code completion, etc. This is done using the **Mark as plain text** action.

When a file is marked as plain text, PhpStorm does not use it anymore for code completion and navigation. Such file is shown as plain text in the editor, and is denoted with a special icon  in the [Project tool window](#).

The reverse action is also available: you can return a file to its original type, using the **Mark as <file type>** action.

To mark a file as plain text

1. In the [Project tool window](#), select the desired file.
2. On the context menu of the selection, choose **Mark as plain text**.

To mark file with its regular type

1. In the [Project tool window](#), select the desired file, marked with  icon.
2. On the context menu of the selection, choose **Mark as <file type>**.

See Also

Concepts:

- [Contents](#)

Reference:

- [File Types Recognized by PhpStorm](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Populating Projects

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts:  

You populate your project by creating new elements of various types (directories and files). PhpStorm suggests the following alternative ways of accessing the corresponding functionality:

- **File | New** in the main menu.
- The **New** context menu command.
- The keyboard shortcut `Alt+InsertCommand N`.

As a result, the **New** menu is shown in which you select the type of an element to be created.

In this section:

- [Creating Directories](#)
- [Creating Files from Templates](#)
- [Creating Empty Files](#)

See Also

Concepts:

- [Project](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

```
ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js'; var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s); });
```



PhpStorm 3.0.0 Web Help

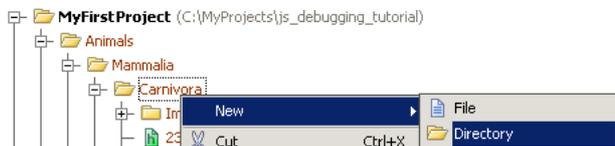
Creating Directories

Previous | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

PhpStorm enables you to create directories recursively, thus producing a whole directory structure.

To create a new directory

1. In the **Project** tool window, select the destination directory node.
2. On the context menu of the selection, choose **New | Directory**, or press `Alt+InsertCommand N` to invoke the context menu.



3. In the **New Directory** dialog box that opens specify the directory name. You can also specify nested directories; in this case, use slashes as the delimiters. Click **OK**. PhpStorm creates the new directory or directory structure.



[Click thumbnail to view larger image.](#)

See Also

Reference:

- [Project Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wiki>
- <http://youtrack.jetbrains.net/issues/WI>

Creating Files from Templates

Previous | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

PhpStorm provides [file templates](#) for most of the [languages that it supports](#). This lets you create the files with the initial content appropriate for the file purpose. For example, there are file templates for HTML/HTML5/XHTML, XML, and JavaScript files, PHP classes and files, XSLT style sheets, and other supported file types.

Generally, the file name extension for a template-based file is set automatically so you don't need to specify it. For example, if you create a JavaScript file, it gets the `.js` extension. New HTML and XML files get the `.html` and `.xml` extensions respectively. New XSLT style sheets get the `.xsl` extension.

Sometimes, you are given an opportunity to select the desired extension from the list. This, for example, is the case when creating PHP classes.

To create a new file from a template

1. In the **Project** tool window, select the directory in which you want to create a new file. Choose **File | New** or press `Alt+InsertCommand N`.

Alternatively, right-click the corresponding directory and select **New** from the context menu.

2. Select the desired file type. Generally, all the options except **File** and **Directory** correspond to using a file template.

Note that certain options may correspond to a group of related file templates. For example, if you select to create a class, you'll be able to create a class, interface, etc. or, in other words, use one of the corresponding related file templates.

Note

An existing file template may be missing from the list if this is a custom template whose file name extension (template extension) does not match registered patterns of any of the recognized file types. In such a case, you may want to register the corresponding pattern for an existing recognized file type or add a new file type and register the corresponding pattern for this new type. For more information, see [Creating and Registering File Types](#).

3. In the dialog that opens, type the name of the file in the corresponding field. Note that you should not type the file name extension.

Specify other information as required. For example, you may be asked to define the values of custom variables if the corresponding file template contains such variables and their values are not currently set.

Note

Sometimes, you may want to change the auto-generated file name extension. To do that, use the [Rename refactoring](#) (**Refactor | Rename**).

See Also

Concepts:

- [File Templates](#)
- [Supported Languages](#)

Procedures:

- [Creating and Editing File Templates](#)

- [Creating Empty Files](#)

Reference:

- [File Templates](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Creating Empty Files

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Generally, all the files that you create when developing applications are [template-based](#). However, sometimes you may want to create empty files.

To create an empty file

1. In the [Project tool window](#), select the directory in which you want to create a new file. Choose **File | New** or press **Alt+Insert/Command N**.
Alternatively, right-click the corresponding directory and select **New** from the context menu.
2. Select **File**.
3. In the **New File** dialog, in the field under **Enter a new file name**, type the file name and extension and click **OK**.
4. If the extension you have specified is not associated with any of the file types recognized by PhpStorm, the **Register New File Type Association** dialog is displayed. In this dialog, you can associate the extension with one of the recognized file types. To do that, select the file type under **Open matching files in PhpStorm** and click **OK**. As a result, the extension is associated with the specified file type.
If there are no appropriate file types for the new extension, you may want to create a new file type and associate the extension with that type. For more information, see [Creating and Registering File Types](#).

See Also

Procedures:

- [Creating Files from Templates](#)
- [Creating and Registering File Types](#)

Reference:

- [Register New File Type Association Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Generating Code

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

This section describes how to quickly and accurately populate your source code with the complicated code constructs.

In this part:

- [Creating Code Constructs by Live Templates](#)
- [Creating Code Constructs Using Surround Templates](#)
- [Unwrapping and Removing Statements](#)
- [Wrapping a Tag. Example of Applying Surround Live Templates](#)
- [Surrounding Blocks of Code with Language Constructs](#)

See Also

Reference:

- [Advanced Editing](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Creating Code Constructs by Live Templates

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

This section describes how to generate code source code using [live templates](#).

Using Live Templates enables you to create such code constructs as

Open the [Live Templates dialog](#) (**File | Settings | IDE Settings | Live Templates**) in PhpStorm to explore the list of available templates.

To insert a live template in the source code

1. Place the caret at the desired position, where the new construct should be added.

2. Do one of the following:
 - On the main menu, choose **Code | Insert Live Template**.
 - Press `Ctrl+J` `Command J`.
 - Type some initial letters of the [template abbreviation](#) to narrow down the suggestion list to the matching occurrences only. Note that suggestion list may contain same abbreviations for different templates.
3. From the [suggestion list](#), select the desired template.
4. Press the template invocation key (this may be `SpaceSpace`, `TabTab` or `EnterEnter`, depending on the template definition). The new code construct is inserted in the specified position.
5. If the selected template is [parametrized and requires user input](#), the editor enters the `template editing mode` and displays the first input field highlighted with the red frame. Type your value in this frame and press `EnterEnter` or `TabTab` to complete input and pass to the next input field. After completing the last input field, the caret moves to the end of the construct, and the editor returns to the regular mode of operation.

Tip

It is also possible to type a template abbreviation, and then press `Ctrl+J` `Command J`.

See Also

Concepts:

- [Live Templates](#)

Procedures:

- [Creating and Editing Live Templates](#)
- [Wrapping a Tag. Example of Applying Surround Live Templates](#)
- [Zen Coding Support](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Creating Code Constructs Using Surround Templates

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac 

This section describes how to wrap fragments of code according to [surround templates](#).

To surround a block of code with a live template

1. In the editor, select the piece of code to be surrounded.
2. Do one of the following:
 - On the main menu, choose **Code | Surround With Live Template**.
 - Press `Ctrl+Alt+J` `Command Alt J`.
3. Select the desired surround template from the suggestion list.

PhpStorm supports two predefined live template variables: `END` and `$SELECTION$`.

- `END` indicates the position of the cursor after the template is expanded. For example, the template `return END` will be expanded into

```
return ;
```

with the cursor positioned right before the semicolon.

- `$SELECTION$` is used in `surround templates` and stands for the code fragment to be wrapped. After the template is expanded, the selected text is wrapped as specified in the template.

For example, if you select `EXAMPLE` in your code and invoke the "`$SELECTION$`" template via the assigned abbreviation or by pressing `Ctrl+Alt+T` `Command Alt T` and selecting the desired template from the list, PhpStorm will wrap the selection in double quotes as follows:

```
"EXAMPLE".
```

Note

You cannot edit the predefined variables `END` and `$SELECTION$`.

See Also

Concepts:

- [Live Templates](#)

Procedures:

- [Creating and Editing Live Templates](#)
- [Creating and Editing Template Variables](#)
- [Using Suggestion List](#)
- [Wrapping a Tag. Example of Applying Surround Live Templates](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 

- <http://youtrack.jetbrains.net/issues/WI>

Unwrapping and Removing Statements

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

PhpStorm enables you to quickly unwrap or extract expressions from the enclosing statements. This action is available for:

- [JavaScript](#)
- [XML and HTML](#)

To unwrap or remove a statement

1. Place the caret on the expression you want to extract or unwrap.
2. Choose **Code | Unwrap/Remove** on the main menu or press **Ctrl+Shift+DeleteCommand Shift Delete**. PhpStorm shows a pop-up window with all the actions that are available in the current context. Statements to be extracted are displayed on the blue background, statements to be removed are displayed on the grey background.
3. Click the desired action or select it using the up and down arrow keys and press **EnterEnter**.

See Also

Concepts:

- [Live Templates](#)

Procedures:

- [Creating and Editing Live Templates](#)
- [Creating and Editing Template Variables](#)
- [Surrounding Blocks of Code with Language Constructs](#)
- [Wrapping a Tag. Example of Applying Surround Live Templates](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Wrapping a Tag. Example of Applying Surround Live Templates

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

As an example of applying a surround template, let's wrap a piece of XML code with tags.

To surround a code fragment

1. [Open](#) the desired file for editing.
2. Select a code fragment:


```
<company>
  <name>Peter</name>
  <name>John</name>
  <name>Sarah</name>
</company>
```

3. Press invocation shortcut **Ctrl+Alt+JCommand Alt J**. PhpStorm suggests the following surround:


```
<company>
  <name>Peter</name>
  <name>John</name>
  <name>Sarah</name>
</company>
```



4. Select the desired template from the suggestion list. The code fragment is surrounded with empty tags:


```
<company>
  <>
  <name>Peter</name>
  <name>John</name>
  <name>Sarah</name>
</>
</company>
```

5. The caret rests within the opening one. On typing the tag name in the opening tag, the name is automatically reproduced in the closing tag:


```
<company>
  <staff>
  <name>Peter</name>
  <name>John</name>
  <name>Sarah</name>
  </staff>
</company>
```

See Also

Concepts:

- [Live Templates](#)

Procedures:

- [Creating Code Constructs by Live Templates](#)
- [Zen Coding Support](#)

Reference:

- [Live Templates](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

```
ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js'; var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s);})();
```



PhpStorm 3.0.0 Web Help

Surrounding Blocks of Code with Language Constructs

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

The **Surround with** feature allows you easily put expressions or statements within blocks or language constructs. This feature in PhpStorm applies to:

Context	Surround with
XML/HTML tags	<ul style="list-style-type: none"> • Tag • CDATA section • <% ... %> • Zen coding
JavaScript statements	<ul style="list-style-type: none"> • (expr) • !(expr) • if • if / else • while • do / while • for • try / catch • try / finally • try / catch / finally • with • function • { } • function expression
ActionScript statements	<ul style="list-style-type: none"> • if • if / else • while • do / while • for • try / catch • try / finally • try / catch / finally • with • function • { } • function expression
PHP statements	<ul style="list-style-type: none"> • if • if / else • while

Example

The screenshot shows two examples of the 'Surround With' feature. In the first, XML code is selected, and the context menu offers options like 'Surround with <tag></tag>' and 'Surround with CDATA section'. In the second, JavaScript code is selected, and the context menu offers options like 'if', 'if / else', 'while', 'do / while', 'for', 'try / catch', etc.

[Click thumbnail to view larger image.](#)

This screenshot shows the 'Surround With' context menu for ActionScript code. The code snippet is:


```
public class Demo extends Sprite {
    public function Demo() {
        var myVar:TextField = new TextField();
        myVar.text = "Hello, World!";
    }
}
```

 The context menu is open, showing options like 'if', 'if / else', 'while', 'do / while', 'for', 'try / catch', etc. The 'if / else' option is highlighted.

- do / while
- for
- foreach
- try / catch
- function

```
function multiply($string1, $string2) {
    return $string1.concatenate($string1,$string2);
}
```

Surround With
 1. {}
 2. if
 3. if / else
 4. while
 5. do / while
 6. for
 7. foreach
 8. try / catch
 9. function

To surround a block of code

1. Select the desired code fragment.
2. On the main menu, choose Code | Surround With, or press Ctrl+Alt+TCommand Alt T. A pop-up window displays the list of enclosing statements according to the context. [Click thumbnail to view larger image.](#)
3. Select the desired surround statement from the list. To do that, use the mouse cursor, or a shortcut key displayed next to each element of the list.

See Also

Concepts:

- [Live Templates](#)
- [Creating and Editing Template Variables](#)

Reference:

- [Advanced Editing](#)
- [Zen Coding Support](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Auto-Completing Code

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

This part describes the various techniques that enable you to speed up the editing process. These techniques include:

- [Basic code completion](#) on Ctrl+SpaceCommand Space.
- [Class names completion](#) on Ctrl+Alt+SpaceCommand Alt Space.
- [Type completion](#) on Ctrl+Shift+SpaceCommand Shift Space.
- [Completing punctuation](#) on EnterEnter.
- [Completing statements](#) with smart EnterEnter.
- [Completing paths](#) in the Select Path dialog box.
- [Expanding words](#) with Alt+SlashAlt Slash.
- Various tips and tricks on [using the suggestion lists](#) that appear on invoking code completion.

See Also

Reference:

- [Editor. Code Completion](#)
- [Advanced Editing](#)
- [Symbols](#)

Getting Started:

- [Familiarize Yourself with PhpStorm Editor](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

2.1+

Basic Code Completion. Completing Names and Keywords

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Basic code completion helps you complete the names of classes, methods, and keywords within the visibility scope. When you invoke code completion, PhpStorm analyses the context and suggests the choices that are reachable from the current position of the caret.

If basic code completion is applied to a part of a parameter, or variable declaration, PhpStorm suggests a list of possible names with regard to the type of the item.

Note

PhpStorm recognizes variables declared in PHP-specific ways, for example, as parameters in function calls or defined by reference argument.

Code completion covers supported file types and custom file types. However, PhpStorm does not recognize the structure of custom file types and suggests completion options regardless of whether a specific type is appropriate in the current context.

Depending on the way you accept code completion from the suggestion list, the completion is inserted to the left of the caret position (insert mode) or in place of the character at caret (replace

mode).

To automatically complete names in the current visibility scope, follow these general steps:

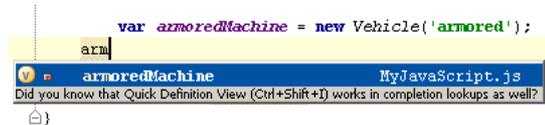
1. Type a name or a part of a name.

Tip

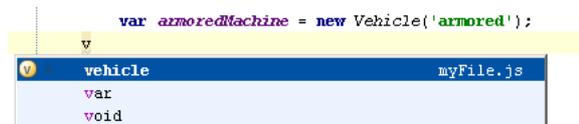
You can narrow down the [suggestion list](#) by typing the initial characters, or invoking code completion after a dot separator. In case of **CamelCase** names, type the initial letters only. So doing, PhpStorm automatically recognizes CamelHumps and matches them to the lower case letters.

2. Choose **Code | Complete Code | Basic** on the main menu or press **Ctrl+Space/Command Space**.

The image below shows a suggestion list containing a variable name:



The image below shows a suggestion list containing keywords:



3. Select the desired completion from the suggestion list, and do one of the following:
 - o Press **Enter/Enter** or double-click the desired choice to insert the completion to the left from the caret.
 - o Press **Tab/Tab** to replace the characters to the right from the caret.

Note

The option **Autocomplete common prefix** (**Settings | IDE Settings | Code Completion**) makes PhpStorm look for common prefixes and automatically complete them in the editor. This feature is useful for the classes with numerous similar-named members.

See Also

Procedures:

- [Using Suggestion List](#)
- [Navigating to Class, File or Symbol by Name](#)

Reference:

- [Editor. Code Completion](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

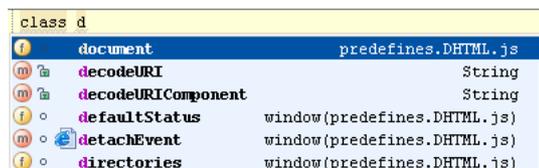
Class Name Code Completion

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Class name completion completes the names of classes, searching through the entire project. If the desired class is not yet imported, it will be imported automatically.

To complete a class name

1. Start typing. Note that you can type the first letters of the class name or the capital letters of the *camel hump* name only.
2. Choose **Code | Complete Code | Class Name** on the main menu or press **Ctrl+Alt+Space/Command Alt Space**.



3. Select the desired class from the suggestion list.

See Also

Procedures:

- [Using Suggestion List](#)

Reference:

- [Editor. Code Completion](#)

Getting Started:

- [Familiarize Yourself with PhpStorm Editor](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

```
ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js'; var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s); });
```



PhpStorm 3.0.0 Web Help

Smart Type Code Completion. Completing Code Based on Type Information

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Smart Type code completion filters the suggestion list and includes only those types that are applicable to the current context.

To invoke smart type code completion

Do one of the following:

- On the main menu, choose **Code** | **Complete Code** | **SmartType**.
- Press **Ctrl+Shift+Space** / **Command Shift Space**.

SmartType code completion automatically highlights selection in the suggestion list that seems most suitable for the current context. Such results display on a darker green background.



[Click thumbnail to view larger image.](#)

As you type, the suggestion list reduces to show matching entries only.

See Also

Procedures:

- [Using Suggestion List](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Completing Statements

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

The Smart Enter completion enables you to create syntactically correct code constructs.

To automatically complete a statement

- Start typing a statement and press **Ctrl+Shift+Enter** / **Command Shift Enter**.

```
function getVehicleInformation() {
  if (
}
  (expected)
```

The punctuation required in the current context is added and the caret moves to the next editing position.

```
function getVehicleInformation() {
  if (
}
  expression expected
  statement expected
```

After pressing **Ctrl+Shift+Enter**

See Also

Reference:

- [Code Style](#)
- [Editor](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Completing Path

- [Import](#) the desired taglib and insert all required import and reference statements.

To invoke the tag name completion

1. Press << and start typing the tag name. PhpStorm displays the list of tag names appropriate in the current context. Use the `ArrowUpArrowUp` and `ArrowDownArrowDown` buttons to scroll through the list.
2. Press `EnterEnter` to accept selection from the list, or continue typing the tag name manually.

Note

PhpStorm automatically inserts mandatory attributes according to the schema.

To insert a taglib declaration

1. Start typing the tag and press `Ctrl+Alt+SpaceCommand Alt Space`.
2. From the list of matching tags, select the desired one. The `uri` of the taglib it belongs to is displayed in brackets.

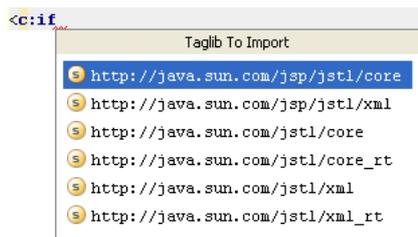


3. Select the desired taglib and press `EnterEnter`. PhpStorm adds the declaration of the selected taglib.

```
<%@ taglib prefix="h" uri="http://java.sun.com/jsf/html" %>
<h:inputText|...
```

To import a taglib

1. Start typing the taglib prefix and press `Alt+InsertCommand N`.
2. Select the desired taglib from the list of suggestions and press `EnterEnter`.



PhpStorm imports the selected taglib and adds the import statement automatically.

See Also

Procedures:

- [Creating Imports](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Hippie Completion. Expanding Words

[Previous](#) | [Next](#) | See Also | [Comments](#) | Shortcuts: (Mac)

This helpful feature analyses your text in the visible scope and suggests to complete a word with a keyword, class name, method or variable. So doing, the `prototype` is highlighted in the source code.

To expand a string at caret to an existing word

1. Type the initial string.
2. Press `Alt+SlashAlt Slash`. The editor shows the first suggested value.
3. Press `EnterEnter` to accept the suggested word, or continue pressing `Alt+SlashAlt Slash` until the desired word is found.

Example:

```
function concatenate($string1, $string2) {
    $result = $string1.$string2;
}

function concatenate($string1, $string2) {
    $result = $string1.$string2;
    return $result;
}
```

See Also

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

```
ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js'; var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s); }());
```



PhpStorm 3.0.0 Web Help

Using Suggestion List

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

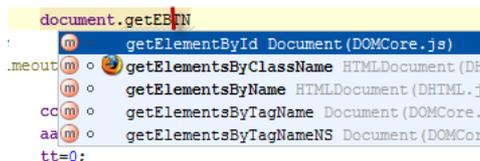
Any code completion or live template operation ends up with a suggestion list, from which you have to select the desired option. PhpStorm provides handy ways to make your selection, complete the source code, sort entries, and view reference information without leaving the suggestion list.

In this section, you will learn:

- [To navigate through the suggestion list and completed expression](#)
- [To complete selection](#)
- [To view reference information in the suggestion list](#)
- [To sort entries in the suggestion list](#)
- [To view hierarchy in the completion pop-up](#)
- [To close the suggestion list](#)

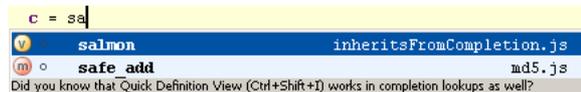
To navigate through the suggestion list and completed expression

- Use the mouse cursor or the UpUp and DownDown arrow keys to navigate through the suggestion list.
- Use the horizontal arrow keys in the completed expression to make the suggestion list wider (left arrow) or closer (right arrow):

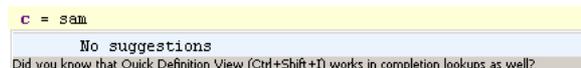


To complete selection

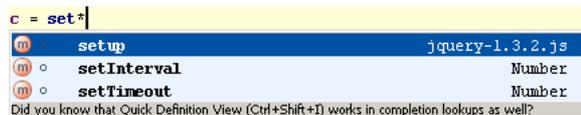
- Use EnterEnter to insert the selected string at the insertion point.
- Use TabTab to replace the string next to the caret with the selected one.
- Use Ctrl+Shift+EnterCommand Shift Enter to make the current code construct syntactically correct (balance parentheses, add missing braces and semicolons, etc.)
- It is possible to enter the desired element name manually when the input position is highlighted and the list of suggestions is open. As you type, the suggestion list shrinks so that only matching element names remain:



If you enter a name that is not used in the scope, PhpStorm will inform you with a pop-up window as shown in the figure below:



- Asterisk wildcard replaces any number of characters located anywhere in the completion string.



Tip

- Most suitable variants in the suggestion list are displayed on the green background on top of the list.
- Entries in bold font on top of the list denote symbols defined in the current class. All the other symbols are shown below.

To view reference information in the suggestion list

- Use the **Quick Definition View**. If you select an entry in a list and press **Ctrl+Shift+ICommand Shift I**, PhpStorm displays the following quick information pop-up window:



[Click thumbnail to view larger image.](#)

- **Quick Information View** works in a suggestion list. If you select an entry in a list, and press **Ctrl+QCommand J**, PhpStorm displays the following quick information pop-up window:



[Click thumbnail to view larger image.](#)

To sort entries in the suggestion list

- Click **A** or **π** in the lower-right corner of the list to toggle between sorting in alphabetical order, or by relevance.

Tip

- The sorting icon appears in the list if it is long enough. For few entries only sorting icon is not displayed.
- Use mouse only to click this icon.
- PhpStorm memorizes the type of sorting.
- Default behavior is defined in the [Code Completion](#) page of the Editor settings.

To view hierarchy in the completion pop-up

1. Start typing the source code, and press **Ctrl+Shift+SpaceCommand Shift Space** or **Ctrl+SpaceCommand Space**. The suggestion list appears.
2. While in the suggestion list, press **Ctrl+HCommand H**. The **Hierarchy** tool window shows the [type hierarchy](#) of the class, selected in the suggestion list.

To close the suggestion list

- Close the suggestion list with **EscapeEscape**, then enter your own code at the code completion position.

See Also

Procedures:

- [Viewing Structure and Hierarchy of the Source Code](#)
- [Auto-Completing Code](#)

Reference:

- [Editor. Code Completion](#)
- [Advanced Editing](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Creating and Optimizing Imports

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

In this section:

- [Creating Imports](#)
- [Optimizing Imports](#)

See Also

Procedures:

- [Generating Maven Dependencies](#)

Reference:

- [Optimize Imports Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Creating Imports

Previous | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

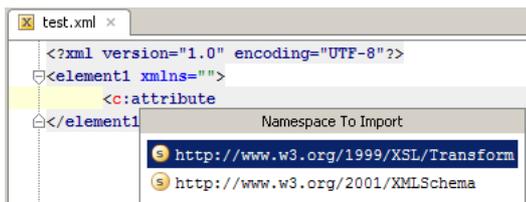
If you reference a namespace that has not been imported, PhpStorm helps you locate this file and add it to the list of imports. When you [type a tag with an unbound namespace](#), import assistant suggests to create a namespace and offers a list of appropriate choices.

To import a namespace, follow these general steps

1. Open the desired file for editing, and start typing a tag. If a namespace is not bound, the following prompt appears:



2. Press **Alt+Enter**. If there are multiple choices, select the desired namespace from the list.



PhpStorm creates a namespace declaration.

See Also

Procedures:

- [Optimizing Imports](#)

Reference:

- [Editor. Auto Import](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Optimizing Imports

Previous | Next | [See Also](#) | [Comments](#) | Shortcuts: Mac

Sooner or later, some of the imported namespaces become redundant to the code. Typically, you have to stop what you are doing, scroll to the head of the file, find the unused imports, and remove them. It is rather easy to forget to remove imports when you remove usages. PhpStorm provides the `Optimize Imports` feature, which enables you, whenever it is convenient, to remove unused imports from your current file, or from all files in the current directory at once.

To optimize imports, follow these steps

1. On the main menu, choose **Code | Optimize Imports**.

Tip

Alternatively, press **Ctrl+Alt+O** / **Command Alt O**.

2. In the [Optimize Imports](#) dialog box, specify where you want PhpStorm to remove unused import statements from.
 - o To have unused imports removed from the current file, select the **File** option.
 - o To have unused imports removed from all files in the current directory, select the **All files in directory...** option.
3. Click **Run**.

See Also

Procedures:

- [Creating Imports](#)

Reference:

- [Editor. Auto Import](#)
- [Optimize Imports Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Markup Languages and Style Sheets

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: (Mac )

Besides [JavaScript and AJAX](#), including related frameworks and technologies, PhpStorm supports editing HTML, CSS, SAS, XML, and XSLT files.

The markup languages and style sheets are integrated into the IDE and can use the most powerful editing features:

- Indentation (`Ctrl+Alt+ICommand Alt I`, `Ctrl+Alt+LCommand Alt L`).
- Intention actions (`Alt+EnterAlt Enter`).
- Viewing code structure (`Alt+7Meta 7`).
- Code completion (`Ctrl+SpaceCommand Space`).
- Navigation in the source code (`Ctrl+BCommand B`).
- Integrated documentation (`Ctrl+QCommand J`).
- Search for usages (`Alt+F7Alt F7`).
- Commenting lines and clearing line comments (`Ctrl+Slash Command Slash` or `Ctrl+Divide Command Divide`, `Ctrl+Shift+Slash Command Shift Slash` or `Ctrl+Shift+Divide Command Shift Divide`).
- Unwrapping and removing tags (`Ctrl+Shift+DeleteCommand Shift Delete`).
- Validation and syntax highlighting.

Note

All these features work if PhpStorm successfully locates the DTD or schema file. In this case, all the files are validated against the DTD or schema, and the editing conveniences become available. Without a DTD or schema, only the well-formedness check is possible.

These features for web contents work same way as for the other source files. Refer to the respective topics of the [Advanced Editing Features](#) part for the detailed descriptions of procedures, and to [Keyboard shortcuts](#).

Note

PhpStorm parses Web contents files according to the following specifications:

- **HTML**: specification is configurable in the **Default HTML language level** in the [Schemas and DTDs](#) page of the **Settings** dialog. By default, specification HTML 4.01 from W3C is assumed.
- **CSS**: specification CSS 2.1. The most common selectors are supported: universal selector `*`, type selectors `.a`, descendant selectors `.a.b`, child selectors `.a .b`, ID selectors `#b`, pseudo-classes and class selectors `DIV.warning`.
- PhpStorm uses **Xerces 2.6**, an XML parser developed by Apache Software Foundation Group.

In this part you will find information that is specific for the web content files only:

- [Generating DTD](#)
- [Generating XML Schema from Instance Document](#)
- [Generating Instance Document from XML Schema](#)
- [Referencing DTD or Schema](#)
- [Validating Web Content Files](#)
- [Viewing Styles Applied to a Tag](#)
- [Viewing Images](#)
- [Changing Color Values in Style Sheets](#)
- [Zen Coding Support](#)

See Also

Procedures:

- [Viewing HTML Source Code of a Web Page in the Editor](#)

Reference:

- [Code Style](#)
- [File Types](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Generating DTD

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

A [Data Type Definition \(DTD\)](#)  is required for running [structure validation checks](#) on a Web content file. PhpStorm can scan any XML file for the existing elements and attributes and generate a DTD for it.

To generate a dtd for an XML file

1. Open the desired file in the active editor tab.
2. On the main menu, choose **Tools | XML Actions | Generate DTD From XML File**. The resulting DTD is added as a section above the first line of the file.

See Also

Concepts:

- [Markup Languages and Style Sheets](#)

Procedures:

- [Validating Web Content Files](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Generating XML Schema from Instance Document

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

An [XSD \(XML Schema Definition\) Schema](#) is required for running [structure validation checks](#) on a Web content file. PhpStorm can scan any XML file for the existing elements and attributes and generate a Schema for it.

To have a schema generated based on an XML instance document

1. With the desired XML document opened in the active editor tab, choose **Tools | XML Actions | Generate XSD Schema from XML File** on the main menu.
2. In the [Generate Schema From Instance Document](#) dialog box that opens configure the Schema generation procedure:
 - In the **Instance Document Path** text box, specify the location of the file to be used as the basis for Schema generation. By default, the field shows the full path to the current file. Accept this suggestion or click the **Browse** button and select the desired file in the **Select Path** dialog box that opens.
 - In the **Result Schema File Name** text box, specify the name of the output file to place the generated Schema in.

Warning

PhpStorm suggests the name of the source XML document with the `.xsd` extension. If you type another name, make sure the extension is correct.

- Specify the location of the generated Schema. By default, the generated Schema file will be placed in the same directory as the source XML instance document. To specify another location, click the **Browse** button and select the desired path in the **Select Path** dialog box, that opens.
- From the **Design Type** drop-down list, select the way to declare elements and complex types.
- From the **Detect Simple Content Type** drop-down list, select the type to use for leaf text.
- In the **Detect Enumerations Limit** text box, type the number of occurrences to cause the Schema enumeration appearance.

Tip

To suppress Schema enumeration, specify 0.

See Also

Concepts:

- [Markup Languages and Style Sheets](#)
- [Plugins](#)

Procedures:

- [Markup Languages and Style Sheets](#)
- [Validating Web Content Files](#)
- [Generating Instance Document from XML Schema](#)

Reference:

- [Generate Schema from Instance Document Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Generating Instance Document from XML Schema

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

To generate an XML instance document from an XML schema

1. With the desired Schema (`.xsd`) file opened in the active editor tab, choose **Tools | XML Actions | Generate XML Document from XSD Schema** on the main menu.
2. In the [Generate Instance Document from Schema](#) dialog box that opens configure the XML instance document generation procedure:
 - In the **Schema Path** text box, specify the location of the Schema to base the XML document generation on. By default, the field shows the full path to the current file. Accept this suggestion or click the **Browse** button and select the desired file in the **Select Path** dialog box that opens.
 - In the **Instance Document Name** text box, specify the name of the output file to place the generated XML in.

Warning

PhpStorm suggests the name of the source XML document with the `.xml` extension. If you type another name, make sure the extension is correct.

- Specify the location of the generated document. By default, it will be placed in the same directory as the source Schema file. To specify another location, click the **Browse** button and select the desired path in the **Select Path** dialog box that opens.

- o From the **Element Name** drop-down list, select the local name of the global element to be used as the root of the generated XML document.
- o Specify whether to take restriction and uniqueness particles into consideration by selecting the corresponding check boxes.

See Also

Concepts:

- [Markup Languages and Style Sheets](#)
- [Plugins](#)

Procedures:

- [Markup Languages and Style Sheets](#)
- [Generating XML Schema from Instance Document](#)

Reference:

- [Generate Instance Document from Schema Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Referencing DTD or Schema

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

For an XML file to validate, you have to reference a DTD or a schema file. You can identify the DTD source in your xml file, for instance, like this:

```
<!DOCTYPE ejb-jar PUBLIC "-//Sun Microsystems, Inc.//DTD Enterprise JavaBeans 2.0//EN" "http://java.sun.com/dtd/ejb-jar_2_0.dtd" >
```

To reference a schema file, PhpStorm supports standard ways, in particular, `xsi` attribute.

If PhpStorm cannot find the specified DTD or schema, they are marked as errors. In this case special intention actions are suggested to:

- [Resolve the reference, if it is represented as a URL.](#)
- [Map a URI to a local file.](#)
- [Permanently ignore](#) it. An ignored URI will not be highlighted as an error any more, until removed from the ignore list.

To fetch a dtd or schema file

1. In the `xml` file, specify the URI of the DTD or schema. If the desired file is not found, the reference is marked as an error.
2. Press `Alt+EnterAlt Enter`.
3. In the suggestion list, choose **Fetch external resource**.

To map a dtd or schema uri to a local file

1. In the `xml` file, specify the URI of the DTD or schema. If the desired file is not found, the reference is marked as an error.
2. Press `Alt+EnterAlt Enter`.
3. In the suggestion list, select the option **Manually Set Up External Resource**. The [Resources](#) dialog box opens showing the new URI selected.
4. Click the **Edit** button.
5. In the **External Resources** dialog box, specify the path to the desired DTD or schema file and click **OK**.
6. Apply changes and close the **Resources** dialog box.

To permanently ignore a dtd or schema

1. In the `xml` file, specify the URI of the desired DTD or schema.
2. Press the `Alt+EnterAlt Enter` keyboard shortcut, if the specified DTD or schema is not found and marked as an error.
3. In the suggestion list, select the option **Ignore External Resource**. The URI is added to the Ignored Resources list of the **Resources** dialog box.

Tip

To remove a URI from the ignore list, open the **Resources** dialog box (**File** | **Settings** | **IDE Settings** | **Resources for Windows and Linux** or **PhpStorm** | **Preferences** | **IDE Settings** | **Resources for Mac OS**), select the desired URI in the **Ignored Resources** list, then click **Remove**.

See Also

Reference:

- [Schemas and DTDs](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Validating Web Content Files

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm performs two different [validity](#) checks:

- On-the-fly validation.
- Full validation.

On-the-fly validation is available for all Web content files and is performed as you edit the file. PhpStorm checks well-formedness, that is, detects various violations of syntax requirements, such as unclosed tags, wrong end-tag name, duplicate tags, unresolved links, etc. All encountered errors are highlighted in the editor.

However, this form of code validation is rather *soft*, that is, not all requirements are taken into account.

Full validation involves structure validation in addition to well-formedness check. Full validation is available for files that are associated with an [XSD \(XML Schema Definition\) Schema](#) or contain a [Data Type Definition \(DTD\)](#). PhpStorm checks whether the structure of your XML file complies with the structure defined in the corresponding DTD or Schema.

Check results are provided as a Message View.

To run full validation on an XML file

1. Open the desired XML file in the editor, or just select it in the [Project](#) tool window.
2. On the context menu, choose **Validate**.

See Also

Procedures:

- [Generating DTD](#)
- [Markup Languages and Style Sheets](#)

Reference:

- [Schemas and DTDs](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Viewing Styles Applied to a Tag

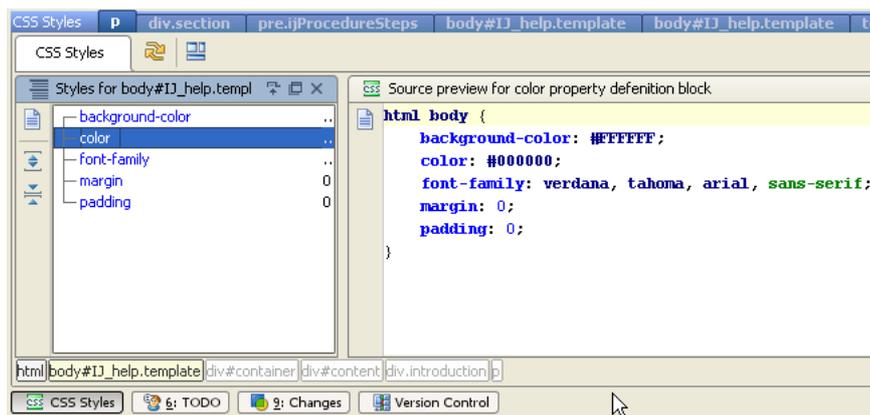
[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

For HTML, XHTML, and XML files, PhpStorm suggests a way to explore all styles applied to an arbitrary tag.

The results for each tag are displayed in the dedicated tabs of the **CSS Styles** tool window. Using this tool window, you can view the list of styles applied to a tag, and the definition of these styles. Besides that, you can navigate from style to the corresponding tag in the source code.

To show styles that are used for a tag

1. Open the desired file in the editor, and right-click the tag you want to explore for applied styles.
2. On the context menu, choose **Show Applied Styles for Tag**.
3. View results in a dedicated tab in the CSS Styles tool window:



See Also

Procedures:

- [Finding Usages in Project](#)

Reference:

- [Find Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Viewing Images

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

PhpStorm suggests several ways to view images, embedded in the HTML files. You can use [navigation to source](#), [open an image in an external graphical editor](#), or [preview images on-the-fly](#). The viewing modes are configurable in the [Images dialog](#).

To view an image in PhpStorm

1. In the Project tool window, select the desired image file.
2. On the context menu of the file, choose **Jump to Source**, or press **F4F4**.

Tip

Configure the path to the desired external editor in the External Editor section of the [Images dialog](#).

To view an image in an external editor

1. In the Project tool window, select an image file.
2. On the context menu of the file, choose **Open in external editor**, or press **Ctrl+Alt+F4Ctrl Alt F4**.

To preview an image, place the caret at a reference to an image in the editor, and do one of the following

- On the context menu, choose **Jump to Source**.
- Invoke the [Navigate to declaration](#) feature by pressing **Ctrl+BCommand B**.

See Also

Concepts:

- [Markup Languages and Style Sheets](#)

Procedures:

- [Navigating to Declaration or Type Declaration of a Symbol](#)

Reference:

- [Images](#)

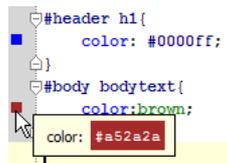
Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Changing Color Values in Style Sheets

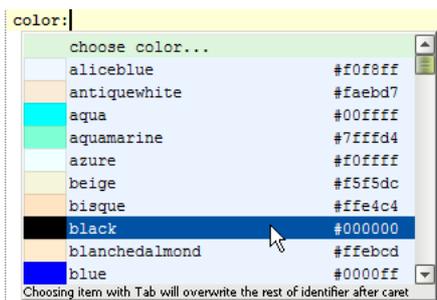
[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

In addition to the editing techniques that are common for all file types, PhpStorm provides specific techniques for **CSS**, **SASS**, **SCSS** and **LESS** files. In particular, it is possible to easily change color values without the necessity to memorize and type color codes. The `color` properties are marked with the icons of the corresponding color in the left gutter area of the editor. Use these icons to view colors and change color values. When you point with your mouse cursor to the color icon, the pop-up window appears, showing the color and its code.



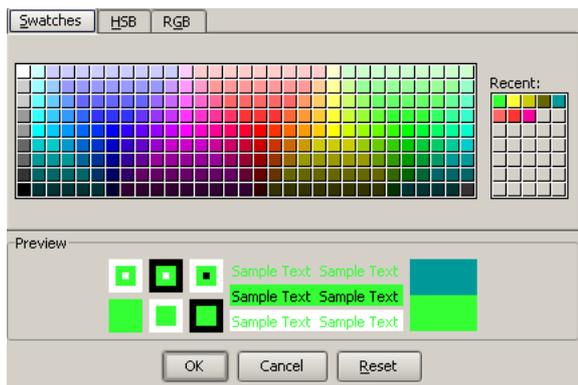
To choose color value in a style sheet

1. Open the desired style sheet for editing.
2. Type `color:`, and then press **Ctrl+SpaceCommand Space**.
3. Select the desired color value from the suggestion list, or choose `color...` to pick a custom one:



To change color value in a style sheet

1. Open the desired style sheet for editing, and locate color property to be changed.
2. Double-click the color icon in the left gutter of the editor.
3. In the Choose Color dialog box, pick the desired new color, and click **OK**:



See Also

Procedures:

- [Rename Refactorings](#)

Web Resources:

- <http://haml-lang.com/>
- <http://sass-lang.com/>
- <http://lesscss.org/>
- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Zen Coding Support

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Tip

PhpStorm supports Zen Coding up to version 0.7.

With PhpStorm, you can edit HTML and CSS code faster by applying Zen Coding features. PhpStorm provides two types of Zen Coding support.

- **Native Zen Coding** support allows you to generate HTML structures based on [selectors](#). You can use your own live templates as parts of complex templates. Basic Zen Coding features also work perfectly for XML structures.
- **Support of additional templates** including more than 200 different HTML, CSS, and XSL live templates. All of them are listed below the Zen Coding node on the [Live Templates](#) page of the Settings dialog box.

You can [configure the shortcut keys](#) to expand Zen Coding selectors and live templates. The [shortcut for expanding selectors](#) also by default applies to expanding Zen Coding live templates. You can [re-define this default setting](#) for each specific live template.

To enable native zen coding support

- [Open the IDE settings](#) and click **Smart Keys** below the **Editor** node. On the [Smart Keys](#) page that opens, select the **Enable Zen Coding** check box.

To enable support of additional HTML, css, and xsl live templates

1. [Open the IDE settings](#) and click **Live Templates**.
2. In the [Live Templates](#) page, that opens, select the check boxes next to the relevant template group. The available options are:
 - o Zen CSS
 - o Zen HTML
 - o Zen XSL

To configure shortcuts to expand abbreviations, do one of the following:

- To specify the shortcut to apply to selectors, open the [Smart Keys](#) page of the Settings dialog box and choose the desired option from the **Expand abbreviation with** drop-down list.
- To re-define the expansion key for a live template, open the [Live Templates](#) page, expand the **Zen Coding** node, and select the desired template. The focus moves to the [Template Text](#) area
- From the **Expand with** drop-down list, select the key to expand the template with.

Note

This setting does not override the [default setting](#) specified for native Zen Coding support; you will just get the possibility to expand the template using either of the specified keys.

See Also

Concepts:

- [Markup Languages and Style Sheets](#)
- [Live Templates](#)

Procedures:

- [Markup Languages and Style Sheets](#)
- [Auto-Completing Code](#)
- [Advanced Editing Procedures](#)

Reference:

- [Editor. Smart Keys](#)
- [Live Templates](#)

External Links:

- <http://blogs.jetbrains.com/webide/2010/03/zen-coding-support-in-webstormphpstorm/>
- <http://code.google.com/p/zen-coding/>

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Expanding Zen Coding Templates with User Defined Templates

Previous | Next | [See Also](#) | [Comments](#) | Shortcuts:

You can expand Zen Coding templates with your own live templates.

Suppose you have a template entry with the following template text:

```
<entry type="$TYPES$" >END$ <entry>
```

To generate a list of entries, you just need to type "entry-list<entry[number=\$] * 5" and press TABTAB. By default, the number attribute will be generated before type. To customize the position where it is generated, you need to add the ATTRS variable to your template, for example:

```
<entry type="$TYPES$" $ATTRS$>END$ <entry>
```

The ATTRS variable must have an empty string as the default value and should be skipped.

To expand a zen coding template with a user-defined template

1. In the [IDE settings](#), open the [Live Templates](#) page, and expand the Zen Coding template group.
2. Select the template to expand. The focus moves to the [Template Text](#) area where the fields show the settings of the selected template.
3. In the [Template Text](#) field, add the required text and variables to the template body.
4. Click the [Edit Variables](#) button. In the [Edit Template Variables](#) dialog box that opens, specify the default variable values in the [Default value](#) field and select the [Skip if defined](#) check box, where necessary.

See Also

Concepts:

- [Markup Languages and Style Sheets](#)
- [Live Templates](#)

Procedures:

- [Markup Languages and Style Sheets](#)
- [Creating and Editing Live Templates](#)
- [Creating and Editing Template Variables](#)

Reference:

- [Live Templates](#)

External Links:

- <http://blogs.jetbrains.com/webide/2010/03/zen-coding-support-in-webstormphpstorm/>
- <http://code.google.com/p/zen-coding/>

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Code Inspection

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm static code analysis tool helps you maintain and clean up your code without actually executing it.

When you inspect your code, you have to tell PhpStorm which types of problems you would like to search for and get reports about. Such configuration can be preserved as an `inspection profile`

Inspection profile defines the types of problems to be sought for, severity of these problems, and color scheme of alerts. Profiles are configurable in the [Inspections](#) dialog box.

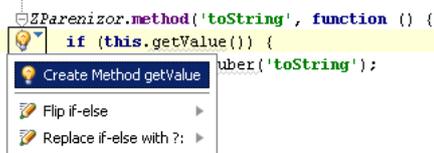
Numerous automated Code Inspections help you easily detect different inconsistencies. In PhpStorm you will find that all inspections are grouped by their goals and sense. Every inspection has an appropriate description. The most common tasks that are covered by the static code analysis are:

- Finding probable bugs.
- Locating dead code.
- Detecting performance issues.
- Improving code structure and maintainability.
- Conforming to coding guidelines and standards.
- Conforming to specifications.

Examples of code inspections

Unresolved JavaScript function or method

This inspection detects references to undefined JavaScript functions or methods.



Examples of php code inspections

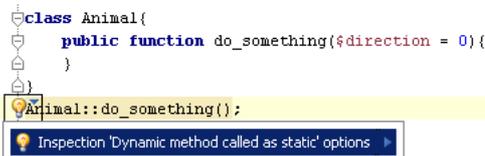
Unresolved include

This inspection detects attempts to include not actually existing files and suggests two quick fixes: to create a file with the specified name or use a PHPDOC annotation.



Dynamic method is called as static

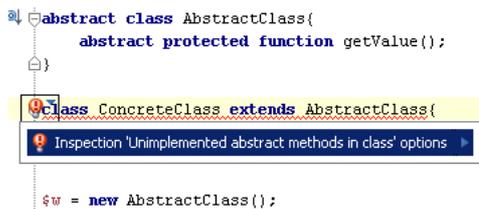
This inspection checks whether a call to a static function is actually applied to a static function.



The function `do_something()` is called as static while actually it is dynamic.

Unimplemented abstract method in class

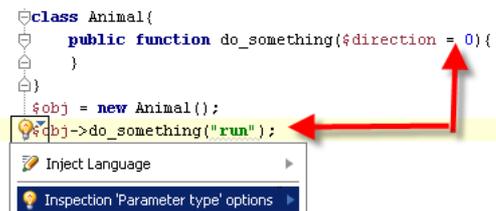
This inspection checks whether classes inherited from abstract superclasses are either explicitly declared as abstract or the functions inherited from the superclass are implemented.



The class `ConcreteClass` is inherited from an abstract class `AbstractClass` and has not been explicitly declared as abstract. At the same time, the function `Getvalue()`, which is inherited from `AbstractClass`, has not been implemented.

Parameter type

PHP variables do not have types, therefore basically parameter types are not specified in definitions of functions. However, if the type of a parameter is defined explicitly, the function should be called with parameters of the appropriate type.



The function `do_something` has the parameter of the type `integer` but is called with a `string`.

Undefined class constant

This inspection detects references to constants that are not actually defined in the specified class.

```

class Animal{
    public function do_something($direction = 0){
    }
}
$svar = Animal::NotExistingConst;

```

The constant NotExistingConst is referenced to as a constant of the class Animal, while actually it is not defined in this class.

Undefined constant inspection

This inspection detects references to constants that are not actually defined anywhere within the inspection scope.

```

$svar = UndefinedConst;

```

The referenced constant UndefinedConst is not defined anywhere within the inspection scope.

Undefined class

This inspection detects references to classes that are not actually defined anywhere within the inspection scope.

```

$svar = new NotExistingClass();

```

The referenced class NotExistingClass is not defined.

Undefined field

This inspection detects references to fields of a class that are not actually defined in this class.

```

class Animal{
    public function do_something($direction = 0){
    }
}
$obj = new Animal();
$svar = $obj->field;

```

The \$obj variable is an instance of the class Animal. The declaration of the \$svar contains a reference to the field field of the class Animal, which is not defined on this class.

Call of undefined function

This inspection detects references to functions that are not defined anywhere within the inspection scope.

```

class Animal{
    public function do_something($direction = 0){
    }
}
$obj = new Animal();
$svar = undefined_function($obj);

```

The called function undefined_function() is not defined anywhere within the inspection scope.

Undefined variable

This inspection detects references to variables that are not declared and initialized anywhere within the inspection scope. PHP does not require that each variable should be declared and initialized. PHP can initialize such variable on the fly and assign it the zero value. However, this inspection allows to detect discrepancies of this kind.

```

$svar = $undefined;

```

See Also

Concepts:

- [Intention Actions](#)

Reference:

- [Inspections](#)
- [Scopes](#)

External Links:

- http://www.jetbrains.com/idea/documentation/static_code_analysis.html

Getting Started:

- [Guided Tour Around PhpStorm User Interface](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

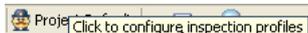
Accessing Inspection Settings

Previous | [Next](#) | [See Also](#) | [Comments](#)

Inspections and inspection profiles are editable in the [Inspections](#) dialog box. PhpStorm provides several ways to gain access to inspection settings.

To access inspections and profiles settings, do one of the following

- Click the current profile in the Status bar:



- On the main toolbar, click and then click **Inspections**.
- On the main menu, choose **File | Settings** for Windows and Linux or **PhpStorm | Preferences |** for Mac OS. Then click **Inspections**.
- In the editor, show the inspection alert and suggestion list, click the right arrow, and choose **Edit inspection profile settings** on the submenu.

See Also

Concepts:

- [Code Inspection](#)

Procedures:

- [Configuring IDE Settings](#)
- [Customizing Profiles](#)
- [Disabling Inspections](#)
- [Running Inspections](#)

Reference:

- [Inspections](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

```
ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js'; var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s); }());
```



PhpStorm 3.0.0 Web Help

Analyzing Inspection Results

Previous | [Next](#) | [See Also](#) | [Comments](#)

To analyze inspection results

- Examine inspection results in the [Inspection](#) tool window.



Click thumbnail to view larger image.

Each run of an inspection with a certain scope is displayed in its own tab in the tool window. Each group of inspections, for which the problems have been detected, is displayed in the left part of the tab. Click the nodes to review inspection results. Select a specific inspection to see its description in the right pane.

In the **Inspection** tool window you can:

- [Jump to the source of the problem.](#)
- [Export inspection results to an XML or HTML file.](#)
- [Suppress inspections.](#)
- [Configure inspection profiles.](#)
- [Resolve problems.](#)

See Also

Procedures:

- [Running Inspections](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Configuring Inspection Severities

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Inspection severity indicates how seriously the discrepancies detected by the inspection impact the project and determines how the detected discrepancies should be presented in the [inspection results](#). By default, you can [assign](#) one of the following severity degrees to an inspection:

- Server problem
- Typo
- Info
- Weak Warning
- Warning
- Error

You can [re-configure](#) their color schemes as well as [create custom severity](#) degrees.

Warning

Any changes made to the color scheme and the list of severity degrees apply to the current inspection only.

To assign a severity degree to an inspection

1. In the list of inspections, navigate to the desired one. If the inspection is disabled, select the check box next to it.
2. To specify the inspection severity, select the desired severity from the context menu of the inspection or from the **Severity** drop-down list in the **Options** area.

To configure severity degrees for an inspection

1. In the **Options** area, click the **Browse** button  next to the **Severity** drop-down list.



The **Severities Editor** dialog box opens.

2. In the **Severities Editor** dialog box, you can perform the following actions:
 - Change the color scheme of a severity: select it in the list and click the **Edit Settings | Colors & Fonts** button. In the [Colors and Fonts](#) dialog box that opens configure the color scheme and fonts to present the selected severity degree.
 - Add a custom severity: click the **Add** button  on the toolbar and type the name for the new severity in the **New Highlight Severity** dialog box that opens. Then click the **Edit Settings | Colors & Fonts** button and specify the color settings for the new severity in the [Colors and Fonts](#) dialog box that opens.
 - Use the **Move Up** button  and **Move Down** button  to change the priority of severity degrees by reordering them in the list.

Warning

The priority of a default severity cannot be changed.

- Remove a severity degree from the list: select it and click the **Remove** button .

Warning

Only custom severity degrees can be removed.

See Also

Concepts:

- [Code Inspection](#)

Procedures:

- [Accessing Inspection Settings](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Changing Highlighting Level for the Current File

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts:  

Use the [Status bar](#) to quickly re-configure highlighting for the file which is currently opened in the editor. With **HECTOR**, you can choose to highlight syntax problems, inspection issues, or turn off highlighting.

To change the highlighting level for the current file

1. Open the **Highlighting level** pop-up window by doing one of the following:
 - On the main menu, choose **Code | Configure Current File Analysis**.

- Press **Ctrl+Alt+Shift+H** / **Command Alt Shift H**.
 - Click the Hector icon  on the Status bar.
2. Move the slider to one of the three available positions that define the highlighting level:
- **None**: turn off problems highlighting in the editor.
 - **Syntax**: highlight syntax problems only.
 - **Inspections**: (default) highlight syntax problems and inspection issues.

See Also

Concepts:

- [Code Inspection](#)

Getting Started:

- [Status Bar](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wiki/>
- <http://youtrack.jetbrains.com/issues/WI>

```
ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js'; var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s); })(window);
```



PhpStorm 3.0.0 Web Help

Disabling Inspections

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

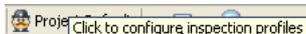
If you think that some inspections report about the problems that do not seem serious, you can [disable alerts for these inspections in the profiles](#) you use to inspect your code.

You can temporarily [disable alerts for certain inspections in the Inspection tool window](#), and thus clear the results report from unnecessary information.

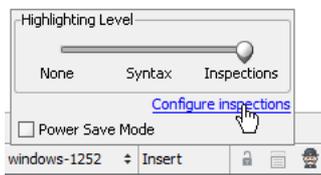
Another option is to [disable alerts "on-the-fly"](#).

To disable an inspection

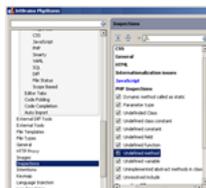
1. Open the [Inspections](#) page of the [Settings](#) dialog box or click the current profile in the Status bar:



2. Click the **Configure inspection link**:



3. Expand the desired inspection node.
4. Clear the check box next to the inspection, for which you want to disable alerts:



[Click thumbnail to view larger image.](#)

5. Apply the changes and close the dialog box.

To disable inspections from the inspection results report

1. In the [Inspection tool window](#), select the inspection you want to disable.
2. On the context menu, choose **Disable inspection**.
3. Press the filter button  to hide the disabled inspection alerts.

To disable an inspection on-the-fly

1. In the editor, press **Alt+Enter** / **Alt Enter** to reveal the inspection alert and suggestion list.
2. Select the inspection to be disabled, then click right arrow button or just press the right arrow key.
3. On the submenu, click **Disable <inspection name>**.

See Also

Concepts:

- [Code Inspection](#)
- [Intention Actions](#)

Procedures:

- [Suppressing Inspections](#)
- [Running Inspections](#)
- [Disabling Intention Actions](#)

Reference:

- [Inspection Tool Window](#)
- [Inspections](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Exporting Inspection Results

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Having inspected your source code, you can save the inspection results for further examination or for sharing with the colleagues. PhpStorm enables you to export inspection results to the HTML or XML format.

To export inspection results

1. On the toolbar of the [Inspection](#) tool window, click the **Export** button .
2. From the **Export To** context menu, select the target format. The available options are **HTML** and **XML**.
3. In the **Select Path** dialog box that opens specify the target directory to store the inspection results in.

See Also

Concepts:

- [Code Inspection](#)

Reference:

- [Inspection Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

```
ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js'; var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s); })(());
```



PhpStorm 3.0.0 Web Help

Resolving Problems

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

As you edit your source code inspections in the current file are performed on-the-fly, so you can fix the detected problems immediately as described in the section [Applying Intention Actions](#).

If you inspect a larger scope of source files, PhpStorm reports results in the **Inspection Results Report** section of the [Inspection](#) tool window. Where possible, PhpStorm suggests ways to fix the encountered problems in the optional **Problem resolution** field of the report. If this field is missing, it is the user's responsibility to resolve problems.

To fix a problem reported by the code inspection

Perform these general steps:

- In the [Inspection](#) tool window, select the inspection item you are interested in.
- If PhpStorm suggests some ways to resolve the problem, select one of the possible fixes below the **Problem resolution** field in the right pane of the tool window or click the **Apply a Quickfix** toolbar button , then select the desired fix from the context menu.



[Click thumbnail to view larger image.](#)

- If no resolutions are suggested, select the inspection item in question and choose **Jump to source** on the context menu of the selection and fix the problem manually.



[Click thumbnail to view larger image.](#)

See Also

Concepts:

- [Code Inspection](#)
- [Intention Actions](#)

Reference:

- [Inspection Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

```
ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js'; var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s); }());
```



PhpStorm 3.0.0 Web Help

Suppressing Inspections

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

For some reasons you may decide to leave as is the parts of your code that are reported as problematic. In this case you can suppress certain inspections for this code in general or for a particular scope. This can be done [from the Inspection tool window](#) or "on-the-fly".

To suppress an inspection

1. Select the desired inspection among inspections reported in the left pane of the [Inspection](#) tool window.
2. Click the  button on the toolbar, or just right-click the selected inspection.
3. On the submenu, choose the desired suppress action. For example:



[Click thumbnail to view larger image.](#)

To suppress inspection alerts on-the-fly

1. Click **Alt+Enter** **Alt Enter**, or click the light bulb icon to show the suggestion list.
2. In the suggestion list, select the scope, where the alert will be suppressed:



[Click thumbnail to view larger image.](#)

Tip

To change the inspection settings, select **Edit inspection profile settings** on the context menu and modify the settings in the [Inspections](#) dialog box, that opens.

See Also

Concepts:

- [Code Inspection](#)
- [Intention Actions](#)

Procedures:

- [Disabling Inspections](#)
- [Running Inspections](#)
- [Disabling Intention Actions](#)

Reference:

- [Inspection Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

```
ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js'; var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s); }());
```



PhpStorm 3.0.0 Web Help

Running Inspections

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Although code analysis is performed on-the-fly, you may want to run it for the whole project or a custom scope and examine the results in a friendly view. PhpStorm displays inspection results in the [Inspection](#) tool window.

Warning

Performing an inspection for a large scope, for example, the whole project, can be time consuming. It is recommended that you break your project into a number of smaller scopes.

To run inspections

1. Open the desired file in the editor. Alternatively select a file or directory in the [Project](#) tool window. The inspection scope will be confined to the opened file or the selection.
2. On the main menu, choose **Code | Inspect Code**. The [Specify Inspection Scope](#) dialog box opens.
3. In the **Inspection scope** area, specify the inspection scope.
 - o To have the source code of the entire project inspected, select the **Whole Project** option.
 - o To have the source code of currently opened file, or the file/folder selected in the **Project** view, select the **File/Module <name>** option.
 - o To apply inspection to a custom scope, select the **Custom scope** option, then select one of the pre-defined scopes from the drop-down list or click the **Browse** button  and configure your own scope in the [Scopes](#) dialog box.
4. To have test source files inspected too, select the **Include test sources** check box.
5. Specify the inspection profile to apply. Do one of the following:
 - o Select the desired profile from the **Inspection Profile** drop-down list.
 - o Click the **Browse** button  and configure your own profile in the [Inspections](#) dialog box.
6. Click **OK**.
7. Examine inspection results in the [Inspection](#) tool window.



[Click thumbnail to view larger image.](#)

Each run of an inspection with a certain scope is displayed in its own tab in the tool window. Each group of inspections, for which the problems have been detected, is displayed in the left part of the tab. Click the nodes to review inspection results. Select a specific inspection to see its description in the right pane.

In the **Inspection** tool window you can:

- o [Jump to the source of the problem.](#)
- o [Export inspection results to an XML or HTML file.](#)
- o [Suppress inspections.](#)
- o [Resolve problems.](#)

See Also

Concepts:

- [Code Inspection](#)

Procedures:

- [Analyzing Inspection Results](#)
- [Exporting Inspection Results](#)
- [Resolving Problems](#)

Reference:

- [Inspection Tool Window](#)
- [Specify Inspection Scope Dialog](#)
- [Inspections](#)
- [Scopes](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

2.0+

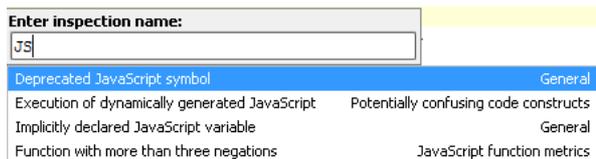
Running Inspection by Name

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts:  

Rather than running all enabled inspections, PhpStorm makes it possible to exactly specify the desired inspection by its name.

To run inspection by name

1. On the main menu, choose **Code | Run Inspection by Name**, or press **Ctrl+Shift+Alt+I** / **Command Shift Alt I**.
2. In the pop-up frame that opens, start typing the inspection name. As you type, the suggestion list shrinks to show the matching inspection only.



3. In the [Specify Inspection Scope](#) dialog box that opens, select the scope you want to inspect. This can be the entire project, the current file, a scope of your choice, or uncommitted files.
4. [Examine results](#) in the Inspection tool window.

See Also

Concepts:

- [Code Inspection](#)

Procedures:

- [Analyzing Inspection Results](#)

Reference:

- [Specify Inspection Scope Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Running Inspections Offline

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

In addition to running code inspections from the main menu, or from the context menus of the Project tool window or Commander, you can also launch the inspector from the command line, without actually running PhpStorm. This way you can perform regular code inspections as a part of your development process, which is especially important for the large projects. Inspection results are stored in XML format.

To launch code inspection from the command line

1. Open PhpStorm_HOME\bin directory.
2. Depending on your operating system, use one of the following launchers:
 - o For Windows: `inspect.bat`
 - o For UNIX and Mac: `inspect.sh`
3. Specify the arguments as required by the launcher.

See Also

Reference:

- [Inspection Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Intention Actions

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm helps you handle the situations when you use classes that haven't been imported, or methods that haven't been written etc., which can result in errors. When a possible problem is suspected, PhpStorm suggests a solution, and in certain cases can implement this solution (properly assign variables, create missing references and more.) Besides syntax problems, PhpStorm recognizes code constructs that can be optimized or improved, and suggests appropriate *intention actions*, denoted with the special icons.

In this section:

- [Intention Action Icons](#)
- [Types of Intention Actions](#)

Nested sections:

- [Applying Intention Actions](#)
- [Configuring Intention Actions](#)
- [Disabling Intention Actions](#)

Intention action icons

Item	Icon Description
Intention actions	A yellow bulb indicates that PhpStorm just proposes to alter your code. It covers a range of situations from warning correction to suggestions for code improvement

- suggested (like micro-refactorings).
- Specific intention action  This sign appears in the suggestion list before each specific intention action. If an intention action alert is disabled, the sign turns to . Disabled intention action is still available and can be enabled again.
- Quick-fix suggested  A red bulb with an exclamation mark indicates that PhpStorm suggests a way to fix an error. It is related to Create from usage intentions and Quick fixes.
- Disabled  Alert is disabled, but the intention action is still available and can be enabled again.

Types of intention actions

Find descriptions of specific intention actions in the [Intentions](#) page of the `Settings` dialog, where they are grouped according to the areas of their usage. Generally, intention actions can be divided into several categories, for example:

Create from Usage

This type of intention action creates new code items: classes, methods, etc. They are smart enough to analyze your code and provide actions suitable for a particular case. The main concept behind this type is that you can begin using new things without declaring them first. You are not taken away from your current task for mundane minutiae like creating declarations, new files, etc. which PhpStorm handles while you keep focused.

Quick Fixes

This type of intention action responds to common coding mistakes: using an improper access modifier, or an expression of the wrong type, or missing resources, etc. PhpStorm catches these kinds of problems as you type, and provides a quick way to fix them using Intentions Actions with appropriate suggestions for the error.



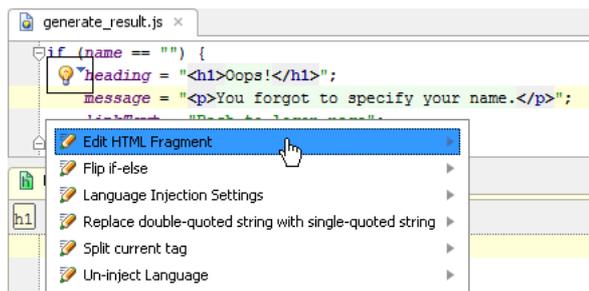
Micro-Refactorings

These intention actions appear for code that is syntactically correct, but can be structurally improved by such things as:

- Converting code constructs.
- Splitting declarations and assignments.
- Splitting or merging statements and tags, etc.

Edit <Injected Language> Fragment

For string literals that represent [language injections](#), the `Edit <Injected Language> Fragment` intention action is available. You can use this intention action to open the corresponding code fragment in a separate editor.



See Also

- Concepts:
- [Code Inspection](#)

- Reference:
- [Intentions](#)

- Web Resources:
- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>

Applying Intention Actions

Previous | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

PhpStorm displays an intention action alert, and it is your task to define which action (if any) should be performed.

To apply an intention action

1. Click the light bulb icon, or use the `Alt+Enter` keyboard shortcut to bring up the suggestion list.
2. Click the desired option, or use the arrow keys to select the option and press `Enter`.

See Also

- Reference:
- [Intentions](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Configuring Intention Actions

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

PhpStorm makes it possible to configure intention action settings either in the [Intentions](#) page of the Settings dialog, or "on-the-fly".

In this section:

- [To configure intention settings using the Settings dialog](#)
- [To configure intention settings on-the-fly](#)

To configure intention settings using the settings dialog

By default, all available intention actions that ship with PhpStorm, are enabled. By changing the Intention Settings, you can disable the actions that are not required for your current working environment.

1. On the main menu, choose **File | Settings** for Windows and Linux or **PhpStorm | Preferences |** for Mac OS.
2. Under the **IDE Settings** node, select **Intentions**.
3. In the [Intentions](#) page, clear the check boxes of the intention actions or action categories that you do not currently need.

Note

Selecting or clearing a category affects all intention actions in this category.

4. Apply changes and close the dialog.

To configure intention settings on-the-fly

1. In the editor, press **Alt+Enter** to reveal the inspection alert and suggestion list.
2. Select the action to be disabled, then click right arrow button or just press the right arrow key.
3. On the submenu, click **Disable <intention action name>**.

See Also

Reference:

- [Intentions](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Disabling Intention Actions

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

If an intention action is enabled, the alert shows automatically, when the caret rests on the problem-causing piece of code. You can disable alert for any type of intention actions, and show it by explicit invocation only.

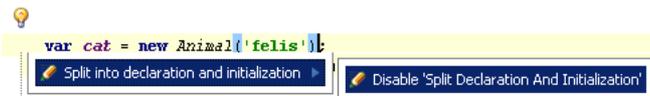
Disabled intention actions are marked with the grey light bulb icon . Though disabled, such actions are still available and can be applied to the source code.

Note

You can suppress intention actions related to inspections "on-the-fly", as described in the section [Suppressing Inspections](#).

To disable an intention action alert

1. Click **Alt+Enter**, or click the light bulb icon to show the suggestion list.
2. Select the action to be disabled and click the right arrow button.
3. On the submenu, click **Disable <intention action name>**:



Note

For the disabled intention action, the menu item changes to **Enable <intention action name>**.

See Also

Concepts:

- [Intention Actions](#)

Reference:

- [Intentions](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Viewing Pages with Web Contents

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

With PhpStorm, you can perform two opposite operations:

- Preview the output of your Web application in the browser to check whether the pages are rendered correctly. PhpStorm can display a page preview in a browser of your choice or you can switch between several browsers. PhpStorm currently supports previews in the following browsers:
 - Mozilla Firefox
 - Internet Explorer
 - Safari
 - Chrome
 - Opera
- View the HTML source code of a Web page in the PhpStorm editor.

In this section:

- [Configuring Browsers](#)
- [Previewing Pages with Web Contents in a Browser](#)
- [Viewing HTML Source Code of a Web Page in the Editor](#)

See Also

Reference:

- [Web Browsers](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Configuring Browsers

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

To have the preview of your application output displayed in the browser, you need to tell PhpStorm which browser you want to be used by default. This browser is called `PhpStorm default browser`.

To enable use of browsers

1. Open the [Web Browsers](#) page of the [Settings](#) dialog box.
2. Specify the browser to use by default.
 - To accept the default operating system browser as default for PhpStorm, select the **Use system default browser** check box.
 - To have another browser used as default, select the **Use** option and specify the location of the executable file of the required browser. Type the path manually or use the **Browse** button , if necessary.

Note

PhpStorm calls the default browser to render external resources as well.

3. Specify the way to have the browser launched.
 - To enable using a browser from the context menu, select the **Active** check box in the relevant section.
 - To have PhpStorm launch a browser using the default system command, select the **Default** check box in the relevant section.
 - To have PhpStorm launch a browser using the executable file, specify the path to it.

See Also

Procedures:

- [Configuring IDE Settings](#)

Reference:

- [Web Browsers](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Previewing Pages with Web Contents in a Browser

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

You can preview a file with Web contents in a browser. This can be the [PhpStorm default browser](#) specified in the [Web Browsers](#) section of the IDE settings or the one of your choice.

To preview an opened file in a web browser, do one of the following:

- On the main menu, choose **View | Open in Browser**. The current file opens in the default browser.
- Choose **View | Web Preview** on the main menu or, with the editor tab having the focus, press **Alt+F2**. Then select the desired browser from the pop-up menu:



- Hover your mouse pointer over the code to show the browser icons bar, and click the icon that indicates the desired browser:



Tip

If you don't want to see the icons toolbar, or would like to see the icons of just those browsers you are interested in, clear the **Active** check boxes for the unnecessary browsers in the [Web Browsers](#) section of the IDE settings.

See Also

Reference:

- [Web Browsers](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Viewing HTML Source Code of a Web Page in the Editor

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

You can open the HTML source code of any Web page in the PhpStorm editor.

Note

No matter which programming language was originally used to develop a page (for example, XML or PHP), PhpStorm shows the resulting HTML code.

To open the HTML source code of a web page in the editor

1. Choose **File | Open URL**.
2. In the **Open URL** dialog box that opens, specify the URL address of the desired Web page. Type the URL address manually or choose a previously specified one from the drop-down list.

See Also

Concepts:

- [Markup Languages and Style Sheets](#)

Procedures:

- [Markup Languages and Style Sheets](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Documenting Source Code in PhpStorm

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm provides convenient features for creating JSDoc and PHPDoc [comments](#).

In this part:

- [Enabling Creation of Documentation Comments](#)
- [Creating Documentation Comments](#)

See Also

Procedures:

- [Viewing Inline Documentation](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Enabling Creation of Documentation Comments

Previous | [Next](#) | [See Also](#) | [Comments](#)

To enable or disable automatic creation of documentation comments

1. [Open the Settings dialog box](#) and navigate to the [Editor. Smart Keys](#) page.
2. In the **Enter** section, select or clear **Insert documentation comment stub** check box.

See Also

Procedures:

- [Creating Documentation Comments](#)
- [Creating Documentation Comments](#)
- [Converters](#)

Reference:

- [Editor. Smart Keys](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Creating Documentation Comments

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

PhpStorm creates stubs of [JSDoc](#) comments on typing the opening tag `/**` and pressing `Enter`. If this feature is [applied to a method or a function](#), `@param` tags are created. In any other places PhpStorm adds an empty documentation stub.

[TODO patterns](#) inside documentation comments are also recognized.

Documentation comments in your source code are available for the [Quick Documentation Lookup](#) and open for review on pressing `Ctrl+Q`/`Command J`.

- [Example of JavaScript comment](#)
- [Enabling creation of documentation comments](#)
- [Creating a JSDoc comment block](#)

Example of JavaScript comment

Consider the following function:

```
function loadDocs(myParam1, myParam2){}
```

Type the opening documentation comment and press `Enter` to generate the documentation comment stub:

```
/**
 *
 * @param myParam1
 * @param myParam2
 */
```

To enable or disable automatic creation of documentation comments

1. [Open the Settings dialog box](#) and navigate to the [Editor. Smart Keys](#) page.
2. In the **Enter** section, select or clear **Insert documentation comment stub** check box.

To create a jsdoc comment block for a method or a function

1. Place the caret before the method or function declaration.
2. Type the opening block comment `/**` and press `Enter`.
3. Describe the listed parameters and return values.

Tip

PhpStorm checks syntax in the comments and treats it according to the [Code Inspections](#) settings.

See Also

Procedures:

- [Viewing Inline Documentation](#)

Reference:

- [Inspections](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Viewing Reference Information

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm facilitates quick and easy access to the API documentation, which you can view immediately in the editor or in an external browser. To gain access to the API documentation from within your project, make sure to attach archives or directories that contain sources of the library classes.

This section describes how to see definitions of symbols, display documentation references, and use the view parameter information feature:

- [Viewing Definition](#)
- [Viewing Inline Documentation](#)
- [Viewing External Documentation](#)
- [Viewing Method Parameter Information](#)

These features are available in all supported language contexts. For information on retrieving reference in JavaScript context, refer to section [Viewing JavaScript Reference](#).

See Also

Getting Started:

- [Familiarize Yourself with PhpStorm Editor](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

```
ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js'; var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s); })(());
```



PhpStorm 3.0.0 Web Help

Viewing Definition

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

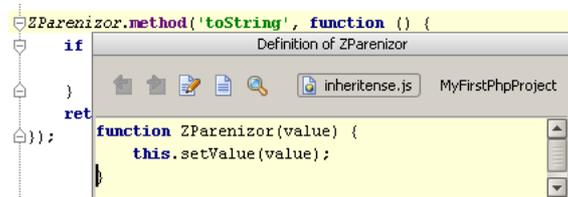
Quick Definition Lookup makes it possible to view definition of a symbol (tag, class, method/function, field, etc.) in a pop-up window.

For [markup languages](#), PhpStorm retrieves definitions of symbols from the specified DTD or schema:



[Click thumbnail to view larger image.](#)

In the JavaScript or PHP context, PhpStorm retrieves the definition of a symbol from the source code and displays the name of the file where the symbol is defined:



In this section:

- [Viewing definition of a symbol.](#)
- [Toolbar of the quick definition window.](#)

To view definition of a symbol at caret, do one of the following

- On the main menu, choose **View | Quick Definition Lookup**.
- Press **Ctrl+Shift+I** (or **Command Shift I** on Mac).
- Keeping the **Ctrl+Ctrl** key pressed, point with your mouse cursor to the symbol of interest, so that it turns to a hyperlink, with the definition of the symbol displayed in a tooltip. Clicking this hyperlink results in opening the respective definition page in the editor.

```
document.write("Fo function createAttributeNS(namespaceURI, qualifiedName)
document.createAttributeNS(x);
```

Use the icons on the toolbar of the pop-up window to navigate to the source code of the definition and view its usages.

Icon	Keyboard shortcut	Action
	Alt+Shift+LeftCommand Shift Left Alt+Shift+RightCommand Shift Right	Navigate to the previous/next screen in the definition pop-up window after using hyperlinks in the definition. Note On a Mac OS X computer, you can also use the three-finger right-to-left and left-to-right swipe gestures.
	F4F4	Open the source code of the definition for editing.
	Ctrl+EnterCommand Enter	Open the read-only source code of the definition.
	Alt+F7Alt F7	Open the usages of the detected definition across the project in the Find tool window.

See Also

Procedures:

- [Configuring JavaScript Libraries](#)
- [Viewing Inline Documentation](#)

Reference:

- [JavaScript Libraries](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

```
ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js'; var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s); })(0);
```



PhpStorm 3.0.0 Web Help

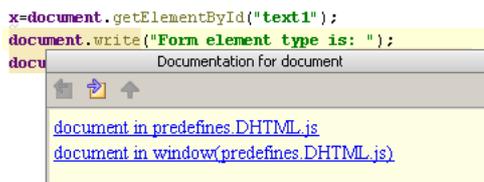
Viewing Inline Documentation

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Quick Documentation Lookup helps you get quick information for any symbol or just method signature information, provided that this symbol has been supplied with Documentation comments in the applicable format.

- For [markup languages](#), PhpStorm retrieves reference from the language specification according to the [Document Type](#) setting.
- For information on retrieving inline documentation in the JavaScript or PHP context, refer to sections [Viewing JSDoc Comments](#) and [Viewing PHPDoc Comments](#) respectively.

So doing, such documentation is properly rendered in the **Quick Documentation Lookup** window:



In this section:

- [Viewing quick documentation.](#)
- [Changing font size in the quick documentation window.](#)
- [Toolbar of the quick documentation window.](#)

To view documentation for a symbol at caret, do one of the following

- On the main menu, choose **View | Quick Documentation Lookup**.
- Press **Ctrl+QCommand J**.

Note

When you explicitly invoke code completion, then quick documentation for an entry selected in the suggestion list can be displayed automatically. The behavior of quick documentation lookup is configured in the [Code Completion](#) page of the Settings dialog.

To change the font size of quick documentation, do one of the following

- Click the button in the upper-right corner of the quick documentation window, and move the slider.

- Rotate the mouse wheel while keeping the `Ctrl` key pressed.

The **Quick Documentation Lookup** window helps navigate to the related symbols via hyperlinks, and provides a toolbar for moving back and forth through the already navigated pages, change font size, and viewing documentation in an external browser.

Icon	Keyboard shortcut	Action
	Alt+Shift+LeftCommand Shift Left Alt+Shift+RightCommand Shift Right	Navigate to the previous or next screen in the definition pop-up window after using hyperlinks in the definition. Note On a Mac OS X computer, you can also use the three-finger right-to-left and left-to-right swipe gestures.
	Shift+F1Shift F1	View documentation in an external browser.
		Click this button to show font size slider. Move the slider to increase or decrease the font size in the quick documentation window as required.

See Also

Procedures:

- [Creating Documentation Comments](#)
- [Configuring JavaScript Libraries](#)
- [Converters](#)
- [Viewing JavaScript Reference](#)
- [Viewing Inline Documentation](#)
- [Plugins](#)

Reference:

- [JavaScript. Libraries](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Viewing External Documentation

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

External documentation makes it possible to get additional information for the symbols at caret.

To view documentation for a symbol at caret in an external browser, do one of the following

- On the main menu, choose **View | External documentation**.
- Press `Shift+F1Shift F1`.
- While in the [Quick Documentation Lookup window](#), click

See Also

Procedures:

- [Viewing Inline Documentation](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

```
ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js'; var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s); });
```



PhpStorm 3.0.0 Web Help

2.1+

Viewing Method Parameter Information

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

This section describes how to configure the `View Parameter Information` behavior and use this functionality.

To view the method parameters

1. Place the caret anywhere within the call of the desired method or function.
2. Choose **View | Parameter Info** on the main menu or press `Ctrl+PCommand P`.

```
var vehicle = new Vehicle('lamborghini');
                    type
```

Note

Parameter information for methods defined through the `@method` [phpDocumentor tag](#) is also available:



[Click thumbnail to view larger image.](#)

To configure the behavior of the view parameter information functionality

1. [Open the IDE Settings](#) and click [Code Completion](#) below the **Editor** node.
2. In the **Parameter info** section, define the following options:
 - To have a complete method or function signature shown rather than a list of required types, select the **Show full signatures** check box.

Tip

Make sure to include the required third-party [libraries](#) in the project source path. Otherwise, names of the parameters will not be displayed.

- To have the list of parameter types for the called method or function shown automatically after a certain delay, select the **Auto pop-up (in ms)** check box and specify the time period in milliseconds.

See Also

Concepts:

- [Library](#)

Procedures:

- [Auto-Completing Code](#)

Reference:

- [Editor. Code Completion](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

3.0+

Navigating Through the Source Code

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm suggests various ways of navigation between the IDE components and within source code. PhpStorm's smart editor makes it possible to navigate across the source code using the code structure rather than plain scrolling. You can find your way in the source code using the method calls, declarations, errors, changes etc.

In this part:

- [Navigating with Bookmarks](#)
- [Navigating Between Files and Tool Windows](#)
- [Navigating Between IDE Components](#)
- [Navigating Between Methods and Tags](#)
- [Navigating Between Test and Test Subject](#)
- [Navigating to Action](#)
- [Navigating to Braces](#)
- [Navigating to Class, File or Symbol by Name](#)
- [Navigating to Declaration or Type Declaration of a Symbol](#)
- [Navigating to File Path](#)
- [Navigating to Implemented/Overridden or Implementing/Overriding Methods](#)
- [Navigating to Line](#)
- [Navigating to Next/Previous Change](#)
- [Navigating to Next/Previous Error](#)
- [Navigating to Recent File](#)
- [Navigating to Navigated Items](#)
- [Navigating with Navigation Bar](#)
- [Navigating with Structure Views](#)

See Also

Reference:

- [Navigation Between Bookmarks](#)
- [Navigation in Source Code](#)
- [Search](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

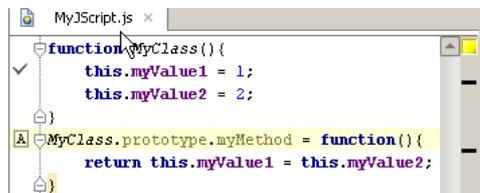
Navigating with Bookmarks

Previous | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

The editor of PhpStorm provides two types of bookmarks:

- **Anonymous bookmarks**, indicated by check signs ✓ in the left gutter. The number of anonymous bookmarks is unlimited.
- **Bookmarks with mnemonics** indicated by **U** or **S** icons in the left gutter. There can be only 10 numbered and 26 lettered bookmarks within a project.

All bookmarks are indicated with the black streaks in the marker bar:



Once created, the bookmarks enable you to easily jump to the desired location within a file, or across the entire project.

To navigate through the bookmarks within the current file, do one of the following

- On the main menu, choose **Navigate | Bookmarks | Next/Previous Bookmark**. The order of visiting bookmarks depends on their order in the collection of bookmarks in the [Bookmarks dialog](#).
- Click the black streak in the marker bar.

To navigate across a project using numbered bookmarks

- Use **Ctrl+NumberCtrl Number** where the <number> corresponds to the desired bookmark.

To navigate among all bookmarks in a project

1. On the main menu, choose **Navigate | Bookmarks | Show Bookmarks dialog**, or press **Shift+F11Shift F11**.
2. In the [Bookmarks dialog](#), select the target bookmark, and press **EnterEnter**.

Note

For your convenience, the target code preview is shown in the right pane of the dialog box.

See Also

Reference:

- [Bookmarks Dialog](#)

Getting Started:

- [Familiarize Yourself with PhpStorm Editor](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Managing Bookmarks

Previous | Next | [See Also](#) | [Comments](#) | Shortcuts: Mac

This section describes how to toggle, view and organize bookmarks. PhpStorm suggests [Bookmarks dialog](#) as a tool for managing bookmarks project-wide.

This section describes how to:

- [Create and delete bookmarks with mnemonics](#)
- [Toggle bookmarks](#)
- [View project bookmarks](#)
- [Delete bookmarks](#)
- [Change the order of bookmarks](#)

To create a bookmark with mnemonics

1. Place the caret at the desired line of code in the editor.
2. Press **Ctrl+F11Command F11** (alternatively, choose **Edit | Toggle Bookmark With Mnemonic** on the main menu), then press one of the keys among 0-9 or A-Z.

Tip

Pressing **Ctrl+F11Command F11** at the line where a bookmark already exists toggles this bookmark.

To toggle an anonymous bookmark on the current line, do one of the following

- On the main menu, choose **Edit | Toggle Bookmark**.

- Press `F11`.

To view all bookmarks in a project, do one of the following

- On the main menu, choose **Edit | Show Bookmarks**.
- Press `Shift+F11`.

To delete bookmarks in a project

1. Open the **Bookmarks** dialog (**Edit | Show Bookmarks**, or `Shift+F11`).
2. Select bookmarks and press `EditorDelete`.

To change the order of bookmarks

1. Open the **Bookmarks** dialog (**Edit | Show Bookmarks**, or `Shift+F11`).
2. Select the desired bookmark.
3. Use **Move Up** (`Ctrl+Up`) and **Move Down** (`Ctrl+Down`) buttons to shift the bookmark in the desired direction.

See Also

Procedures:

- [Navigating with Bookmarks](#)

External Links:

- [Bookmarks Dialog](#)

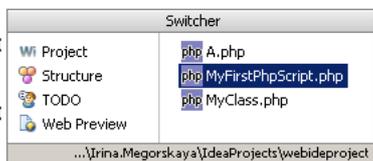
Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Navigating Between Files and Tool Windows

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

PhpStorm suggests a handy way to switch between files opened in the editor, split editor tabs, and tool windows (docked or floating). This is similar to the application switchers in the various operating systems. The switcher consists of two columns: the left one displays the list of tool windows, while the right one displays the list of files currently opened in the editor.



To switch between open files and tool windows

1. Press `Ctrl+Tab` or `Ctrl+Shift+Tab`.
2. Keeping `Ctrl` pressed, use the following keys:
 - `Up` and `Down` arrow keys, `Tab` or `Shift+Tab` to go up and down the list in both panes.
 - `Alt` to toggle between the right and the left panes of the switcher pop-up window.
 - `EditorDelete` or `BackSpace` to close the editor tab where the selected file is opened, or hide a tool window, it is shown.
3. Having selected the desired file or tool window, release the `Ctrl` key. The corresponding file or tool window gets the focus and the switcher pop-up window disappears.

Tip

If you want to close the switcher without selecting any file or tool window, just use the left or right arrow keys.

See Also

Reference:

- [Navigation Between IDE Components](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Navigating Between IDE Components

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Suppose you have selected a file or member in the one of the tool windows, and would like to quickly find it in the Project view. The **Select Target** pop-up menu moves the focus to the selected

component of PhpStorm:

- [Project tool window](#)
- [Structure tool window](#)
- [Navigation Bar](#)
- [Changes tool window](#)
- [Version Control tool window](#)
- [Data Sources tool window](#)

To navigate to the desired component

1. Press **Alt+FlAlt F1** to show the **Select Target** pop-up menu.
2. Use arrow keys or the mouse pointer to select the desired component. If your target is the **Project tool window**, you can select the desired view: Project, Scopes, etc.



Tip
 Double-click selected file or member in one of the tool windows, or press **EnterEnter** to open it in the editor.

See Also

- Reference:
- [Navigation Between IDE Components](#)

- Getting Started:
- [Navigating Through the Source Code](#)

- Web Resources:
- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>

Navigating Between Methods and Tags

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Since your source code can contain numerous methods, it is convenient to navigate to the beginning of the next or previous method. In the Web contents, this feature enables navigating between tags.

To navigate to the next or previous method or tag, do one of the following:

- On the main menu, choose **Navigate | Next Method / Previous Method** respectively.
- Use **Alt+UpCommand Up / Alt+DownCommand Down** keyboard shortcuts.

Tip

- The **Show Method Separator** option (**File | Settings | IDE Settings | Editor | Appearance for Windows and Linux or PhpStorm | Preferences | IDE Settings | Editor | Appearance for Mac OS**) adds a line between adjacent methods and thus improves visibility of the source code.
- In HTML files with the JavaScript tags, this behavior depends on the caret location. If the caret rests inside a JavaScript block, this means of navigation enables jumping between JavaScript functions. If the caret rests on a `<script>` tag, then navigation is performed between the tags.

See Also

- Reference:
- [Navigation Between IDE Components](#)

- Getting Started:
- [Navigating Through the Source Code](#)

- Web Resources:
- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>

Navigating Between Test and Test Subject

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

[PHPUnit testing](#) support in PhpStorm provides the possibility to navigate between a test and test subject.

To jump from a test to its test subject

1. Open the desired test class in the editor.
2. On the main menu or on the context menu of the editor, choose **Navigate | Test Subject**. Alternatively, press `Ctrl+Shift+T` `Command Shift T`. The test subject for the current test class opens in the dedicated tab of the editor and gets the focus.

To jump from a class or file to its test

1. Open the desired class in the editor.
2. On the main menu or on the context menu of the editor, choose **Navigate | Test**. Alternatively, press `Ctrl+Shift+T` `Command Shift T`. The test for the current class or files opens in the dedicated tab of the editor and gets the focus.

See Also

Procedures:

- [Testing](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

3.0+

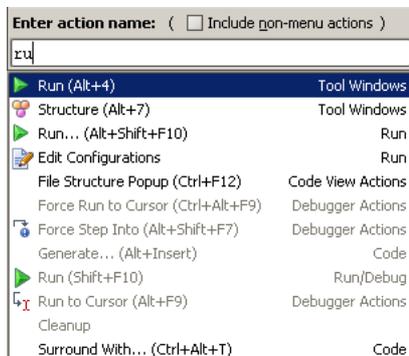
Navigating to Action

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

PhpStorm helps you quickly find the desired action, without digging through the menus and toolbars. The concept *action* covers the commands of the main menu and various context menus, commands performed through the toolbar buttons of the main toolbar and tool windows.

To find an action

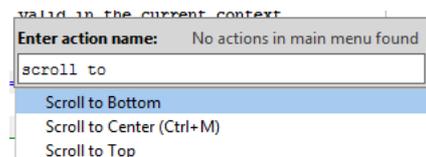
1. Choose **Help | Find Action** on the main menu or press `Ctrl+Shift+A` `Command Shift A`.
2. In the pop-up window that opens start typing the desired action name. As you type, the suggestion list displays the matching names of actions. The actions that are not valid in the current context are displayed gray.



3. Double-click the desired entry in the suggestion list or select it using the arrow keys. Then press `Enter`.

Tip

This way you can invoke the *Scroll to Top* and *Scroll to Bottom* actions:



These actions are not mapped to certain keyboard shortcuts, neither they appear in the menus. If necessary, configure keyboard shortcuts for these actions as described [here](#).

See Also

Getting Started:

- [Guided Tour Around PhpStorm User Interface](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Navigating to Braces

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

The **Go to braces** command helps you jump to the [borders of the current code block](#). With the command, you can also quickly navigate to the [borders of the higher code blocks](#).

To navigate to the borders of a code block, do one of the following:

- To navigate to the code block start, press `Ctrl+OpenBracketCommand` `OpenBracket`, with the caret anywhere inside the code block.
- To navigate to the code block end, press `Ctrl+CloseBracketCommand` `CloseBracket`, with the caret anywhere inside the code block.

To navigate to the borders of the closest higher code block, do one of the following:

- To jump to the higher code block start, press `Ctrl+OpenBracketCommand` `OpenBracket`, with the caret at the [current code block opening brace](#).
- To jump to the higher code block end, press `Ctrl+CloseBracketCommand` `CloseBracket`, with the caret at the [current code block closing brace](#).

Tip

Practically, you can just press `Ctrl+OpenBracketCommand` `OpenBracket` or `Ctrl+CloseBracketCommand` `CloseBracket` as many times as you need, until the caret is positioned at the start or end of the desired code block.

See Also

Reference:

- [Navigation in Source Code](#)

Getting Started:

- [Navigating Through the Source Code](#)
- [Familiarize Yourself with PhpStorm Editor](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

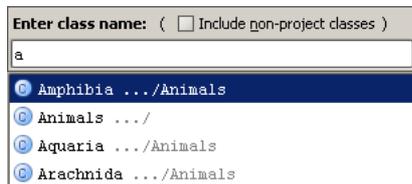
Navigating to Class, File or Symbol by Name

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

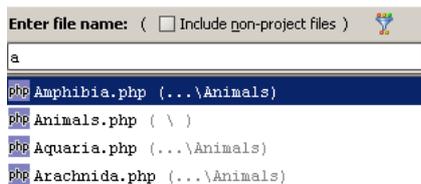
Navigate commands enable you to quickly jump to the desired classes, files, or symbols specified by names. PhpStorm suggests a look-up list of matching names, from which you can select the desired one, and open it in the editor. This navigation honors `CamelCase` and `snake_case` capitalization and wildcards. Refer to the [tips](#) for detailed list of available techniques.

To navigate to a class, file or symbol with the specified name

1. On the main menu, choose **Navigate | Class, File, or Symbol** respectively, or use the following shortcuts:
 - o Class: `Ctrl+NCommand` `N`



- o File: `Ctrl+Shift+NCommand` `Shift N`



Tip

You can narrow down the search scope by selecting the file types to search in. Just click the filter button , and clear the check boxes next to the files types you are not interested in.

- o Symbol: `Ctrl+Shift+Alt+NCommand` `Shift Alt N`

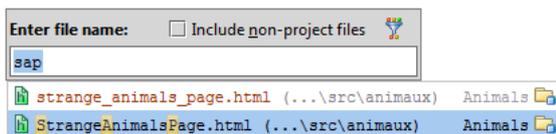


2. In the pop-up window, start typing the desired name. As you type, the suggestion list shrinks, displaying the matching names only.
3. Double-click the desired entry in the suggestion list, or select it using the arrow keys, and press **Enter**.

Tip

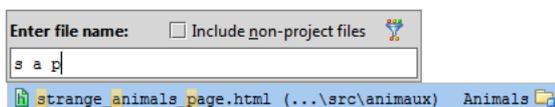
While working in the navigation pop-up window, use the following helpful techniques:

- Include non-project files in the look-up list and thus make available matching files from SDKs and libraries.
- If the look-up list is too long, type more characters to shrink it, or click the ellipsis sign at the end of the list, to reveal its next portion.
- Type the initial letters of the **CamelHumps** names, for example:

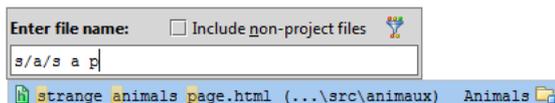


Note that PhpStorm automatically recognizes **CamelHumps** and matches them to the lower case letters.

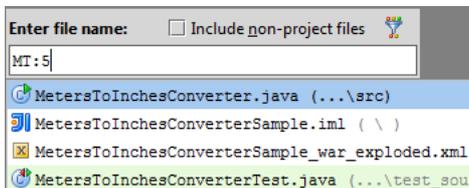
- Type any letters separated with spaces for **snake_case** names, for example:



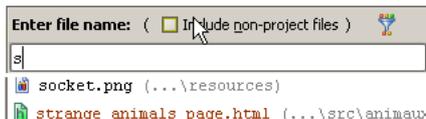
- In the navigation to file pop-up window, type letters delimited with slashes to denote nested directories:



- Type line number after a file name, delimited with a colon, to navigate to the specified line:



- Use * wildcard to represent any number of characters.
- If while typing in one of the **Navigate to Class/File/Symbol** pop-up windows you notice that you need another one, just invoke the necessary dialog box. The text you have already entered will not disappear.
- Press **Alt+F1** to invoke the **Select Target** pop-up window, and choose the desired IDE component.
- Note that for the projects under version control, the entries in the look-up list are color coded according to their status:



- When there is a **detached editor frame** with a certain file, you can opt to open this file in the main PhpStorm frame by pressing **Enter**, or activate the detached frame by pressing **Shift+Enter**.

See Also

Procedures:

- [Opening and Reopening Files in the Editor](#)
- [Navigating Between IDE Components](#)

Reference:

- [Appearance](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Navigating to Declaration or Type Declaration of a Symbol

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

While editing your source code, you might need to navigate to the location where a particular named code reference (a symbol) has been first declared. **Go To | Declaration** command enables you to navigate back to the initial declaration of a symbol from any place in the source code, even if it is from inside another class, or comment.

Note

Navigate | Declaration/Type Declaration

- applies to the symbols of source code, CSS, HTML or XML tags and attributes, DTD and schema elements and attributes, and references in comments.
- does not apply to the primitive types.

To navigate to the declaration of a symbol

1. Place the caret at the desired symbol in the editor.
2. Do one of the following:
 - o On the main menu, choose **Navigate | Declaration**.
 - o Press **Ctrl+Command B**.
 - o Click the middle mouse button.
 - o Keeping **CtrlCtrl** pressed, point to the symbol, and click, when it turns to a hyperlink. You can also see declaration at the tooltip while keeping **CtrlCtrl** pressed.

```
var vehicle = new Vehicle('lamborghini');
vehicle.c var vehicle = new Vehicle('lamborghini')
alert(vehicle.getVclInfo());
```

To navigate to the type declaration of a symbol

1. Place the caret at the desired symbol in the editor.
2. Do one of the following:
 - o On the main menu, choose **Navigate | Type Declaration**.
 - o Press **Ctrl+Shift+Command Shift B**.
 - o Keep depressed the **Ctrl+ShiftCtrl Shift** keys, and hover your mouse pointer over the symbol. When the symbol turns to a hyperlink, clicking the mouse button to open type declaration in the editor. You can also see the declaration at the tooltip while keeping **Ctrl+ShiftCtrl Shift** pressed.

```
function Vehicle(type)
var vehicle = new Vehicle('lamborghini');
```

See Also

Concepts:

- [Markup Languages and Style Sheets](#)

Procedures:

- [Viewing Images](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

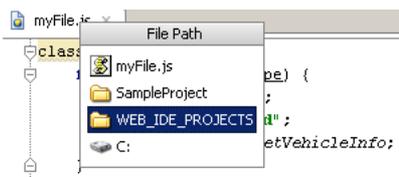
Navigating to File Path

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

It may be helpful to open a file in a file manager (for example, in the Windows Explorer, if Windows is your OS). PhpStorm allows you to easily navigate to any part of a file path, right from the editor.

To navigate to a file path from the editor

1. Hold **CtrlCtrl** and click the relevant tab in the editor.
2. In the drop-down list, select the element you need. For example, if you select **WEB_IDE_Projects** as shown below, the file manager opens for **C:\web_ide_projects**, with the **WEB_IDE_Projects** directory selected:



Note

To just view the path to a file, place the mouse pointer over the tab where the file is opened:



See Also

Reference:

- [Editor](#)

Getting Started:

- [Navigating Through the Source Code](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>
-

Navigating to Implemented/Overridden or Implementing/Overriding Methods

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

PhpStorm provides an easy way to navigate up and down through the hierarchy of methods. If a method is overridden / implemented by a certain method, or overrides / implements some method itself, it is marked with an icon in the gutter area of the editor. When the mouse cursor hovers over such icon, the method information is displayed as the tooltip:

- : This method implements a method required by an implemented interface or extended abstract class.
- : This method of an interface or an abstract class is implemented by one or more descendents.
- : This method overrides a method defined by its superclass.
- : This method is overridden by one or more subclasses.

You can use these icons, keyboard and mouse shortcuts, or menu commands to navigate to the corresponding points of the origin.

To navigate up and down through the method hierarchy

Do one of the following:

- Click the gutter icon and select the desired ascendant or descendant class from the list.
- On the main **Navigate** menu, choose **Super Method**, or **Implementation(s)** respectively.
- Use keyboard shortcuts `Ctrl+UCommand U` or `Ctrl+Alt+BCommand Alt B` for the super method and implementation respectively.

See Also

Reference:

- [General](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>
-

Navigating to Line

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Navigate | Line command is the most basic way to navigate to the line with the specified number.

To navigate to a line in the editor

1. On the main menu, choose **Navigate | Line**, or press `Ctrl+GCommand G`.
2. In the **Go To Line** dialog box, type the target line number and click **OK**.

Note

The line number is displayed in the status bar at the lower edge of the PhpStorm main window.

See Also

Reference:

- [Editor](#)

Getting Started:

- [Navigating Through the Source Code](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>
-

Navigating to Next/Previous Change

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

If you edit a file that is under version control, PhpStorm provides several ways to move back and forth with the updates. In particular, you can use the navigation commands, keyboard shortcuts, and the change markers.

To navigate to the next/previous change in the editor

Do one of the following:

- On the main menu, choose **Navigate | Next / Previous Change**.
- Use keyboard shortcuts `Shift+Ctrl+Alt+DownShift` `Command Alt Down` Or `Shift+Ctrl+Alt+UpShift` `Command Alt Up`.
- Point to a [change marker](#), and click the arrow up  or arrow down  buttons.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Using Change Markers to View and Navigate Through Changes in the Editor](#)
- [Viewing Recent Changes](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Navigating to Next/Previous Error

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts:  

Another method of code navigation is to [move between found errors and warnings](#). The caret is positioned immediately before the error or warning. If no errors and warnings are found in the file, PhpStorm displays the corresponding message in a pop-up window.

You can [configure to skip warnings](#) and navigate only between detected errors.

To configure the error navigation

1. Right click the [Validation Side Bar](#).
2. On the context menu, specify the navigation mode:
 - To have PhpStorm skip warnings, choose **Go to errors only**.
 - To have PhpStorm jump both to errors and warnings, choose **Go to next error/warning**.

To navigate between errors or warnings, do one of the following

- On the main menu, choose **Navigate | Next / Previous Highlighted Error**.
- Use keyboard shortcuts `F2F2` and `Shift+F2Shift` `F2` respectively.

See Also

Reference:

- [Editor](#)

Getting Started:

- [Navigating Through the Source Code](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Navigating to Recent File

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts:  

In this section:

- [Navigating to a recently opened file](#)
- [Navigating to a recently edited file](#)
- [Navigating to the latest edit location](#)

To navigate to a recently opened file

1. On the main menu, choose **View | Recent Files** or press `Ctrl+ECommand E`.
2. From the **Recent Files** pop-up window that opens select the desired file.

To navigate to a recently edited file

1. On the main menu, choose **View | Recently Changed Files** or press `Ctrl+Shift+ECommand Shift E`.

2. From the **Recently Edited Files** pop-up window that opens select the desired file.

Tip

- The recently opened or recently modified files are selected from from the history list. The number of entries in the history list is configurable in the **Recent file limit** text box in the [Editor](#) dialog box.
- Navigating to recent files applies to the search results as well. By pressing `Ctrl+Command E` in the [Find](#) tool window, you can have the list of recent search results shown.

To jump to the latest edit location

- Do one of the following:
 - On the main menu, choose **Navigate | Last Edit Location**
 - Press `Ctrl+Shift+BackSpace``Command Shift BackSpace`.

See Also

Procedures:

- [Viewing Recent Changes](#)

Reference:

- [Editor](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Navigating to Navigated Items

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac 

You can go back and forth along the items that have already been navigated to. This feature applies to the items reached using all navigation commands except for the simplest ones, such as arrow keys, Pageup, Page Down, Home and End.

To navigate to the navigated items

Do one of the following:

- On the main menu, choose **Navigate | Back / Forward**.
- Use keyboard shortcuts `Ctrl+Alt+LeftCommand Alt Left`, or `Ctrl+Alt+RightCommand Alt Right`.
- On the main toolbar, click  or .

Note

On a Mac OS X computer, you can also use the three-finger right-to-left and left-to-right swipe gestures.

See Also

Reference:

- [Navigation Between IDE Components](#)

Getting Started:

- [Navigating Through the Source Code](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

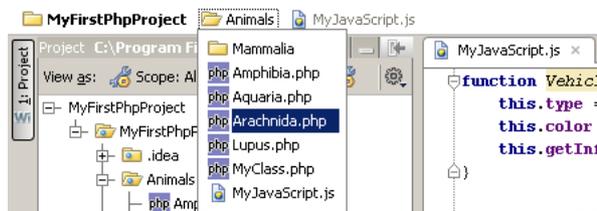
Navigating with Navigation Bar

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac 

Use the Navigation bar as a handy tool to find your way across the project.

To navigate to a file using the navigation bar

1. Press `Alt+HomeAlt Home` to activate the Navigation bar.
2. Use the arrow keys or the mouse pointer to locate the desired file.
3. Double-click the selected file, or press `EnterEnter` to open it in the editor.



See Also

Reference:

- [Navigation Between IDE Components](#)

Getting Started:

- [Navigating Through the Source Code](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

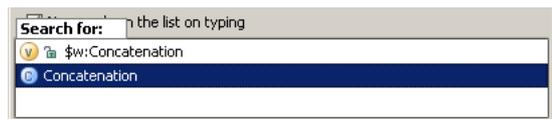
Navigating with Structure Views

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Use the **Structure** pop-up window or the **Structure** tool window to quickly jump to the desired member of a file in the editor. The Structure views provide quick navigation for all supported file types

To navigate to a member in the editor

1. Choose **View | File Structure Pop-up** on the main menu or press **Ctrl+F12Command F12**.
2. In the **File Structure** pop-up window that opens select the **Narrow down the list on typing** check box and start typing the desired member name.
You can include the members, inherited from the parent classes, by checking the option **Show inherited members** or pressing **Ctrl+F12Command F12** again.



3. Use the navigation keys to select the desired node. Then do one of the following:
 - o If the cursor rests on a top or intermediate node (for example, class or element) , double-click this node or press **EnterEnter** to expand it in the Structure pop-up, or press **F4F4** to jump to its declaration in the editor.
 - o If the cursor rests on a leaf node (for example, a member ,) or lowest-level element) , double-click this node or press **EnterEnter** to jump to its declaration in the editor.

In the case of an inherited member, the respective parent class opens in the editor.

Note

You can also use the **Structure** tool window (**Alt+7Meta 7**). This view is flexibly configurable and useful for many tasks, apart from navigation. However, the **File Structure** pop-up window is the easiest way for quick navigation.

See Also

Procedures:

- [Viewing Structure of a Source File](#)

Reference:

- [Structure Tool Window](#)
- [Project Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

```
ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js'; var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s); }());
```



PhpStorm 3.0.0 Web Help

Searching Through the Source Code

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm provides extensive search and replace capabilities, which includes basic search and replace, search and replace in paths, finding usages, code-aware structural search and more. Some of the replacement facilities (like renaming classes and members) are performed by means of refactoring.

All the search commands can be found under the Find node of the Edit menu:



Click thumbnail to view larger image.

In this part you will learn how to:

- [Find and replace](#) text in the current file.
- [Find specific word](#).
- [Search and replace across the project](#).
- [Find usages](#).
- [Highlight usages](#).
- [View usages of a symbol across the project, and jump to the desired usage](#).
- [Work with the search results](#).

See Also

Getting Started:

- [Speed Search in the Tool Windows](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Finding and Replacing Text in File

Previous | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

The standard facility helps you find and replace text strings in the active editor.

In this section:

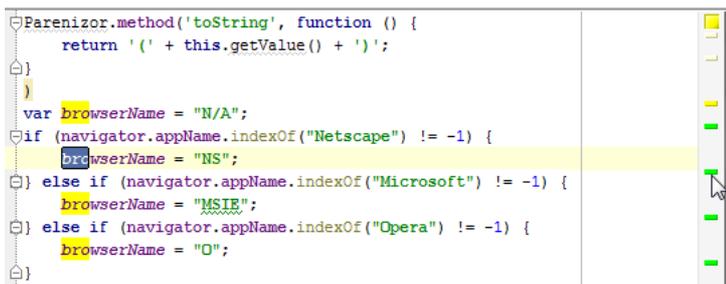
- [Search in the current file](#)
- [Replace in the current file](#)
- [Working with search results](#)
- [Search and replace options](#)

To find a text string in the current file

1. On the main menu, choose Edit | Find | Find, or press **Ctrl+F** Command F or **Alt+F3** Alt F3. The search pane appears on top of the active editor.
2. If necessary, specify the [search options](#).
3. In the search field, start typing the search string:



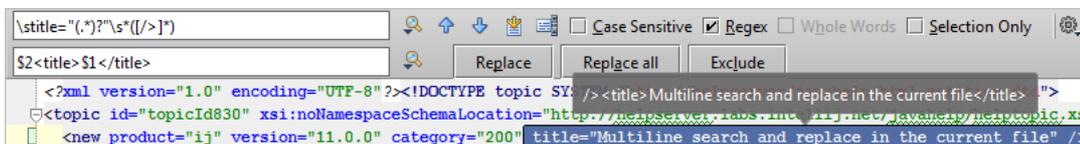
As you type, the first occurrence of the search string after the current cursor position is selected; the other occurrences are highlighted in the editor. In addition, the matching occurrences are marked in the right gutter with stripes.



4. [Explore search results](#).

To replace a text string in the current file

1. On the main menu, choose Edit | Find | Replace, or press **Ctrl+R** Command R. The search and replace pane appears on top of the active editor.
2. If necessary, specify the [search and replace options](#).
3. In the search field, start typing the search string. As you type, the matching occurrences are highlighted in the editor, and a [replace dialog box](#) pops up automatically at the first occurrence, suggesting you to replace the current occurrence, or all of them, with an empty string.
4. Start typing the replacement string. It is immediately reflected in the replace dialog box. Note that for the regular expressions replacement preview is shown at the tooltip.



5. [Explore search results](#), and, using the buttons of the [replace dialog box](#), replace occurrences as required.

Working with search results

- To initiate a new search, do one of the following (depending on the current focus):
 - If the editor has the focus, press **Ctrl+F** **Command F** or **Alt+F3** **Alt F3**.
 - If the search field has the focus, press **Ctrl+A****Command A**

In both cases, the existing search string will be selected, and can start typing a new one.

- To jump between occurrences, do one of the following:
 - Press **Shift+F3** **Shift F3** or **Ctrl+Shift+L** **Command Shift L** or **F3** **F3** or **Ctrl+L** **Command L**.
 - Use **↑** or **↓** buttons in the Search pane.
 - Click the gutter stripes.
- The search pane shows the number of found occurrences. If no matches are found, the search pane is red highlighted:



- Use Code completion in Find and Replace panes. Start typing the search string, press **Ctrl+Space****Command Space**, and select the appropriate word from the suggestion list.
- Use the recent history of searches: with the search pane already open, click **🔍** to show the list of recent entries.
- With the search or replace pane already opened, use **Ctrl+R****Command R** or **Ctrl+F** **Command F** or **Alt+F3** **Alt F3** to toggle between panes. So doing, the search and replace strings are preserved.
- To cancel operation and close the pane, press **Escape****Escape**.

Search and replace options

Item	Description	Search/Replac
	Click this button to show the history of the recent entries.	Search, replace
	Click these buttons to navigate through the occurrences of the search string.	Search, replace
	Click this button to show search results in the Find tool window .	Search, replace
	Click this button to enable entering the search string in several lines. If this button is not pressed, you still can enter search strings that occupy several lines, but should work in the regular expression mode and use escape characters. For example: 	Search, replace
Case sensitive	If this check box is selected, PhpStorm will distinguish between upper and lowercase letters while searching.	Search, replace
Match whole words only	If this check box is selected, PhpStorm will search for whole words only, that is, for character strings separated with spaces, tabs, punctuation, or special characters. This check box is disabled, if the Regular expressions check box is selected.	Search, replace
Regex	If this check box is selected, the search string will be perceived as a regular expression . Note If this check box is selected, PhpStorm shows replacement preview in a tooltip, as shown in the image above .	Search, replace
Preserve case	If this check box is selected, PhpStorm retains the case of the first letter and the case of the initial string in general. For example, <i>MyTest</i> will be replaced with <i>Yourtest</i> if you specify <i>yourtest</i> as the replacement. This check box is disabled, if Regular expressions check box is selected.	Replace
Selection only	If this check box is selected, search and replacement will be confined to the selected text only.	Replace
Exclude/Include	Click Exclude button to skip the current occurrence and exclude it from the Replace all operation. The button for this occurrence changes to Include .	Replace
Replace	Click this button to replace the current occurrence and proceed to the next one.	Replace
Replace all	Click this button the replace all found occurrences in the current file, or in the selection.	Replace

See Also

Reference:

- [Regular Expression Syntax Reference](#)

Getting Started:

- [Familiarize Yourself with PhpStorm Editor](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Finding and Replacing Text in Project

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

PhpStorm extends search and replace capability to the entire project, or any directory with its nested hierarchy. Explore search results in the [Find tool window](#).

To find a text in all files within the specified path

1. On the main menu, choose **Edit | Find | Find in Path**, or press **Ctrl+Shift+F** (Command Shift F).
2. In the [Find In Path Dialog](#), specify the following options:
 - Text to find (you can select one from the recent history drop-down list).
 - Search scope (project, module or directory).
 - Search options (case sensitivity, whole words, and regular expressions).
3. Click **Find**.

Tip

If the search takes too long time, click **Background** in the search progress window. In this case the search progress is indicated in the Status bar.

To replace a text in all files within the specified path

1. On the main menu, choose **Edit | Find | Replace in Path**, or press **Ctrl+Shift+R** (Command Shift R).
2. In the [Replace In Path dialog](#), specify the search and replace strings, search options and scope, and click **Find**. Encountered occurrences of the search string are displayed in the [Find tool window](#).
3. In the Find tool window, select one or more occurrences you want to replace, and click **Replace selected** button. If you are ready to replace all occurrences, click **Do Replace All**.

See Also

Procedures:

- [Working with Search Results](#)

Reference:

- [Find and Replace in Path](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Finding Usages

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Search for usages is an important part of the code analysis, which enables you to clarify dependencies. PhpStorm suggests several types of search for usages:

- [Finding Usages in Project](#)
- [Finding Usages in the Current File](#)
- [Highlighting Usages](#)
- [Viewing Usages of a Symbol](#)
- [Viewing Recent Find Usages](#)

See Also

Concepts:

- [Code Inspection](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

```
ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js'; var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s); })();
```



PhpStorm 3.0.0 Web Help

Finding Usages in Project

Previous | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: 

PhpStorm provides different search options depending whether you are searching for usages of a class, method, field, parameter, or throw statements, and extends search for usages to the files in supported languages. For example, in CSS, XML and HTML files you can search for the usages of styles, classes, tags and attributes.

Explore search results in the [Find tool window](#).

To find usages of a symbol in a project

1. Select a symbol to find usages for. To do that, place the caret within the desired symbol in the editor, or click the symbol in the Project tool window.
2. Do one of the following:
 - o On the main menu, choose **Edit | Find | Find Usages**
 - o Choose **Find Usages** on the context menu
 - o Press **Alt+F7** **Alt F7**.
3. In the Find Usages dialog box, specify the search options, which depend on the type of the symbol you are searching for. Refer to the Find Usages dialogs for the detailed description of options.
4. Specify how PhpStorm should behave if only one occurrence is found. If the option **Skip results tab with one usage** is checked, PhpStorm will scroll to the only encountered usage, rather than display it in the Find tool window.
5. Specify the scope of search: select one of the pre-defined scopes from the drop-down list, or click the ellipsis button to define your custom scope, which can include external libraries and project files.
6. Click **Find** button.
7. In the [Find tool window](#), explore search results. Use the  button to represent search results in meaningful groups by type of usage.

See Also

Concepts:

- [Scope](#)

Reference:

- [Find Usages. Class Options](#)
- [Find Usages. Package Options](#)
- [Find Usages. Variable Options](#)
- [Find Usages. Method Options](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Finding Usages in the Current File

Previous | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: 

To find usages of a symbol in the current file

1. Click the desired symbol in the editor, or in the Structure view.
2. On the main menu, choose **Edit | Find | Find Usages in File**, or press **Ctrl+F7** **Command F7**. The encountered usage is highlighted in the editor.

See Also

Procedures:

- [Highlighting Usages](#)

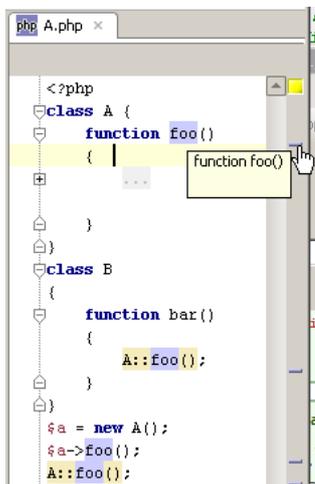
Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Highlighting Usages

Previous | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: 

The search command **Highlight Usages in File** (**Ctrl+Shift+F7** **Command Shift F7**) makes it possible to visualize usages of a symbol in the current file. All found usages of a symbol in the current file are highlighted and color-coded, as defined in the [Colors and Fonts](#) dialog box, to represent read or write access to the symbol. In addition to the highlights of occurrences in text, the stripes of the same colors appear in the marker bar, accompanied with tooltips.



The behavior of usage highlighting is configurable: you can make PhpStorm show usages of a symbol at caret automatically, or invoke it with a command.

This section describes how to:

- [configure highlight usages behavior](#)
- [highlight usages](#)
- [navigate among usages](#)
- [remove highlighting](#)

To configure highlight usages behavior

1. [Open Settings dialog](#).
2. In the [Editor](#) page of the IDE Settings, select the check box **Highlight usages of element at caret** to enable usage highlighting.

To highlight usages of a symbol in the current file

1. Place the caret at the selected symbol in the editor. If automatic usages highlighting is enabled, see all its occurrences in the current file highlighted. Otherwise, proceed to the next step.
2. On the main menu, choose **Edit | Find | Highlight Usages in File**, or press **Ctrl+Shift+F7** / **Command Shift F7**.

To navigate among usages, do one of the following

- Click on a stripe in the marker bar to navigate to the respective usage location.
- Use **F3** / **F3** or **Ctrl+L** / **Command L** and **Shift+F3** / **Shift F3** or **Ctrl+Shift+L** / **Command Shift L** keyboard shortcuts, to navigate to the next and previous usages respectively.

To remove highlighting of usages

- Press **Escape** / **Escape**.

See Also

Reference:

- [Editor. Colors and Fonts](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Viewing Usages of a Symbol

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Using the **Show Usages** function, you can bring up a list of the usages of a symbol across the whole project. So doing, the pop-up window with the list of usages of a symbol features a toolbar with the following buttons:

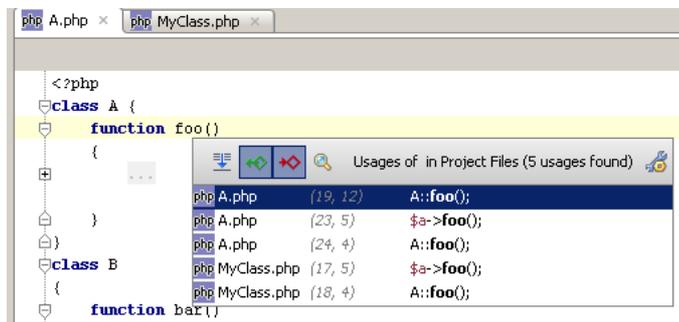
Icon	Shortcut	Description
	Ctrl+F / Command F or Alt+F3 / Alt F3	Merge usages of the symbol from the same line.
	Ctrl+R / Command R	Show read access to the symbol.
	Ctrl+W / Command W	Show write access to the symbol.
	Ctrl+I / Command I	Show usages in the import statements.
	Alt+F7 / Alt F7	Show search results in the Find Usages tool window.
	Ctrl+Shift+Alt+F7 / Command Shift Alt F7	Open the Find Usages dialog box for the selected symbol, enabling you to change search options.

Tip

In addition to the possibility of viewing usages, you can use this function as a quick means of navigation.

To view the usages of a symbol across the project

1. Place the caret at the desired symbol in the editor.
2. On the main menu, choose Edit | Find | Show Usages, or press Ctrl+Alt+F7/Command Alt F7.
3. Use the toolbar buttons to present search results in the desired way.
To jump from search results to a line of source code, click the desired entry.
To close the list, press Escape/Escape.



See Also

Procedures:

- [Finding Usages in Project](#)
- [Highlighting Usages](#)

Reference:

- [Find Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Viewing Recent Find Usages

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

With PhpStorm you can easily navigate to the recent search results if any find usages action took place during the PhpStorm session.

To view recent find usages

1. Select Edit | Find | Recent Find Usages on the main menu.
2. Select the needed search item from the pop-up list of the recent search results:

See Also

Reference:

- [Find Tool Window](#)
- [Ctrl](#)
- [Search](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Finding Word at Caret

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Search for a word at caret enables you to quickly find the exact match for the current word, without changing any search options. With the match once found, you can navigate between the occurrences of the term.

To find the word at caret, do one of the following

- On the main menu, choose Edit | Find | Find Word At Caret.
- Use Ctrl+F3/Command F3 keyboard shortcut.

To navigate between the occurrences of the word at caret

1. Press F3 / F3 or Ctrl+L / Command L to go to the next occurrence.
2. Press Shift+F3 / Shift F3 or Ctrl+Shift+L / Command Shift L to go to the previous occurrence.

See Also

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>
-

Working with Search Results

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

The search results display in the [Find tool window](#). The results of each search display in a separate tab.

Using the controls and the context menu of this tool window as well as the main menu, you can:

- Navigate [to source code](#).
- [Exclude and include](#) search results in refactoring.
- Add selected search results [to favorites](#).
- Show the results of the [recent find usages](#).
- View [local history](#) of the selected usage.
- Perform [version control](#) operations using the specific VCS, associated with the parent directory of the usage.

See Also

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>
-

Refactoring Source Code

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

PhpStorm offers a wide variety of code refactorings, which track down and correct the affected code references automatically.

In this part you will find:

- [General refactoring procedure](#)
- [Procedures and examples of the available refactorings](#)

To perform refactoring, follow these general steps

1. Select a symbol or code fragment to refactor. The set of available refactorings depends on your selection. You can select symbols in the following PhpStorm components:
 - Project view
 - Structure tool window
 - Editor
2. On the main **Refactor** menu or on the context menu of the selection, choose the desired refactoring or press the corresponding keyboard shortcut (if any). In the dialog box that opens, specify the refactoring options.
3. To apply the changes immediately, depending on the refactoring type, click **Refactor** or **OK**.
4. For certain refactorings, there is an option of previewing the changes prior to actually performing the refactoring. In such cases the **Preview** button is available in the corresponding dialog.

To preview the potential changes and make the necessary adjustments, click **Preview**. PhpStorm displays the changes that are going to be made on a dedicated tab of the [Find tool window](#).

One of the possible actions at this step is to exclude certain entries from the refactoring. To do that, select the desired entry in the list and press **DeleteDelete**.

Note

If conflicts are expected after the refactoring, PhpStorm displays a dialog with a brief description of the encountered problems. If this is the case, do one of the following:

- Ignore the conflicts by clicking the **Continue** button. As a result, the refactoring will be performed, however, this may lead to erroneous results.
- Preview the conflicts by clicking the **Show in View** button. PhpStorm shows all conflicting entries on the **Conflicts** tab in the [Find tool window](#), enabling you to navigate to the problematic lines of code and to make the necessary fixes.
- Cancel the refactoring and return to the editor.

5. When you are satisfied with the proposed results, click **Do Refactor** to apply the changes.

PhpStorm provides the following common refactorings:

- [Copy/Clone](#)
- [Introduce Constant](#)
- [Introduce Field](#)
- [Extract Method](#)
- [Inline](#)
- [Introduce Variable](#)
- [Move Refactorings](#)
- [Rename Refactorings](#)
- [Safe Delete](#)
- [Introduce Parameter](#)

The following language-specific refactorings are available:

- [Introduce Parameter in JavaScript](#)

- [Change Signature in JavaScript](#)
- [Introduce Variable in JavaScript](#)

See Also

Procedures:

- [Working with Search Results](#)

Reference:

- [Refactoring Dialogs](#)

External Links:

- <http://www.refactoring.com/>

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Copy/Clone

Previous | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

The Copy refactoring allows to [copy](#) a class, file, or directory with its entire structure from one directory to another, or [clone](#) it within the same directory.

To copy a class, file, or directory

1. Select the desired item in one of the views or open it in the editor.
2. Do one of the following:
 - o On the main menu or on the context menu of the selection, choose **Refactor | Copy**.
 - o Press **F5F5**.
 - o Select the desired class in the **Project** tool window and drag it to the target destination with the **CtrlCtrl** key pressed.
3. In the **Copy** dialog box that opens specify the new name and destination, then click **OK**.

To clone a class or file

1. Select the desired item in one of the views or open it in the editor.
2. Press **Shift+F5Shift F5**.
3. In the **Clone** dialog box that opens specify the new name, then click **OK**.

See Also

Reference:

- [Copy Dialogs](#)

Getting Started:

- [Copy and Paste Between PhpStorm and Explorer/Finder](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Introduce Constant

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

The Introduce Constant refactoring makes your source code easier to read and maintain. It also helps you avoid using hard coded constants without any explanations about their values or purpose.

- [Examples](#)
- [To introduce a constant](#)

Examples

- [JavaScript Example](#)
- [PHP Example](#)

JavaScript example

Before

```

    Parenizor.method('toString', function () {
return '(' + this.getValue() + ')';
}
)

```

After

```

    Parenizor.method('toString', function () {
const string = '(' + this.getValue() + ')';
return string;
}
)

```

PHP example

Before

```
public static function find($params){
    if (isset($params['param_query'])) {
        $result = MyDatabase::execute($params['param_query']);
    }
}

public static function findAll($params){
    if (isset($params['param_query'])) {
        $result = MyDatabase::executeAll($params['param_query']);
    }
}
```

After

```
const
    PARAM_QUERY = 'param_query';

public static function find($params){
    if (isset($params[self::PARAM_QUERY])) {
        $result = MyDatabase::execute($params[self::PARAM_QUERY]);
    }
}

public static function findAll($params){
    if(isset($params[self::PARAM_QUERY])){
        $result = MyDatabase::executeAll($params[self::PARAM_QUERY]);
    }
}
```

To introduce a constant

1. In the editor, select the expression to be replaced with a constant.
2. Choose **Refactor | Introduce Constant** on the main menu or on the context menu of the selection. Alternatively press **Ctrl+Alt+C** Command **Alt C**.
3. In the **Introduce Constant** dialog box, type the name of the new constant in the Name text box.
4. Specify the scope to apply refactoring in:
 - o To have only the selected expression replaced, clear the **Replace all occurrences** check box.
 - o To have PhpStorm replace the selected expression wherever it is used, select the **Replace all occurrences** check box.
5. Click **OK** to start the refactoring.

See Also

Procedures:

- [Refactoring Source Code](#)
- [Introduce Variable](#)

Reference:

- [Introduce Constant Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Introduce Field

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

The **Introduce Field** refactoring declares a new field and initializes it with the selected expression. The original expression is replaced with the usage of the field.

- [Example](#)
- [Introducing a field using the Introduce Field dialog](#)

Example

Before

```
public static function find($params){
    if (isset($params['param_query'])) {
        $result = MyDatabase::execute($params['param_query']);
    }
}

public static function findAll($params){
    if (isset($params['param_query'])) {
        $result = MyDatabase::executeAll($params['param_query']);
    }
}
```

After

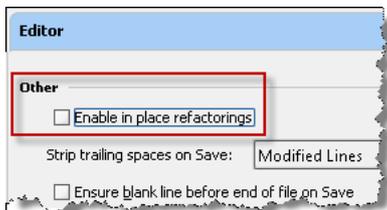
```
public static
    $query = 'param_query';

public static function find($params){
    if (isset($params[self::$query])) {
        $result = MyDatabase::execute($params[self::$query]);
    }
}

public static function findAll($params){
    if(isset($params[self::$query])){
        $result = MyDatabase::executeAll($params[self::$query]);
    }
}
```

To introduce a field using the Introduce Field dialog

If the **Enable in place refactorings** check box is cleared on the Editor settings, the **Introduce Field** refactoring is performed by means of the **Introduce Field** dialog box.



1. In the editor, select the expression or variable to be replaced with a field, or just place the cursor within such an expression or variable declaration.
2. In the main menu, or the context menu of the selection, choose **Refactor | Introduce Field**, or press **Ctrl+Alt+F** (Windows) or **Command+Alt+F** (Mac).
3. In the Expressions pop-up menu, select the expression to be replaced. Note that PhpStorm highlights the selected expression in the editor.
4. In the **Introduce Field** dialog that opens, specify the type and name of the new field.
5. To automatically replace all occurrences of the selected expression (if it is found more than once), select the option **Replace all occurrences**.
6. Click **OK** to create the field.

See Also

Reference:

- [Introduce Field Dialog](#)

Web Resources:

- <http://www.jetbrains.com/idea/faq/faq-introduce-field.html>
- <http://youtrack.jetbrains.com/issue/WI-10000>

Extract Method

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

When the **Extract Method** refactoring is invoked, PhpStorm analyses the selected block of code and detects variables that are the input for the selected code fragment and the variable that is the output for it.

Note

In the JavaScript context, this refactoring always results in a function.

In the PHP context, the result of applying the **Extract Method** refactoring depends on the location of the selected code fragment.

- If the selection is made inside a method of a class, the refactoring extracts a method. This case is applicable when you are using PHP 5.0 and higher.
- If the selection is made inside a function or a script, the refactoring extracts a function.

- [JavaScript Example](#)
- [PHP Extract Method Example](#)
- [PHP Extract Function Example](#)
- [Extracting a JavaScript function](#)
- [Extracting a PHP function or method](#)

JavaScript example

Before

```
function multiplication(a,b) {
    c = a + b;
    d = c * c;
    return d;
}
```

After

```
function sum(a,b);
    return a + b;

function multiplication(a,b) {
    c = sum(a,b);
    d = c * c;
    return d;
}
```

PHP extract method example

Before

```
public function init()
{
    $this->_router = $this->getFrontController()->getRouter();
}
```

After

```
public function init()
{
    $this->_router = $this->getRouter();
}

public function getRouter()
{
    return $this->getFrontController()->getRouter();
}
```

PHP extract function example

Before

```
<?php
    if ( 'POST' != $_SERVER['REQUEST_METHOD'] )
    {
        header('Allow: POST');
        header('HTTP/1.1 405 Method Not Allowed');
        header('Content-Type: text/plain');
        exit;
    }
?>
```

After

```
<?php
    function printEmptyHeader()
    {
        header('Allow: POST');
        header('HTTP/1.1 405 Method Not Allowed');
        header('Content-Type: text/plain');
    }

    if ( 'POST' != $_SERVER['REQUEST_METHOD'] )
    {
        printEmptyHeader();
        exit;
    }
?>
```

To extract a function or method in the php context

1. In the editor, select a block of code to be transformed into a function or method.

Tip

The selected code fragment does not necessarily have to be a set of statements. It may also be an expression used somewhere in the code.

2. On the main menu or on the context menu of the selection, choose **Refactor | Extract Method** or press **Ctrl+Alt+M** / **Command Alt M**.

The title of the [dialog box that opens](#) is either **Extract Method** or **Extract Function** depending on the [intended refactoring output](#).

3. Specify the name of the new function or method.
4. If a method is to be extracted, choose the relevant [access modifier](#) for it in the **Visibility** area:
 - o **Public**
 - o **Protected**
 - o **Private**
5. Configure the output of the new method or function in the **Return output variable(s) through** area.

The **Output variable(s)** read-only field displays all the variables that make up the output of the selected code fragment. Specify the way in which the new method or function will [return these variables](#) to the callee.

- o To have the output variables returned by value, select the **Return statement option**. The result of this choice depends on the number of detected output variables. If there is exactly one output variable, it will be used as the return value. If the selection outputs several variables, these variables will be returned as an array.
- o To have the output variables returned by reference, select the **Parameter(s) passed by reference** option. In this case, no return statement will be generated. Instead, all the detected output variables will be added to the list of [input parameters](#). Their names will be prepended with an ampersand & both in the **Parameters** list and in the declaration of the new method/function, as shown in the **Signature preview** read-only area.

6. In the **Parameters** area, configure the input for the new function or method.

Originally, the list contains the variables that PhpStorm has detected as the input for the new method or function. However, if you have chosen to [return the new method/function output by reference](#), the list also contains all the output variables.

To change the order of parameters, use the **Move Up** and **Move Down** buttons. This order determines the order in which the parameters are listed in the declaration of the new method or function.

7. Optionally, to have PhpStorm transform tail [break](#) or [continue](#) statements if the selection contains any, select the **Replace tail "break/continue" statements with return statement** check box.
8. View and check the declaration of the function or method to be generated in the **Signature preview** read-only area.

See Also

Reference:

- [Extract Method Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Inline

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

PhpStorm provides the following inline refactorings:

- The [Inline Variable](#) refactoring replaces redundant variable usage with its initializer. This refactoring is opposite to [Introduce variable](#).

Tip

The variable must be initialized at declaration. If the initial value is modified somewhere in the code, only the occurrences before modification will be inlined.

- The [Inline Function/Method](#) refactoring results in placing the method's or function's body into the body of its caller(s); the method or function is deleted. This refactoring is opposite [Extract](#)

[Method](#).

Inline variable

- [JavaScript Example](#)
- [PHP Example](#)

JavaScript example

Before

```

Parenizor.method('toString', function ()
{
    var string = '(' + this.getValue() + ')';
    return string;
}
    
```

After

```

Parenizor.method('toString', function ()
{
    return '(' + this.getValue() + ')';
}
    
```

PHP example

Before

```

public function getFeedObject($title, $description)
{
    global $wgSitename, $wgContLanguageCode, $wgFeedClasses, $wgTitle;
    $feedTitle = "$wgSitename - {$title} [$wgContLanguageCode]";
    if (!isset($wgFeedClasses[$this->format]))
        return false;
    return new $wgFeedClasses[$this->format]
($feedTitle, htmlspecialchars());
}
    
```

After

```

public function getFeedObject($title, $description)
{
    global $wgSitename, $wgContLanguageCode, $wgFeedClasses, $wgTitle;
    if (!isset($wgFeedClasses[$this->format]))
        return false;
    return new $wgFeedClasses[$this->format]
("$wgSitename - {$title} [$wgContLanguageCode]", htmlspecialchars());
}
    
```

Inline method or function

Before

```

function
sum(a, b){
    return a + b;
}

function multiplication(a, b){
    c = sum(a, b);
    d = c * c;
    return d;
}

function division(a, b){
    result = sum(a, b) / multiplication(a, b);
    return result;
}
    
```

After

```

function
multiplication(a, b){
    c = a + b;
    d = c * c;
    return d;
}

function division(a, b){
    result = a + b / multiplication(a, b);
    return result;
}
    
```

To perform the inline refactoring

1. Place the caret in the editor at the desired symbol to be inlined.
2. Choose **Refactor | Inline** on the main menu or on the context menu of the selection. Alternatively press **Ctrl+Alt+N** or **Command Alt N**.
3. In the **Inline** dialog box that corresponds to the selected symbol, confirm the inline refactoring.

See Also

Procedures:

- [Extract Method](#)
- [Introduce Variable](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Introduce Variable

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

The **Introduce Variable** refactoring puts the result of the selected expression into a variable. It declares a new variable and uses the expression as an initializer. The original expression is replaced with the new variable ([see the examples below](#)).

In JavaScript and ActionScript, you can replace an expression with a variable or a constant. For JavaScript 1.7 or a later version, there is also an option of introducing a local variable. Refer to the section [Introduce Variable in JavaScript](#) for details.

To introduce a new variable, you can use:

- [In-place refactoring](#) (this is the default option for %language%). In this case you specify the name of the new variable right in the editor.
- [The Introduce Variable dialog](#) where you specify all the information necessary for introduction of a new variable.

This dialog is always used when working with JavaScript. To make this dialog accessible for %language%, you have to disable [in-place refactorings in the editor settings](#).

You can select the expression to be replaced with a variable yourself. You can as well use [smart expression selection](#). In this case PhpStorm will help you select the desired expression.

PHP example

Before

```
public function getFeedObject($title, $description)
{
    global $wgSitename, $wgContLanguageCode, $wgFeedClasses, $wgTitle;
    if (!isset($wgFeedClasses[$this->format]))
        return false;
    return new $wgFeedClasses[$this->format]
        ("{$wgSitename} - {$title} [{$wgContLanguageCode}]", htmlspecialchars());
}
```

After

```
public function getFeedObject($title, $description)
{
    global $wgSitename, $wgContLanguageCode, $wgFeedClasses, $wgTitle;
    $feedTitle = "{$wgSitename} - {$title} [{$wgContLanguageCode}];
    if (!isset($wgFeedClasses[$this->format]))
        return false;
    return new $wgFeedClasses[$this->format]
        ($feedTitle, htmlspecialchars());
}
```

To introduce a variable using in-place refactoring

1. In the editor, select the expression to be replaced with a variable. You can do that yourself or use the **smart expression selection** feature to let PhpStorm help you. So, do one of the following:

- Highlight the expression. Then choose **Refactor | Introduce Variable** from the main or the context menu. Alternatively, press **Ctrl+Alt+V** or **Command Alt V**.
- Place the cursor before or within the expression. Choose **Refactor | Introduce Variable** from the main or the context menu, or press **Ctrl+Alt+V** or **Command Alt V**.

In the **Expressions** pop-up menu, select the expression. To do that, click the required expression. Alternatively, use the **Up** and **Down** arrow keys to navigate to the expression of interest, and then press **Enter** to select it.

Note

The **Expressions** pop-up menu contains all the expressions appropriate for the current cursor position in the editor.

When you navigate through the suggested expressions in the pop-up, the code highlighting in the editor changes accordingly.

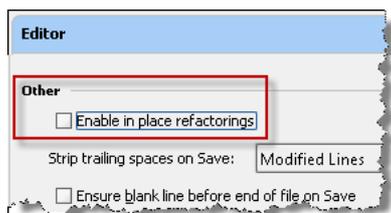
2. If more than one occurrence of the selected expression is found, select **Replace this occurrence only** or **Replace all occurrences** in the **Multiple occurrences found** pop-up menu. To select the required option, just click it. Alternatively, use the **Up** and **Down** arrow keys to navigate to the option of interest, and press **Enter** to select it.

3. Specify the name of the variable. Do one of the following:

- Select one of the suggested names from the pop-up list. To do that, double-click the suitable name. Alternatively, use the **Up** and **Down** arrow keys to navigate to the name of interest, and **Enter** to select it.
- Edit the name by typing. The name is shown in the box with red borders and changes as you type. When finished, press **Escape**.

To introduce a variable using the Introduce Variable dialog

If the **Enable in place refactorings** check box is cleared on the Editor settings, the **Introduce Variable** refactoring is performed by means of the **Introduce Variable** dialog box.



1. In the editor, select the expression to be replaced with a variable. You can do that yourself or use the **smart expression selection** feature to let PhpStorm help you. So, do one of the following:

- Highlight the expression. Then choose **Refactor | Introduce Variable** from the main or the context menu. Alternatively, press **Ctrl+Alt+V** or **Command Alt V**.
- Place the cursor before or within the expression. Choose **Refactor | Introduce Variable** from the main or the context menu, or press **Ctrl+Alt+V** or **Command Alt V**.

In the **Expressions** pop-up menu, select the expression. To do that, click the required expression. Alternatively, use the **Up** and **Down** arrow keys to navigate to the expression of interest, and then press **Enter** to select it.

Note

The **Expressions** pop-up menu contains all the expressions appropriate for the current cursor position in the editor.

When you navigate through the suggested expressions in the pop-up, the code highlighting in the editor changes accordingly.

2. In the **Introduce Variable dialog**:

1. Specify the variable name next to **Name**. You can select one of the suggested names from the list or type the name in the **Name** box.
2. If more than one occurrence of the selected expression is found, you can select to replace all the found occurrences by selecting the corresponding check box. If you want to replace only the current occurrence, clear the **Replace all occurrences** check box.
3. For ActionScript, you can choose to introduce a constant rather than a variable. To do that, select the **Make constant** check box.
4. Click **OK**.

Note

For the JavaScript procedure, refer to the section [Introduce Variable in JavaScript](#).

See Also

Procedures:

- [Refactoring Source Code](#)

Reference:

- [Introduce Variable Dialog](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi> 
- <http://youtrack.jetbrains.com/issues/WI> 

Move Refactorings

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: 

`Move` refactorings allow to move files and directories within a project. So doing, PhpStorm automatically corrects all references to the moved symbols in the source code.

The following `Move` refactorings are available:

- The `Move File` refactoring moves a file to another directory.
- The `Move Directory` refactoring moves a directory to another directory.

To perform a move refactoring, follow these general steps:

1. Select the symbol to be moved and do one of the following:
 - On the `Refactor` menu, or on the context menu, choose `Move`.
 - Press `F6`.
 - In the [Project tool window](#), drag the symbol to the new destination.

The dialog that opens depends on the type of the selected symbol.

2. Specify the move options according to the type of the item to be moved. See option descriptions in the [Move](#) dialog box reference.
3. [Preview and apply the changes](#).

See Also

Reference:

- [Move File Dialog](#)

Getting Started:

- [Copy and Paste Between PhpStorm and Explorer/Finder](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi> 
- <http://youtrack.jetbrains.com/issues/WI> 

Rename Refactorings

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: 

`Rename` refactorings allow to rename symbols, automatically correcting all references in the code.

The following rename refactorings are available in PhpStorm:

- `Rename Class`. The following usages are renamed:
 - Import statements
 - Qualified names of classes
- `Rename Method`. The following usages are renamed:
 - All calls of the method.
 - All overridden/implemented methods in subclasses.
- `Rename Field`.
- `Rename Function`.
- `Rename Variable`.
- `Rename Parameter`. The following usages are renamed:
 - All usages of the parameter.
 - The corresponding `param` tag in documentation comment.
- `Rename CSS color value`.
- `Rename File`.
- `Rename Directory`.

To rename a symbol, follow these general steps

1. Select the item to be renamed.
2. Choose **Refactor** | **Rename** on the main menu or on the context menu of the selection or press `Shift+F6` `Shift F6`.
3. In the **Rename** dialog box that opens, specify additional configuration settings:
 - o To have the changes applied to comments and strings, select the **Search in comments and strings** check box.
 - o When renaming a file, specify whether references should be renamed too.

Note

The set of controls and their names depend on the type of the symbol to be renamed.

4. [Preview and apply changes](#).

See Also

Reference:

- [Rename Dialogs](#)
- [Find Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Safe Delete

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

The **Safe Delete** refactoring allows to safely remove symbols from the source code.

To safely delete a symbol

1. Select the symbol to be deleted.
2. Choose **Refactor** | **Safe Delete** on the main menu or in the context menu of the selection. Alternatively, press `Alt+Delete` `Meta Delete`.

Note

You can also press the `Delete` in the **Project** tool window. Then you can either perform a **Safe Delete** or simply delete the symbol.

3. In the **Safe Delete** dialog box that opens, specify additional settings for the refactoring:
 - o To have the changes applied to comments and strings, select the **Search in comments and strings** check box.
 - o To have the changes applied to text files (such as documentation, HTML, JSP and other files included in your project), select the **Search for text occurrences** check box.
4. Click **OK** to continue. If no usages of the symbol are found, the refactoring removes the symbol. If PhpStorm finds any occurrences of the symbol in the code, the **Usages Detected** dialog box appears.
5. In the **Usages Detected** dialog box, do one of the following:
 - o To proceed with refactoring and delete the symbol, click the **Ignore** button, leaving all the usages of the symbol intact. In this case, to make your code compilable, you have to manually inspect the code and make the necessary corrections.
 - o To view the list of direct usages of the symbol, click the **View usages** button. PhpStorm stops the refactoring procedure and displays a list of direct usages of the selected symbol in the **Find** tool window. Navigate to each usage location and manually correct the code. When ready, click the **Rerun Safe Delete** button to invoke the refactoring again.

See Also

Reference:

- [Safe Delete Dialog](#)
- [Find Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Introduce Parameter

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

- [Introduce Parameter in ActionScript](#)
- [Introduce Parameter in JavaScript](#)

See Also

Code Examples:

- [ActionScript](#)
- [JavaScript](#)

Procedures:

- [Introducing a parameter in ActionScript](#)

- [Introducing a parameter in JavaScript](#)

Reference:

- [Introduce Parameter Dialog for ActionScript](#)
- [Introduce Parameter Dialog for JavaScript](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Introduce Parameter in ActionScript

Previous | [Next](#) | [See Also](#) | [Comments](#)

This section discusses the [Introduce Parameter](#) refactoring in ActionScript.

- [Example](#)
- [Introducing a parameter in ActionScript](#)

Example

Before

```
// Two ways of introducing a parameter for the function
// formatPrice() will be shown.

public function foo():void {
    formatPrice(0);
}

// The new parameter will be optional in the first
// of the examples and required in the second example.

public function formatPrice(value:int):String {
    trace("currency: " + "$");
    return "$" + value;
}
```

After

```
// The function formatPrice() may be called as before because
// the new parameter is introduced as an optional parameter.

public function foo():void {
    formatPrice(0);
}

// The default value for the new parameter is specified
// in the function definition.

public function formatPrice(value:int, s:String = "$"):String {
    trace("currency: " + s);
    return s + value;
}

// The new parameter in this example is a required parameter.
// So two values must be passed to formatPrice() now.

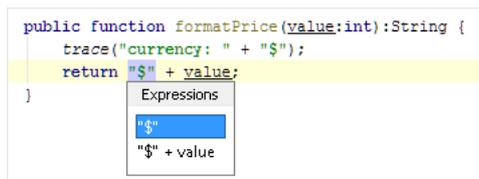
public function foo():void {
    formatPrice(0, "$");
}

// The new parameter is a required parameter because the default
// value for it is not specified in the function definition.

public function formatPrice(value:int, s:String):String {
    trace("currency: " + s);
    return s + value;
}
```

Introducing a parameter in ActionScript

1. In the editor, place the cursor within the expression to be replaced by a parameter.
2. Do one of the following:
 - o Press **Ctrl+Alt+PCCommand Alt P**.
 - o Choose **Refactor | Introduce Parameter** in the main menu.
 - o Select **Refactor | Introduce Parameter** from the context menu.
3. If more than one expression is detected for the current cursor position, the **Expressions** list appears. If this is the case, select the required expression. To do that, click the expression. Alternatively, use the **UpUp** and **DownDown** arrow keys to navigate to the expression of interest, and then press **EnterEnter** to select it.



4. In the [Introduce Parameter dialog](#):
 1. Usually, PhpStorm sets a proper parameter type itself. If necessary, you can select another appropriate type from the **Type** list.
 2. Specify the parameter name in the **Name** field.
 3. The **Value** field, initially, contains the expression that you have selected. Normally, you don't need to change this.

If the new parameter is going to be an optional parameter, the specified value will be used as the default parameter value in the function definition.

If the new parameter is introduced as a required parameter, the specified value will be added to the function calls.

For information about required and optional parameters, see the discussion of function parameters in [Flex/ActionScript documentation](#).

4. If you want the new parameter to be an optional parameter, select the **Optional parameter** check box.
5. If more than one occurrence of the expression is found within the function body, you can choose to replace only the selected occurrence or all the found occurrences with the references to the new parameter. Use the **Replace all occurrences** check box to specify your intention.
6. Click OK.

```
public function formatPrice(value:int, g:String = "$"):String {
    trace("currency: " + g);
    return g + value;
}
```

See Also

Reference:

- [Introduce Parameter Dialog for ActionScript](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Introduce Parameter in JavaScript

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

The `Introduce Parameter` refactoring is used to add a new parameter to a method declaration and to update the method calls accordingly.

- [Examples](#)
- [Introducing a parameter in JavaScript](#)

Examples

The following table shows two different ways of introducing a function parameter.

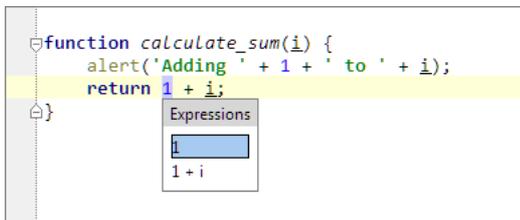
In the first of the examples a new parameter is introduced as an optional parameter. So the corresponding function call doesn't change.

In the second of the examples the parameter is introduced as a required parameter. So the corresponding function call changes accordingly.

Before	After
<pre>// A new parameter will be added to this // function to replace the 1's: function calculate_sum(i) { alert('Adding ' + 1 + ' to ' + i); return 1 + i; } function show_sum() { // Here is the function call: alert('Result: ' + calculate_sum(5)); } // When adding a new parameter we'll specify // that it should be an optional one.</pre>	<pre>// The new parameter i2 has been added // as an optional parameter: function calculate_sum(i, i2) { i2 = i2 1; alert('Adding ' + i2 + ' to ' + i); return i2 + i; } function show_sum() { // The function call has not changed: alert('Result: ' + calculate_sum(5)); }</pre>
<pre>// A new parameter will be added to this // function to replace the 1's: function calculate_sum(i) { alert('Adding ' + 1 + ' to ' + i); return 1 + i; } function show_sum() { // Here is the function call: alert('Result: ' + calculate_sum(5)); } // When adding a new parameter we'll specify // that it should be a required one.</pre>	<pre>// The new parameter i2 has been added // as a required parameter: function calculate_sum(i, i2) { alert('Adding ' + i2 + ' to ' + i); return i2 + i; } function show_sum() { // The function call changed accordingly: alert('Result: ' + calculate_sum(5, 1)); }</pre>

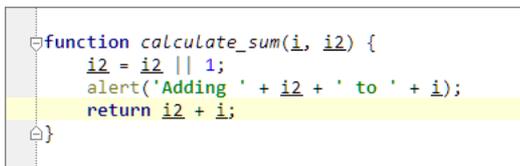
[Introducing a parameter in JavaScript](#)

1. In the editor, place the cursor within the expression to be replaced by a parameter.
2. Do one of the following:
 - o Press `Ctrl+Alt+P`/`Command Alt P`.
 - o Choose **Refactor | Introduce Parameter** in the main menu.
 - o Select **Refactor | Introduce Parameter** from the context menu.
3. If more than one expression is detected for the current cursor position, the **Expressions** list appears. If this is the case, select the required expression. To do that, click the expression. Alternatively, use the `Up` and `Down` arrow keys to navigate to the expression of interest, and then press `Enter` to select it.



4. In the **Introduce Parameter** dialog:
 1. Specify the parameter name in the **Name** field.
 2. The **Value** field, initially, contains the expression that you have selected. Normally, you don't need to change this. (Depending on whether the option **Optional parameter** is selected or not, this will be the value assigned to the new parameter in the function body or passed to the function in the function calls.)
 3. If, when introducing the new parameter, you don't want to change the function calls, select the **Optional parameter** check box. If you do so, the value specified in the **Value** field will be assigned to the new parameter in the function body. The calls to the function (if any) won't change.

If you want to pass the value (specified in the **Value** field) to the new parameter through the existing function calls, clear the **Optional parameter** check box. If you do so, all the function calls will change according to the new function signature; no explicit assignment of the parameter value will be added to the function body.
 4. If more than one occurrence of the expression is found within the function body, you can choose to replace only the selected occurrence or all the found occurrences with the references to the new parameter. Use the **Replace all occurrences** check box to specify your intention.
 5. Click **OK**.



See Also

Procedures:

- [Refactoring Source Code](#)
- [JavaScript-Specific Guidelines](#)

Reference:

- [Introduce Parameter Dialog for JavaScript](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Analyzing Applications

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

In this part:

- [Viewing Structure and Hierarchy of the Source Code](#)
- [Analyzing External Stacktraces](#)
- [Viewing PSI Structure](#)
- [Analyzing Duplicates](#)

See Also

Concepts:

- [Code Inspection](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Viewing Structure and Hierarchy of the Source Code

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm enables you to examine the hierarchy of classes, methods, and calls in the Hierarchy tool window, and explore the structure of source files in the Structure tool window.

Note

- Both Hierarchy and Structure tool windows are available from the **View** menu.
- Hierarchy tool window only becomes available, when a hierarchy is built.
- Hierarchies are built in the **Navigate** menu.

In this part:

- [Building Hierarchies](#)
- [Viewing Hierarchies](#)
- [Viewing Structure of a Source File](#)

See Also

Reference:

- [Hierarchy Tool Window](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
 - <http://youtrack.jetbrains.com/issues/WI>
-

Building Hierarchies

Previous | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

PhpStorm provides facilities for building the following hierarchies:

- **Type** hierarchies that show parent and children classes of a class.
- **Method** hierarchies that show classes where a method:
 - Is defined.
 - Is not defined.
 - Should be defined if the class is not abstract.
- **Call** hierarchies that show callers (supertypes) or callees (subtypes) of a method.

When built, a hierarchy can be immediately [viewed and examined](#) in the **Hierarchy** tool window. By default, every new built hierarchy overwrites the contents of the current tab. You can [retain](#) the current tab and have the next hierarchy built in a new one.

To build a hierarchy of types

1. Select the desired class in the **Project** tool window or open it in the editor.
2. On the main menu, choose **View | Type Hierarchy** or just press **Ctrl+HCommand H**.

To build a method hierarchy

1. Open the file in the editor and place the caret at the declaration of the desired method. Alternatively, select the desired method in the **Project** view.
2. On the main menu, choose **View | Method Hierarchy** or press **Ctrl+Shift+HCommand Shift H**.

To build a hierarchy of method calls

1. Open the file in the editor and place the caret at the declaration or usage of the desired method. Alternatively, select the desired method in the **Project** view.
2. On the main menu, choose **View | Call Hierarchy** or press **Ctrl+Alt+HCommand Alt H**.

See Also

Procedures:

- [Viewing Hierarchies](#)
- [Retaining Hierarchy Tabs](#)

Reference:

- [Hierarchy Tool Window](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
 - <http://youtrack.jetbrains.com/issues/WI>
-

Viewing Hierarchies

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Once [built](#), hierarchies can be brought up for close examination in the **Hierarchy** tool window.

This section describes how to:

- [Show the Hierarchy tool window](#).

Warning

The Hierarchy tool window is not shown when there are no hierarchies to display. You have to build hierarchies first.

Refer to the section [Building Call Hierarchy](#) to learn how to build hierarchies.

- [Navigate between tabs](#).
- [Toggle between hierarchy views](#): show ascending or descending hierarchy (callee vs. caller methods, parent vs. children classes etc.)

To show the hierarchy tool window, do one of following

- On the main menu, choose **View | Tool Windows | Hierarchy**.
- Use **Alt+8Meta 8** keyboard shortcut.

To navigate between the tabs of the hierarchy window, do one of the following

- Right-click the currently displayed tab, and choose **Select Next Tab/Select Previous Tab** on the context menu.
- Use the **Alt+RightCommand Right** and **Alt+LeftCommand Left** keyboard shortcuts.
- Click the currently displayed tab, and choose the next tab to display.

To toggle between views, use the toolbar of the hierarchy tool window

- Click the  button to show caller methods, or supertypes.
- Click the  button to show callee methods, or subtypes.

See Also

Procedures:

- [Building Hierarchies](#)
- [Retaining Hierarchy Tabs](#)

Reference:

- [Hierarchy Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Viewing Structure of a Source File

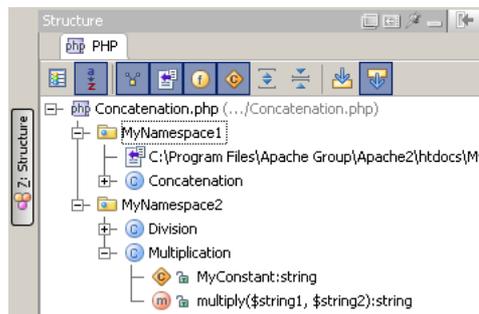
[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

You can examine the structure of the file currently opened in the editor using the **Structure** tool window or the **Structure** pop-up window.

By default, PhpStorm shows all the namespaces, classes, methods, and functions presented in the current file. To have [other members displayed](#), press the corresponding toolbar buttons.

To view the file structure, do one of the following

- On the main menu, choose **View | Tool Windows | Structure**.
- Press **Alt+7Meta 7**.



To have class fields displayed

- Press the **Show Fields** button .

To have constants displayed

- Press the **Show Constants** button .

To have inherited members displayed

- Press the **Show Inherited** button .

By default, PhpStorm shows only methods, constants, and fields defined in the current class. If shown, inherited members are displayed gray.

To have included files displayed

- Press the **Show Includes** button .

See Also

Reference:

- [Structure Tool Window](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi> 
- <http://youtrack.jetbrains.com/issues/WI> 

```
ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js'; var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s); }());
```



PhpStorm 3.0.0 Web Help

Analyzing External Stacktraces

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

You might want to analyze exceptions received by someone else, for example, QA engineers, or investigate a deadlock, or a hang-problem. Unlike the exceptions that you get in the debug mode or when running unit tests, these exceptions do not have links that help you navigate to the corresponding locations in the source code.

With PhpStorm, you can simply copy an exception or full thread dump, paste it to the Stacktrace Analyzer, explore information, and navigate to the corresponding source code.

To analyze an external stack trace or thread dump

1. On the main menu, choose **Tools | Analyze Stacktrace**.
2. In the **Analyze Stacktrace** dialog box that opens, paste the external stack trace or thread dump into the **Put a thread dump here:** text area.
3. Click **OK**. The stacktrace is displayed in the [Run](#) tool window.

See Also

Reference:

- [Run Tool Window](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi> 
- <http://youtrack.jetbrains.com/issues/WI> 

Viewing PSI Structure

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

If you want to explore the internal structure of source code as it is interpreted by PhpStorm, use the PSI viewer.

Note that PSI viewer command is only available when there is at least one plugin module in project.

To view psi structure of the source code

1. On the main menu, choose **Tools | View PSI Structure**.
2. In the [PSI Viewer](#) dialog box, type or paste the fragment of source code to be analyzed in the **Text** area, select file type, and specify the other options.
3. Click **Build PSI Tree**, and preview the generated PSI tree in the **PSI Structure** pane.
If the source code in the **Text** area has been changed, click the same button to refresh preview.

See Also

Reference:

- [PSI Viewer](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi> 
- <http://youtrack.jetbrains.com/issues/WI> 

Analyzing Duplicates

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm helps you find repetitive blocks of code in a certain range, which can be a single file, a project, a module, or a custom scope. Results of analysis display in the dedicated tab of the [Duplicate tool window](#).

To search for duplicates

1. On the main menu, choose **Code | Locate Duplicates**.
2. In the [Specify Code Duplication Analysis Scope dialog](#), specify the analysis scope (whole project, current file, the files that are not under version control, or some custom scope). In addition, you can include test sources into the analysis too. Click **OK**.
3. In the [Code Duplication Analysis Settings dialog](#), check the options to define your preferences for the analysis. You can opt to request identical match for code fragments to be considered duplicates, or specify a certain limit below which the code constructs are not considered duplicates (to avoid reporting about each `if` construct in the source code). Click **OK**.
4. In the [Duplicates tool window](#), explore search results:
 - o View the list of duplicates in the left pane of the tool window.
 - o View differences between the found duplicates in the right pane. Use the arrow buttons to place the selected duplicate in one of the sections of the differences viewer and compare fragments of the code.
 - o Navigate to the duplicates in the editor, using **Jump to Source** or **Show Source** commands of the duplicates' context menu.
 - o Eliminate duplicates from the source code by applying the [Extract method refactoring](#) to the detected repetitive blocks of code that are found and highlighted automatically.
 - o Use the **»**, **«**, or **☒** buttons to insert or remove differences.

See Also

Procedures:

- [Extract Method](#)

Reference:

- [Specify Code Duplication Analysis Scope](#)
- [Code Duplication Analysis Settings](#)
- [Duplicates Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Running

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

This section describes the procedures that are common for the various types of applications:

- [Creating and Editing Run/Debug Configurations](#)
- [Creating and Saving Temporary Run/Debug Configurations](#)
- [Running Applications](#)
- [Rerunning Applications](#)
- [Reviewing Results](#)
- [Stopping and Pausing Applications](#)

For the details related to running applications in the supported frameworks, refer to [Language and Framework-Specific Guidelines](#)

See Also

Concepts:

- [Running and Debugging](#)

Reference:

- [Run/Debug Configurations](#)
- [Run Tool Window](#)

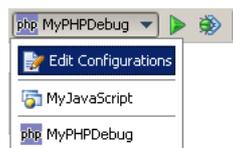
Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Creating and Editing Run/Debug Configurations

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

The available [run/debug configurations](#) are displayed in the drop-down list in the Run area of the main toolbar:



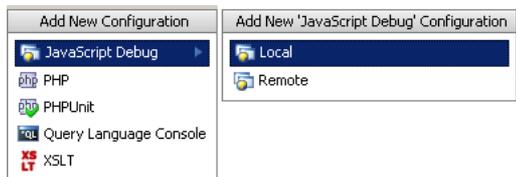
PhpStorm provides the **Run/Debug Configuration** dialog box as a tool for handling run/debug configurations: create configuration profiles or change the existing ones.

There are two main ways to create a run/debug configuration:

- [Create a run configuration manually](#) on the base of the default one, using the [Run/Debug Configuration](#) dialog box.
- [Save a temporary run configuration](#).

To create a run/debug configuration

1. Open the [Run/Debug Configuration](#) dialog box by doing one of the following:
 - o On the main menu, choose **Run | Edit Configurations**.
 - o Press **Alt+Shift+F10** **Shift F10**, then press **00** to display the **Edit Configuration** dialog box or select the configuration from the pop-up window and press **F4F4**.
2. In the [Run/Debug Configuration](#) dialog box, click  on the toolbar or press **InsertInsert**. The drop-down list shows the default run/debug configurations. Select the desired configuration type:



The fields that appear in the right pane display the default settings for the selected configuration type.

Tip

If you want to change the default run/debug configuration settings, expand the **Defaults** node, select the desired configuration type, and modify it as required.

3. In the **Name** text box, specify the name of the new configuration. This name will be shown in the list of available configurations.

Tip

To use an existing configuration as a pattern, create its copy by clicking the **Copy** button  on the toolbar, then change it as required.

4. Specify additional parameters depending on the configuration type. Refer to the descriptions of run/debug configuration parameters below the [Run/Debug Configurations](#) section.
5. Apply the changes and close the dialog box.

See Also

Concepts:

- [Run/Debug Configuration](#)

Procedures:

- [Creating and Saving Temporary Run/Debug Configurations](#)
- [Running](#)
- [Debugging](#)

Reference:

- [Run/Debug Configurations](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Creating and Saving Temporary Run/Debug Configurations

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts:  

Sometimes you might need to run or debug a certain class or file, without creating a dedicated run configuration. In this case, PhpStorm provides a [temporary run configuration](#).

Temporary run/debug configuration is added to the list of available configurations and works same way as the [permanent run/debug configurations](#). You can change its settings using the [Run/Debug Configuration](#) dialog box and optionally save it as permanent.

Creating a run/debug configuration involves specifying the URL address of the Web page to open. For temporary configurations, PhpStorm assembles this URL address automatically. To have it done properly, you need to specify the project root folder URL on the [PHP](#) page of the [Settings](#) dialog box.

To create a temporary run configuration

1. Select the desired PHP class or file in the **Project** tool window. or open it in the editor.
2. On the context menu of the selection, choose **Run <name>** or **Debug <name>**. Alternatively, press **Ctrl+Shift+F10** **Command Shift F10**.
PhpStorm creates a temporary configuration, which appears in the **Select Run/Debug Configuration** drop-down list on the main toolbar when the run or debug session is over.

To save a temporary configuration, do one of the following

- In the **Select Run/Debug Configuration** drop-down list on the main toolbar, choose **Save <configuration name>**.
- In the **Run/Debug Configuration** dialog box, click the  button.

See Also

Concepts:

- [Run/Debug Configuration](#)

Procedures:

- [Creating and Editing Run/Debug Configurations](#)
- [Running](#)
- [Debugging](#)

Reference:

- [Run/Debug Configurations](#)

Web Resources:

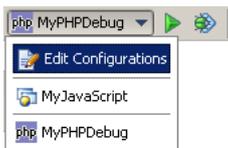
- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Running Applications

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

PhpStorm enables running [entire applications](#) as well as particular [classes or files](#).

When running an application, PhpStorm uses the settings defined in the [Run/Debug Configuration](#) dialog box. All run configurations that exist in a project, are available in the **Select Run/Debug Configuration** drop-down list on the main toolbar:

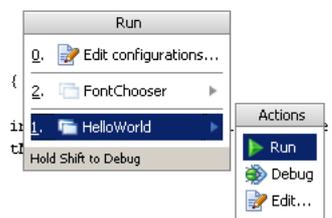


To run an application

- On the main toolbar, select the desired run configuration, and do one of the following:
 - Choose **Run | Run** on the main menu.
 - Click .
 - Press **Shift+F10**.

Tip

Alternatively, press **Alt+Shift+F10**, select the desired run configuration from the pop-up menu, and press **Enter**.



From this pop-up menu you can:

- Invoke **Edit Configuration dialog**.
- Edit the selected configuration before launch (**F4**).
- Instantly delete a temporary configuration (**EditorDelete**).
- Switch from run to debug and vice versa (hold **Shift**).
- Access a previously selected configuration (1).
- Access context-dependent configuration (2 or 3).

This pop-up menu can also be quickly accessed by pressing **F9**, when you're not running any debug session.

See Also

Concepts:

- [Run/Debug Configuration](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Rerunning Applications

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

You can re-run an application if its tab is still opened in the **Run** window. The program re-runs with the initial settings.

To re-run an application

1. In the Run window, select the tab where the desired application is opened.

2. In the toolbar of the Run window, click the Rerun button , or press `Ctrl+F5`/`Command F5`.

Tip

If you want to re-run an application without losing focus in the editor tab you are working in, press `Shift+F10`/`Shift F10`.

See Also

Concepts:

- [Running and Debugging](#)

Procedures:

- [Running Applications](#)

Reference:

- [Run Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Reviewing Results

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

You can review any output from your running applications in the Run window console. The output from each application is displayed in its own tab of the [Run](#) tool window, named after the corresponding [run configuration](#).

If you re-run an application, the new output overwrites the contents of the tab. To preserve the output of an application, even if you re-run it, [pin](#) the output tab.

See Also

Concepts:

- [Running and Debugging](#)

Procedures:

- [Running](#)

Reference:

- [Run Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Stopping and Pausing Applications

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac 

In the [Run](#) tool window, you can stop the program, or pause its output. If a program is stopped, its process is interrupted and exits immediately. When program output is paused, the program continues running in the background, but its output is suspended.

To stop a program, do one of the following

- In the Run tool window, click the Stop button  on the toolbar or press `Ctrl+F2`/`Command F2`.
- To close the active tab, click the Close button , or press `Ctrl+Shift+F4`/`Command Shift F4`.

To suspend program output

- In the Run tool window, click the Pause button  on the toolbar.

See Also

Procedures:

- [Running Applications](#)
- [Rerunning Applications](#)

Reference:

- [Run Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Debugging

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm provides a complete range of facilities to debug source code:

- Breakpoints in HTML, JavaScript, and PHP.
- Customizable breakpoint properties: conditions, pass count, etc.
- Frames, variables, and watches views in the JavaScript debugger UI.
- Runtime evaluation of JavaScript expressions.
- Support for the XDebug tool to debug [PHP applications](#).

To debug an application, perform the following general steps

1. [Configure debugger options](#).
2. [Define a run/debug configuration](#) for the application to be debugged.
3. [Create breakpoints](#) in the source code.
4. [Launch](#) the debugging session.
5. During the debugger session, [step through the breakpoints](#), [examine a suspended program](#), [evaluate expressions](#), change values on-the-fly, and [set watches](#).

See Also

Concepts:

- [Run/Debug Configuration](#)
- [Breakpoints](#)

Reference:

- [Debug Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Using Breakpoints

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

In this section:

- [Accessing Breakpoint Properties](#)
- [Configuring Breakpoints](#)
- [Creating Line Breakpoints](#)
- [Enabling, Disabling and Removing Breakpoints](#)
- [Moving Breakpoints](#)
- [Navigating Back to Source](#)

See Also

Concepts:

- [Run/Debug Configuration](#)
- [Breakpoints](#)

Reference:

- [Debugger](#)
- [Debug Tool Window](#)
- [Breakpoints](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Accessing Breakpoint Properties

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac 

View the list of all breakpoints contained in an application, using the [Breakpoints](#) dialog box. For each individual breakpoint in the list, you can view and change its properties as required. PhpStorm suggests several ways to gain access to the breakpoints properties.

To view the list of all breakpoints and their properties, do one of the following:

- On the main menu, choose **Run | View Breakpoints**.
- Press **Ctrl+Shift+F8** / **Command Shift F8**.
- Right-click a breakpoint icon in the left gutter of the editor, and choose **Properties** on the context menu:



- In the toolbar of the Debug tool window, click .

See Also

Concepts:

- [Breakpoints](#)

Reference:

- [Breakpoints](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Configuring Breakpoints

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

For a breakpoint, you can configure the following properties:

- Actions to be performed upon hitting a certain breakpoint.
- Suspend policy, which defines whether the application should be suspended upon hitting the breakpoint.
- Dependencies on other breakpoints.

All settings are done in the [Breakpoints](#) dialog box, for a breakpoint selected in the list.

To configure actions, suspend policy and dependencies of a breakpoint

1. [Open](#) the **Breakpoint properties** dialog box, click the tab that corresponds to the breakpoint type, and select the desired breakpoint in the list.
2. To display the reaching of a breakpoint as a text message in the debugging console, check the option **Log message to console**.
3. To set a breakpoint the current one depends on, select it from the **Depends on** drop-down list.
4. In the **Suspend policy** area, select one of the option buttons to specify the way a running program will be paused upon reaching a breakpoint. For more information on the **Suspend policy** option, refer to [Breakpoints dialog](#) section.

See Also

Concepts:

- [Run/Debug Configuration](#)
- [Breakpoints](#)

Reference:

- [Breakpoints](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Creating Line Breakpoints

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac 

A **line breakpoint** is a breakpoint assigned to a specific line in the source code.

Note

Line breakpoints can be set on executable lines. Comments, declarations and empty lines are not the valid locations for the line breakpoints.

To create a line breakpoint

1. Place the caret on the desired line of the source code.
2. Do one of the following:
 - Click the left gutter area at a line where you want to toggle a breakpoint.
 - On the main menu, choose **Run | Toggle Line Breakpoint**.
 - Press **Ctrl+F8** / **Command F8**.

See Also

Concepts:

- [Run/Debug Configuration](#)
- [Breakpoints](#)

Reference:

- [Breakpoints](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Enabling, Disabling and Removing Breakpoints

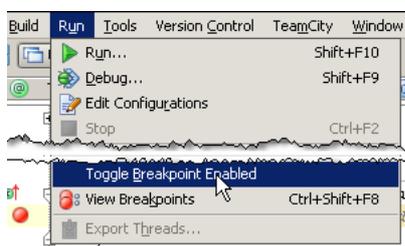
[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

This section describes how to:

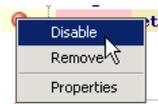
- Temporarily [disable or enable](#) a breakpoint, so that the breakpoint icon changes from  to , and vice versa.
- Permanently [remove](#) a breakpoint.

To toggle between the enabled and disabled state of a breakpoint

1. Place the caret at the desired line with a breakpoint.
2. Do one of the following:
 - o On the main menu, choose **Run | Toggle Breakpoint Enabled**:



- o Right-click the desired breakpoint icon, and choose **Disable** or **Enable** on the context menu:



- o **Alt+Click** on the breakpoint icon.

Tip

Alternatively, [open the Breakpoints dialog box](#), select the desired breakpoint, and select or clear the check box to its left.

To permanently remove a breakpoint, do one of the following

- [Open the Breakpoints dialog box](#), select the desired breakpoint, and click **Remove**.
- Right-click the desired breakpoint in the editor, and choose **Remove** on its context menu.

See Also

Concepts:

- [Run/Debug Configuration](#)
- [Breakpoints](#)

Reference:

- [Breakpoints](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Moving Breakpoints

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

To move a breakpoint

- Just drag a line breakpoint to the needed line.

See Also

Concepts:

- [Run/Debug Configuration](#)
- [Breakpoints](#)

Reference:

- [Breakpoints](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Navigating Back to Source

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

To navigate to the breakpoints source

1. [Open the Breakpoints dialog box](#).
2. Select the breakpoint entry from the list.
3. Do one of the following:
 - o To view the selected breakpoint without closing the dialog box, click **View Source**.
 - o To open the file with the selected breakpoint for editing, and close the dialog box, click **Go to**. So doing, the caret rests at the line marked with the breakpoint in question.

See Also

Concepts:

- [Breakpoints](#)

Reference:

- [Breakpoints](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Configuring Debugger Options

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm supports debugging for JavaScript and PHP applications, classes, and files. The JavaScript debugging functionality is incorporated in PhpStorm, you only need to [configure its settings](#).

To debug PHP applications, you need to [configure a dedicated tool](#).

Warning

Debugging for JavaScript applications is supported only in the [Firefox](#) and [Google Chrome](#) browsers.

See Also

Procedures:

- [Configuring a Debugging Engine](#)
- [Configuring JavaScript Debugger](#)

Reference:

- [Debugger](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Starting the Debugger Session

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Before debugging

- [Set breakpoints](#) in the source code.
- If necessary, create or modify the corresponding [Run/Debug configuration](#).

The debugging session starts with the selected run/debug configuration. Note that several debug processes can be launched simultaneously.

To start debugging an application, do one of the following

- Select the run/debug configuration to execute, and then

- Click the **Debug** button  on the toolbar.
 - Choose **Run | Debug** on the main menu.
 - Press **Shift+F9**.
- Press **Alt+Shift+F9** **Shift F9**, select the configuration from the pop-up menu, and press Enter.

See Also

Concepts:

- [Running and Debugging](#)

Procedures:

- [Monitoring the Debug Information](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Pausing and Resuming the Debugger Session

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

When a breakpoint is hit, or when a running thread or an application is paused manually, the debugging session is suspended.

To pause the debugger session do one of the following

- On the main menu, choose **Run | Pause Program**.
- Click  on the Debug toolbar.

To resume the debugger session do one of the following

- On the main menu, choose **Run | Resume Program**.
- Click  on the Debug toolbar.
- Press **F9**.

See Also

Reference:

- [Debug Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Examining Suspended Program

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

When a breakpoint is hit, or a program execution is manually [suspended](#), you can [examine](#) your application by analyzing frames.

A `frame` corresponds to an active method or function call. A frame stores the local variables of the called method or function, the arguments to it, and the code context that enables expression evaluation.

All currently active frames are displayed on the **Frames** pane of the [Debug](#) tool window, where you can [switch](#) between them and analyze the information stored therein.

To navigate between frames

- Use up and down arrow buttons on the toolbar.
- Use **Up** and **Down** shortcuts.

Note

You do not need to perform any actions to navigate to the frame's source code. PhpStorm automatically jumps to the source code of the selected frame in the editor.

In this section:

- [Adding, Editing and Removing Watches](#)
- [Inspecting Watched Items](#)
- [Evaluating Expressions](#)
- [Navigating to Source Code from the Debug Tool Window](#)

See Also

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>

- <http://youtrack.jetbrains.net/issues/WI>

Adding, Editing and Removing Watches

Previous | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

If you want to evaluate a number of variables or expressions in the context of the current frame, and view all of them simultaneously, you can create `watches` for them. The values of the expression are updated with each step through the application, but are only visible when the application is suspended. Unlike the [Expression Evaluation feature](#), these expressions are persisted as the part of your project.

This section describes how to add items to watches, change and remove watches.

To add an item to a watch

Do one of the following:

- In the [Watches](#) pane, click , or just press `InsertInsert`.
- On the [Variables](#) pane, in the [Inspection](#) window, or in the [Evaluate Expression](#) dialog box, right-click the desired item and choose **Add to Watches** on the context menu.
- Select the desired item in the [Variables](#) pane and drag it to the [Watches](#) pane.
- Select item in the editor, right-click it and select **Add to Watches** on the context menu.

To edit a watch

- To change the expression represented by a watch, right-click the desired watch and select **Edit** on the context menu.

To remove a watch

1. In the [Watches](#) pane, select a watch to be deleted.
2. On the context menu, choose **Remove Watch**, or press `EditorDeleteEditorDelete`.

Tip

You can navigate from a backtrace in the [Watches](#) pane to the respective line of the source code. To do that, right-click a line of backtrace, and choose **Jump to Source** on the context menu, or just press `F4F4`.

See Also

Procedures:

- [Debugging](#)
- [Pausing and Resuming the Debugger Session](#)

Reference:

- [Debug Tool Window](#)
- [Evaluate Expression](#)
- [Debug Tool Window. Variables](#)
- [Debug Tool Window. Watches](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

```
ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js'; var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s); }());
```



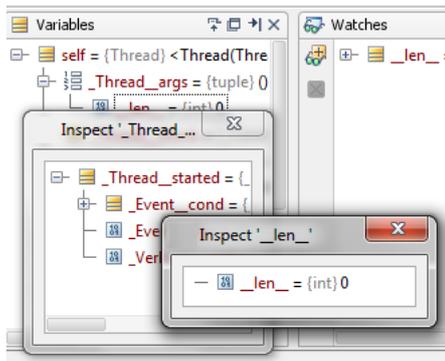
PhpStorm 3.0.0 Web Help

Inspecting Watched Items

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm helps inspect any reference in its own window. For example, if you need to examine several references in detail, you can open an inspection window for each of them. So doing, a separate window is created for each reference and all of its child references.

The inspection windows are non-modal, and you can launch as many as of them required. All changes of the references are immediately reflected in the corresponding inspection windows.



To inspect a reference

1. Select the item to be inspected on the **Variables** or **Watches** pane.
2. On the context menu, choose **Inspect**.

See Also

Procedures:

- [Debugging](#)

Reference:

- [Debug Tool Window](#)
- [Evaluate Expression](#)
- [Debug Tool Window. Variables](#)
- [Debug Tool Window. Watches](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Evaluating Expressions

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

PhpStorm enables you to [evaluate an arbitrary expression](#) in the following modes:

- **Expression Mode** for evaluating single-line expressions.
- **Code Fragment Mode** for evaluating short code portions introducing them in the **Statements to evaluate** text field. Supported constructs are declarations, assignments, loops and if/else.

Besides that, PhpStorm provides a way to [quickly evaluate](#) in the editor an expression at caret, or a selection.

Warning

While using the **Expression Evaluation** feature, be aware of the following:

- A method can be invoked within the **Expression Evaluation** dialog only if the debugger has stopped at a breakpoint, but has not been paused.
- Expression Evaluation can be only "single-level". In other words, if PhpStorm stops at a breakpoint within a method called from the Expression Evaluation, you cannot use the Expression Evaluation feature again.

To evaluate an arbitrary expression

1. Open the [Evaluate Expression](#) dialog box. Do one of the following:
 - Choose **Run | Evaluate Expression** on the main menu.
 - Press **Alt+F8Alt F8**.
 - To have a specific variable evaluated, select it on the **Variables** pane, then choose **Run | Evaluate Expression** or press **Alt+F8Alt F8**.
2. In the **Evaluate Expression** dialog box, specify the expression to evaluate. Do one of the following:
 - In the **Expression** field, type the expression in question or choose one of the previously evaluated expressions from the drop-down list.

Note

If you have selected a specific variable on the **Variables** pane, this variable will be displayed in the **Expression** text box.

- To evaluate a code fragment, click the **Code Fragment Mode** button and fill in the **Code Fragment** text box.

Tip

To return to the original mode, click the **Expression mode** button.

3. Click the **Evaluate** button. The **Result** read-only field shows the evaluation output. If the specified expression cannot be evaluated, the **Result** field explains the reason.

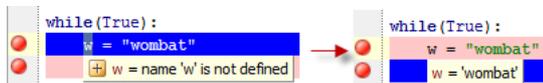
Tip

During the debugger session, the value of any expression at caret is shown at the tooltip every time you hover your mouse pointer over it. If an expression contains children, clicking  expands the node and displays all children.

However, `Quick evaluate expression` helps you view the expression value using the keyboard only.

To quickly evaluate an expression in the editor

1. Place the caret at the desired location, or select an expression to be evaluated.
2. Press `Ctrl+Alt+F8` / `Command Alt F8`. The tooltip with the expression value appears under the selected expression:



See Also

Procedures:

- [Debugging](#)

Reference:

- [Debug Tool Window](#)
- [Evaluate Expression](#)
- [Debug Tool Window. Variables](#)
- [Debug Tool Window. Watches](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Navigating to Source Code from the Debug Tool Window

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

To navigate to the source code, do one of the following:

- Select the desired item in the **Variables** tab and press `F4F4`.
- Right-click an item in the **Variables** tab, and select **Jump to Source** from the context menu.

See Also

Procedures:

- [Debugging](#)

Reference:

- [Debug Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Stepping Through the Program

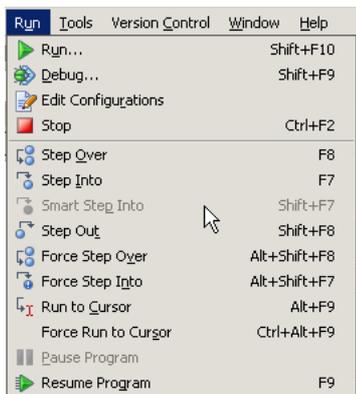
[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

When a breakpoint is reached, the `Debug` tool window becomes active and enables you to get control over the program's execution. For this purpose, you can use the `Run` menu commands, or the icons on the `stepping toolbar` of the `Debug` tool window.

Each stepping action advances the `execution point` to the next execution location, depending on the action you choose.

To step through the program using breakpoints, do one of the following:

- On the main menu, choose `Run | <stepping command>`



- Use keyboard shortcuts.
- Use the buttons in the [stepping toolbar](#) of the Debug tool window.



See Also

Concepts:

- [Breakpoints](#)
- [Running and Debugging](#)

Reference:

- [Debug Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Choosing a Method to Step Into

Previous | Next | [See Also](#) | [Comments](#) | Shortcuts: Mac

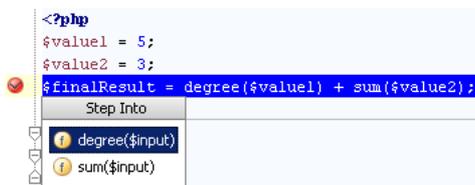
When you reach a line with calls of several methods, you can choose the method you want to step into.

Tip

If you choose a method of a class stepping into which is suppressed on the [Debugger Stepping](#) page of the [Settings](#) dialog box, the suppression is overridden as when you invoke the `Force Step Into` command.

To choose a method to step into

1. On the main menu, choose `Run | Smart Step Into` or press `Shift+F7`.
2. In the pop-up window, choose the desired method from the list.



See Also

Concepts:

- [Breakpoints](#)
- [Running and Debugging](#)

Reference:

- [Debug Tool Window](#)
- [Debugger Stepping](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Finding the Current Execution Point

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

When a program is suspended, the source file, associated with the current execution point, is opened in the editor. The current `execution point` (the next line to be executed) is marked with a blue line.

You can visit the other files, and then return to the current execution point using the actions described in this section.

To find the current execution point, do one of the following

- On the main menu, choose **Run | Show Execution Point**.
- Press `Alt+F10` `Alt F10`.
- Click  on the [stepping toolbar](#) of the Debug tool window.

See Also

Concepts:

- [Breakpoints](#)

Reference:

- [Debug Tool Window](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

```
ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js'; var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s); })());
```



PhpStorm 3.0.0 Web Help

Monitoring the Debug Information

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Information of a debugging session is displayed in the dedicated tabs of the [Debug](#) tool window, named after the selected run/debug configuration.

For each session, use the [Console](#) tab to view the debugger messages and application output, and the [Debug](#) tab to monitor threads and frames:



[Click thumbnail to view larger image.](#)

When displaying and modifying local variables or watches values, PhpStorm uses the [Default Encoding](#) setting for the current project or the [IDE encoding](#) if no encoding is specified at the project level. The same setting is used when showing the PHP console script output.

See Also

Concepts:

- [Running and Debugging](#)

Procedures:

- [Starting the Debugger Session](#)
- [Stepping Through the Program](#)
- [Finding the Current Execution Point](#)
- [Evaluating Expressions](#)
- [Adding, Editing and Removing Watches](#)
- [Inspecting Watched Items](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Testing

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

This section covers the issues, that are common for all the supported testing frameworks:

- [Creating Run/Debug Configuration for Tests](#)
- [Performing Tests](#)
- [Monitoring and Managing Tests](#)
- [Rerunning Tests](#)
- [Terminating Tests](#)

- [Viewing and Exploring Test Results](#)

For the framework-specific testing procedures, refer to the sections:

- [Testing PHP Applications](#)
- [Unit Testing JavaScript](#)

To create and run unit tests on php applications, perform the following general steps:

- [Enable PHPUnit support](#).
- [Generate unit tests](#) and mark the folder where they are stored as [test folder](#).
- [Create a run configuration](#):
 - To run unit tests locally, create a [PHPUnit](#) configuration.
 - To run unit tests on a remote server, create a [PHPUnit on Server](#) configuration.
- [Launch unit tests](#) and [monitor test results](#) in the [Run](#) tool window.

In this part:

- [Creating Run/Debug Configuration for Tests](#)
- [Performing Tests](#)
- [Monitoring and Managing Tests](#)
- [Rerunning Tests](#)
- [Terminating Tests](#)
- [Viewing and Exploring Test Results](#)

See Also

Procedures:

- [Configuring Project Settings](#)
- [Configuring Content Roots](#)
- [PHP-Specific Guidelines](#)

Reference:

- [Run Tool Window](#)
- [Test Runner Tab](#)
- [Run/Debug Configuration: PHPUnit](#)
- [Run/Debug Configuration: PHPUnit on Server](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wiki>
- <http://youtrack.jetbrains.net/issues/WI>

```
ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js'; var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s); }());
```



PhpStorm 3.0.0 Web Help

Creating Run/Debug Configuration for Tests

Previous | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

You can run your tests (test cases, test suites, etc.) using [run/debug configurations](#), in the way similar to running ordinary applications. PhpStorm provides a framework for creating special run/debug configurations for testing purposes, where a test can be specified as a target.

In addition to the [regular test configuration creation](#), PhpStorm supports creating [test configurations for specific targets](#), for example for all tests in a file, class, or directory.

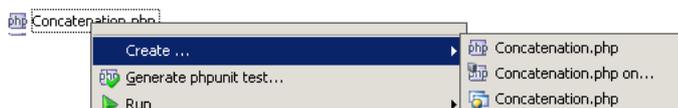
With PhpStorm you can also create configurations for running PHP unit tests locally or on a remote server.

To create a test configuration

1. Open the [Run/Debug Configuration](#) dialog box by doing one of the following:
 - On the main menu, choose **Run | Edit Configurations**.
 - Press `RunConfigurationRunConfiguration` and choose **Edit Configuration** on the context menu.
2. Click the **Add** button  on the toolbar and select the desired configuration type:
 - To have unit tests executed locally, choose **PHPUnit**.
 - For remote test execution, choose **PHPUnit on Server**.
3. In the dialog box that opens, specify the test scope, configuration parameters, and activities to perform before test execution. Apply the changes and close the dialog box.

To create a test configuration for a specific target

1. In the **Project** tool window, right-click the desired test directory, individual file, or class.
2. On the context menu of the selection, choose **Create**. PhpStorm displays a list of suggested configurations.



Each configuration has the name of the selected target and is supplied with an icon indicating its type.

- o - the icon marks a **PHPUnit** configuration for running tests locally.
- o - the icon marks a **PHPUnit on Server** configuration for running tests remotely.

Choose the configuration of the desired type.

3. In the [Run/Debug Configuration](#) dialog box that opens, specify the configuration parameters and activities to perform before test execution. Apply the changes and close the dialog box.

See Also

Procedures:

- [Creating and Editing Run/Debug Configurations](#)

Reference:

- [Run/Debug Configurations](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Performing Tests

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Generally, PhpStorm runs and debugs tests in the same way as other applications, by running the run/debug configurations [you have created](#). When doing so, it passes the specified test classes or methods to the test runner.

In many cases, you can initiate a testing session from a context menu. For this purpose, the **Run** and **Debug** commands are provided in certain context menus. For example, these commands are available for a test class, directory, or a package in the [Project tool window](#). They are also available for a test class or method you are currently working with in the editor.

If you run a test for which there is no [permanent run/debug configuration](#), a temporary configuration is created. You can then save such a configuration if you want to keep it.

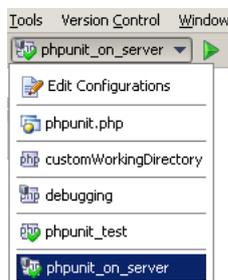
The tests run in the background, so you can execute several tests at the same time.

Each test gets its own tab in the [Run tool window](#) (the [Test Results tab](#)).

Running or debugging a test

To start running or debugging a test, you can use the main toolbar, or a context menu in the **Project** tool window or in the editor:

- Using the main toolbar:
 1. Select the necessary run/debug configuration from the list on the main toolbar.



2. Click **Run** or **Debug** to the right of the list. (Alternatively, choose **Run | Run** (Shift+F10) or **Run | Debug** (Shift+F9) from the main menu.)

Tip

To see the list of available run or debug configurations and to quickly select the one you want, you can use the following keyboard shortcuts: **Alt+Shift+F10** for the run configurations, or **Alt+Shift+F9** for the debug configurations.

- Using a context menu:

Right-click a test file or test class in the **Project** tool window, or open it in the editor, and right-click the background. On the context menu, choose **Run < class name>/Run < file name>** or **Debug**.

Tip

For a test method, open the class in the editor and right click anywhere in the method. The context menu suggests the command **Run / Debug <method name>**.

See Also

Reference:

- [Run Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

```
ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js'; var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s);})();
```



PhpStorm 3.0.0 Web Help

Monitoring and Managing Tests

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Test progress and results display in the dedicated [test runner tabs](#) of the Run tool window.

You can [rerun](#), [terminate](#), and [suspend](#) execution of tests same way as you do it for running applications. In addition to the common running actions, in the test runner you can:

- Navigate through the list of test cases using arrow keys.
- Navigate between failed tests using the and buttons or `Ctrl+Alt+UpCommand Alt Up` or `Ctrl+Alt+DownCommand Alt Down` keyboard shortcuts.
- View the total number of tests being run in the current session. The summary information is displayed in a message line on the top of the tool window. After completion of the tests, the message informs you about the number of failed tests and elapsed time.
- View testing progress in the progress bar.
- Show or hide information about the passed tests, using the button.
- Navigate from the stack trace to the problem location in the source code by clicking the hyperlink in the **Output** pane.
- Enable and disable the following functionality by clicking the cog button and selecting the relevant items from the context menu:
 - Monitoring execution of the current test.
 - Have the first failed test selected automatically upon completing the suit.
 - Navigating to the stack trace.
 - Automatic scrolling to the source code.
 - Have the corresponding source code opened at exceptions.
 - Have statistic shown in the **Statistics** pane.

See Also

Reference:

- [Test Runner Tab](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Rerunning Tests

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

You can repeat your test session, or individual tests without leaving your test runner tab of the Run tool window. The tests are performed again using the same run configuration as in the initial run.

To rerun a testing session

Do one of the following:

- Click the rerun button on the toolbar of the Run tool window.
- Use `Ctrl+F5Command F5` keyboard shortcut.

To rerun an individual test

1. In the testing tab of the test runner, right click a test case node or a test.
2. On the context menu, choose **Run <test target>**.

See Also

Reference:

- [Test Runner Tab](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Terminating Tests

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

You can abort a running test session at any time. So doing, all tests that are current running will stop immediately. The icons of the tests in the test runner reflect the statuses of the tests (passed, failed, aborted, or never run).

To stop a testing session, do one of the following

- On the toolbar of the test runner, click the stop button .
- Use `Ctrl+F2Command F2` keyboard shortcut.

See Also

Reference:

- [Test Runner Tab](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

```
ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js'; var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s); }());
```



PhpStorm 3.0.0 Web Help

Viewing and Exploring Test Results

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Depending on the selected node in the tree view of tests, the Test Runner displays information on the various levels. In the Test Runner, you can view statistics of the tests, navigate to stacktrace, show or hide successful tests, and more.

Use the [Testing toolbar](#) to control visual representation of the test results.

To show or hide statistics

The **Statistics** tab shows information about the elapsed time and memory usage of each test.

- Click the  button on the Testing toolbar to reveal the drop-down menu, and then click the **Show Statistics** checked command.

Note

The values displayed are not accurate and only give an approximate estimate of the test performance. For example, if a garbage collector works during the test run, the memory usage shown in the **Statistics** tab is wrong.



Click thumbnail to view larger image.

See Also

Reference:

- [Test Runner Tab](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Remote Hosts

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Deploying applications on remote hosts, managing their contents, and synchronizing with deployed applications are supported via the **Remote Hosts Access** bundled plugin, which is by default enabled. If not, activate it in the [Plugins](#) page of the [Settings](#) dialog box.

After you have [configured access](#) to a remote host, you can [upload](#), [download](#), and [manage files](#) on it directly from PhpStorm.

Moreover, you can suppress uploading or downloading specific files or entire folders by [excluding them from deployment](#).

Finally, you can optimize you workflow by configuring content roots so specific folders are not involved in indexing, which significantly saves project indexing time.

To configure access to a remote host

1. Create a [server configuration](#).
2. [Define mappings](#) to set correspondence between local folders, remote folders, and URL addresses to access the data on the server.
3. [Customize the upload](#) procedure by specifying additional options.
4. Specify the [files and folders to be excluded from deployment](#).

See Also

Procedures:

- [Running](#)

Reference:

- [Deployment](#)
- [Remote Host Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Creating a Server Configuration

Previous | [Next](#) | [See Also](#) | [Comments](#)

A `server` configuration defines:

1. The host to upload the sources to.
2. The `server configuration root`: the highest folder in the destination host hierarchy that can be accessed through the server configuration.
3. The URL address to access the server configuration root.
4. The protocol to transfer the data.
5. Correspondence between local folders, destination folders on the host, and URL addresses to access uploaded folders.

You can define as many configurations as necessary, thus enabling flexible switching between deployment setups.

To create a server configuration

1. Open the [Deployment](#) dialog box. Do one of the following:
 - [Open the project settings](#) and click **Deployment**.
 - Choose **Tools | Deployment | Configuration** on the main menu.
2. On the [Deployment](#) page, click the **Add** toolbar button . In the [Add Server](#) dialog box that opens, specify the configuration name and the way to access the destination.
 - To bind your sources with the upload destination via an [association between a volume and a directory on another volume](#), choose **Mounted Folder**.
 - To copy the sources to the destination via the **File Transfer Protocol** or the **Safe File Transfer Protocol**, choose **FTP** or **SFTP** respectively.
3. Click **OK** to return to the **Connection** tab of the [Deployment](#) page, where the name of the created configuration and the selected access type are displayed in the corresponding fields. Change the settings, if necessary.

Tip

Click the **Use as Default** toolbar button  to have PhpStorm silently apply the current configuration in the following cases:

- [Automatic upload of changed files](#).
 - [Manual upload of files](#) without choosing the target host.
 - [Comparing local files and folders](#) with their remote versions.
4. Specify the [connection settings](#) depending on the chosen configuration type.
 5. Specify the `server configuration root`, which is the highest folder accessible through the server configuration.
 - For **FTP** or **SFTP** configurations, specify the path to the root folder relative to the host in the **Root Path** text box.
 - For **Mounted Folder** configurations, specify the full path to the destination on the mounted folder.
 6. In the **Web server root URL** text box, type the URL address to access the server configuration root. Click the **Open** button to make sure that the specified URL address is accessible and points at the correct Web page.
 7. Specify bilateral or trilateral [mappings](#), depending on the chosen configuration type.

See Also

Procedures:

- [Remote Hosts](#)
- [PHP-Specific Guidelines](#)

Reference:

- [Deployment](#)
- [Deployment: Connection Tab](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Defining Mappings

Previous | [Next](#) | [See Also](#) | [Comments](#)

To map local folders to remote folders and URL addresses

1. [Open the project settings](#) and move to the [Deployment](#) page.

Tip

Alternatively, choose **Tools | Deployment | Configuration** on the main menu.

2. On the left-hand pane, select the server configuration to define mappings for and switch to the [Mappings](#) tab.

Tip

If no applicable server configuration is available, [create a new configuration](#).

3. In the **Paths** area, define the mappings between local folders, destination folders, and corresponding URL addresses. Use the **Add** and **Remove** buttons to add and delete items to the list of mappings.
 - o In the **Local Path** text box, specify the absolute path to the desired local folder in the project tree.
 - o In the **Deployment Path** text box, specify the destination to upload the selected folder to. Type the path relative to the server configuration root or the path to the destination folder relative to the mounted folder.
 - o In the **Web Path** text box, type the path relative to the server document root which is specified in the [URL](#) text box on the [Deployment](#) page.

Tip

If the mapping for a project source root is missing, PhpStorm informs you about it. Click **Fix** to have the corresponding line added to the list and specify the mapping.

See Also

Procedures:

- [Remote Hosts](#)
- [PHP-Specific Guidelines](#)

Reference:

- [Deployment: Mappings Tab](#)
- [Deployment](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Customizing Upload

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

To customize the deployment procedure

1. [Open the project settings](#).
2. Click **Options** below the [Deployment](#) node or choose **Tools | Deployment | Options** on the main menu. In the [Options](#) dialog box that opens, specify additional settings:
 - o To have PhpStorm skip specific files or entire folders during deployment, in the **Exclude items by name** text box, specify the patterns that define the names of these files and folders. Use semicolons as delimiters. Wildcards are welcome.

The exclusion is applied recursively. This means that if a matching folder has subfolders, the contents of these subfolders are not deployed either.
 - o Specify details of the deployment procedure by selecting or clearing corresponding check boxes.
3. On the [Deployment](#) page, click the **Advanced Options** button and specify additional uploading settings in the **Advanced FTP Options** dialog box that opens:
 - o To set the client to the [passive mode](#), select the **Passive mode** check box.
 - o To have the hidden files and directories (those with names that start with a dot .) shown in the [Server Browser Tool Window](#), select the **Show Hidden Files** check box.
 - o To ensure [compatibility in child file naming](#) with your FTP server, select the **Compatibility mode** check box. This is helpful if the remote FTP server reports the following error:

```
Invalid descendant file name <file name>
```

Note

Selecting this option may slow down synchronization with the server.

See Also

Language and Framework-Specific Guidelines:

- [Remote Hosts](#)
- [PHP-Specific Guidelines](#)

Reference:

- [Deployment](#)
- [Advanced Options Dialog](#)
- [Options](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Uploading and Downloading Files

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

There are four main ways to have the local files uploaded to the target host:

- [Manually](#), through a menu command.
- [Automatically, before launching](#) the application or starting a debugging session.
- [Automatically, every time a file is updated](#).

Note

In this case, the [default server configuration](#) is used.

Besides uploading files, you can also have PhpStorm [download files](#) that were deployed according to the [default server configuration](#).

To upload files or folders manually

1. Choose **Tools | Deployment** on the main menu.
2. Do one of the following:
 - To run the upload according to the default configuration, choose **Upload to <default server configuration>**.
 - To run the upload according to a specific configuration, choose **Upload to**. Then from the pop-up menu, select the server configuration that defines access to the required host

To have application sources uploaded automatically before application start

1. During [creation of a run/debug configuration](#), select the **Upload Files to Remote Host** check box in the **Before launch** area.
2. Click the **Browse** button  next to the check box. In the **Upload to Remote Host** dialog box that opens, specify the following:
 - The server configuration to access the required host.
 - The local root folders to upload. All the folders and files under the specified roots will be uploaded.

To have PhpStorm upload changed files automatically, do one of the following

- Choose **Tools | Deployment | Automatic Upload** on the main menu.
- [Open the project settings](#) and click **Options** below the **Deployment** node. In the **Options** dialog box, select the **Upload changed files automatically to the default server** check box.

To download files

- Choose **Tools | Deployment | Download from <default server configuration>** on the main menu.

Note

This functionality is available only if you have previously appointed a configuration as default.

See Also

Procedures:

- [Remote Hosts](#)

Reference:

- [Deployment](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

2.0+

Accessing Files on Remote Hosts

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Once you have [deployed](#) an application onto a remote host, the following manipulations are available:

You can also [compare](#) remote files and folders with their local versions.

Note

Access to remote hosts is controlled through [server configurations](#) of the type **FTP**, **SFTP**, or **Mounted Folder**.

To access a remote host

1. Open the [Remote Host](#) tool window. On the main menu, choose one of the following:
 - **View | Tool Windows | Remote Host**
 - **Tools | Deployment | Browse Remote Host**
2. Select the required server configuration from the drop-down list. The tool window shows the remote host contents as a tree view of files and folders.

Tip

If now relevant server configuration is available in the drop-down list, click the **Browse** button  and configure access to the host in the [Deployment](#) dialog box that opens.

To create a directory on a remote host

- Select the parent directory and choose **Create Folder** on the context menu.

To remove a file or folder on a remote host

- Select the required file or folder and choose **Delete** on the context menu.

To rename a file or folder on a remote host

- Select the required file or folder in the tree view, choose **Rename** on the context menu, and specify the new name in the **Rename** dialog box that opens.

To move a file or folder

1. Select the file or folder and choose **Cut** on the context menu.
2. Select the folder to move the file or folder to and choose **Paste** on the context menu. Then confirm the changes in the **Move remote Items** dialog box that opens.

Tip

Alternatively, select the required file or folder and drag it to the new location. Then confirm the changes in the **Move remote Items** dialog box that opens.

See Also

Procedures:

- [Remote Hosts](#)

Reference:

- [Remote Host Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Excluding Files and Folders from Deployment

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Suppressing deployment (that is, uploading, downloading, and synchronization) for files or folders with sources ensures that the sources are protected against accidental update. When applied to non sources, it saves system resources because huge amounts of media, caches, or temporal files are no longer copied hither and thither.

You may need to suppress deployment in the following cases:

1. You are going to work with externally created and uploaded source code. To process these remote sources in PhpStorm, you have to download them and arrange them in a project. However there are some sources that you should not update at all. On the other hand, the folders on the remote host also may contain huge amounts of media, caches, temporal files, that you actually do not need in your work.
2. You have already downloaded the data from the server and arranged them in a PhpStorm project. However, for this or that reason, you need to have some files or folders on the server protected against deployment, for example, to prevent accidental overwriting.
3. The local copy of an application contains both source code and other data that you do not need to upload. Besides, you want to protect some sources against overwriting by mistake. In this case, you can suppress deployment for all files and folders that should not be uploaded.

There are two ways to **exclude folders** from deployment:

- Explicitly, by marking the corresponding paths as excluded [during project creation](#), or in the [Remote Host](#) tool window, or in the [Excluded Paths](#) tab of the [Deployment](#) dialog box.

Tip

In the [Remote Host](#) tool window, you can exclude both entire folders and specific files.

- **By name**, that is, by specifying [patterns that determine the names](#) of files and folders to be excluded in the **Exclude Items by Name** text box of the [Options](#) dialog box.

Separate files can be protected against deployment only through [excluding them by name](#).

Note

In either case, the exclusion is applied recursively. This means that if the path to a folder is explicitly marked as excluded or the folder name matches the pattern, the contents of all its subfolders, if any, are also protected against deployment.

In this section:

- [Excluding folders on server from deployment during project creation](#)
- [Excluding folders on server from deployment after project creation](#)
- [Excluding local folders from deployment](#)
- [Excluding files and folders from deployment by name](#)
- [Removing exclusion mark](#)

To exclude a folder on remote host from deployment during project creation

1. Initiate creation of a project from existing files by choosing **New Project from Existing Files** on the main menu. The **New Project** wizard starts.

- On the first, [Choose Your Scenario](#), page of the wizard, specify the method to access the remote host, then follow the wizard.
- On the [Choose Remote Path](#) page of the wizard, select the folder and click the  icon on the toolbar or choose **Project Root** on the context menu. PhpStorm marks the selected folder with the  icon.

To exclude a folder on server from deployment after project creation, do one of the following:

- Choose the required folder right in the [Remote Host](#) tool window:
 - On the main menu, choose **Tools | Deployment | Browse Remote Host** or **View | Tool Windows | Remote Host**.
 - In the [Remote Host](#) tool window that opens, select the relevant [server configuration](#) from the drop-down box.
 - Select the folder to exclude and choose **Exclude Path** on the context menu of the selection.
- Add the required folder to the list of excluded paths:
 - [Open the project settings](#) and click **Deployment** or choose **Tools | Deployment | Configuration** on the main menu.
 - In the [Deployment](#) dialog box, switch to the [Excluded Paths](#) tab. The tab shows list of previously excluded local and remote folders.
 - Click the **Add deployment path** button. An empty line is added to the list.
 - Click the **Browse** button . The **Select remote excluded path** dialog box that opens shows the data on the host accessed through the selected server configuration. Select the required folder.
 - When you **OK**, you return to the [Excluded Paths](#) tab, where the selected remote folder is added to the list.

To exclude a local folder from deployment

- Do one of the following:
 - [Open the project settings](#) and click **Deployment**.
 - Choose **Tools | Deployment | Configuration** on the main menu.
- In the [Deployment](#) dialog box, switch to the [Excluded Paths](#) tab. The tab shows list of previously excluded local and remote folders.
- Click the **Add local path** button. In the empty line that is added to the list, specify the location of the folder to be protected against deployment. Type the path manually or click the **Browse** button  and choose the required folder in the **Select Path** dialog box that opens.

To have files or folders excluded by name

- Do one of the following:
 - [Open the project settings](#) and click **Deployment**, then click **Options** below the [Deployment](#) node.
 - Choose **Tools | Deployment | Options** on the main menu.
- In the [Options](#) dialog box that opens, specify the patterns that define the names of these files and folders in the **Exclude items by name** text box. Use semicolons as delimiters. Wildcards are welcome.

The exclusion is applied recursively. This means that if a matching folder has subfolders, the contents of these subfolders are not deployed either.

additional settings:

To get an excluded file or folder involved in deployment again

- Select the file or folder to return to deployment and choose **Remove Path from Excluded** on the context menu of the selection.

Tip

Returning a folder to deployment automatically affects all its subfolders and files.

See Also

Language and Framework-Specific Guidelines:

- [Remote Hosts](#)
- [Creating a Project from Downloaded Files](#)
- [Customizing Upload](#)

Reference:

- [Deployment: Excluded Paths Tab](#)
- [Options](#)
- [Deployment](#)
- [Remote Host Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Creating a Project from Downloaded Files

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

From time to time, you may need to update externally created and uploaded applications. To do that in PhpStorm, you need to arrange the downloaded files in a [project](#).

PhpStorm interacts with the remote host where the files are located according to a [server access configuration](#). You can use an appropriate existing configuration or define a new one during project creation.

To create a PhpStorm project from downloaded files

1. On the main menu, choose **New Project from Existing Files** to start the **New Project** wizard.
2. On the first, [Choose Your Scenario](#), page of the wizard, specify the method to access the remote host. Do one of the following:
 - o To have PhpStorm download the remote files via network or from a mounted folder, choose **Web server is on a remote host, files are accessible via network share or mounted drive**.
 - o To have PhpStorm download the remote files via the FTP or SFTP protocol, choose **My web server is on a remote host, files are accessible via FTP/SFTP**.

Click **Next** when ready.

3. On the second, [Specify Local Path](#), page of the wizard, specify the project location and choose the download strategy to use:
 1. Appoint the parent folder to the create the project folder in. Type the path to it manually in the **Project local path** text box or click the **Browse** button  and select the desired location in the **Select Path** dialog box that opens.
 2. Specify the project name in the **Project name** text box. As you type, the name is added to the project path.
 3. Choose the download strategy in the **Deployment options** area.
 - To have PhpStorm download the files according to the settings specified in the [Options](#) dialog box on the [Deployment](#) page, click the **Default** option.
 - To customize the default settings, click the **Custom** option and [change the settings](#) in the [Create New Project: Review Deployment Options](#) dialog box that will open when you click **Next**.
4. Click **Next** when ready.

4. On the [Add Remote Server](#) page, [create a server access configuration](#) to enable interaction between PhpStorm and the remote host where the files are located.

If you already have at least one server access configuration, PhpStorm brings you to the [Specify Remote Server](#) page. Just skip this step.

5. On the [Specify Remote Server](#) page, choose the [server access configuration](#) according to which PhpStorm will interact with the remote host. Do one of the following:
 - o To use an existing configuration, click the **Use existing server** option and select the configuration from the list. The list also shows the main access settings.
 - o To define a new configuration, click the **Add new remote server** option and [create the configuration](#), as described in the previous step. The [Add Remote Server](#) dialog box will open, when you click **Next**.
6. On the [Choose Remote Path](#) page, configure the structure of the project to be created.

The page shows a tree view of folders under the [server configuration root](#).

1. Specify which remote folder will become the [project root](#). Select the relevant folder and click the  icon on the toolbar or choose **Project Root** on the context menu of the selection. PhpStorm marks the selected folder with the  icon.
2. Under the project root, specify the folders that you do not need downloaded for some reason. These can be sources that you should not update at all, media, caches, temporal files, that you actually do not need in your work.

Select the relevant folder and click  or choose **Exclude Path** on the context menu of the selection.

Note

Exclusion is applied recursively. This means that if the selected folder contains subfolders they are automatically marked as excluded as well.

3. If necessary, [configure folders under the content root](#) by specifying which of the folders to be downloaded contain only resources or internal information.

Select the folder in question and assign the desired status to it:

- To have PhpStorm consider the contents of the selected folder as unit tests, click the **Test Sources** toolbar button  or choose **Test Sources** on the context menu of the selection.
- To have PhpStorm ignore the selected directory during indexing, parsing, code completion, etc., click the **Excluded** toolbar button  or choose **Excluded** on the context menu of the selection.
- To have PhpStorm "see" the selected directory without involving it in indexing, parsing, code completion, etc., click the **Resource Root** toolbar button  or choose **Resource Root** on the context menu of the selection.

To return a folder to its regular status, select the folder in question in the list of folders under the content root, and click . Alternatively, select the folder in question in the tree and click the status icon once more.

See Also

Language and Framework-Specific Guidelines:

- [Remote Hosts](#)
- [Excluding Files and Folders from Deployment](#)
- [Configuring Folders Within a Content Root](#)

Reference:

- [New Project from Existing Files Wizard](#)
- [Deployment: Excluded Paths Tab](#)
- [Deployment](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi> 
- <http://youtrack.jetbrains.com/issues/WI> 

Comparing Deployed Files and Folders with Their Local Versions

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Correspondence between local and remote files and folders is set through [server access configuration mappings](#). It is clear, that a local file or folder can be mapped to an unlimited number of remote counterparts, while each remote file or folder is mapped to one and only one file or folder respectively.

Consequently, for each remote file or folder, PhpStorm unambiguously detects its local version to compare with, so you can have remote items compared with their local counterparts at any time. On

the contrary, remote versions of local files or folders cannot be identified uniquely, therefore PhpStorm can compare local items only with their remote versions mapped through the [default server access configuration](#).

PhpStorm provides a dedicated [Differences Viewer for Folders](#) for comparing files in remote folders and their local versions against the file size, content, or timestamp. The [Differences Viewer](#) show the contents of the local and remote directories in the [Item List](#), in the left and right panes respectively. The contents of the selected file are shown in the lower pane, with the differences being color-highlighted.

Besides exploring differences, the tool also provides interface for synchronizing the contents of folders.

In this section:

- [Accessing the Remote Host tool window](#)
- [Comparing a remote folder with its local version](#)
- [Comparing a remote file with its local version](#)
- [Comparing a local folder with its remote version](#)
- [Comparing a local file with its version on a remote host](#)
- [Comparing two folders in the Difference Viewer](#)
- [Synchronizing contents of folders](#)

To open the remote host tool window, do one of the following:

- On the main menu, choose **Tools | Deployment | Browse Remote Host**.
- On the main menu, choose **View | Tool Windows | Remote Host**.

Note

The tool window can be accessed this way after you have opened it using the **Tools | Deployment | Browse Remote Host**.

To compare a remote folder with its local version

1. [Open the Remote Host tool window](#).
2. Select the folder in question. Then choose **Tools | Deployment | Sync with Local** on the main menu or **Sync with local** on the context menu of the selection.
3. In the [Differences Viewer for Folders](#) that opens, [explore the differences](#) and [synchronize the files](#), where applicable.

To compare a remote file with its local version

1. [Open the Remote Host tool window](#).
2. Select the file in question, and then choose **Compare with local version** on the context menu of the selection.
3. In the [Differences Viewer for Files](#) dialog box, that opens, explore the differences and apply them, if necessary, using the  and  buttons.

To compare a local folder with its remote version

This action is available only if you have a default server access configuration appointed.

1. Select the folder in question in the [Project](#) tool window, and then choose **Tools | Deployment | Sync with Deployed to <default server access configuration>** on the main menu.
2. In the [Differences Viewer for Folders](#) that opens, [explore the differences](#) and [synchronize the files](#), where applicable.

To compare a local file with its remote version

This action is available only if you have a default server access configuration appointed.

1. Select the file in question in the [Project](#) tool window, and then choose **Tools | Deployment | Compare Local File with Deployed Version** on the main menu.
2. In the [Differences Viewer for Files](#) dialog box, that opens, explore the differences and apply them, if necessary, using the  and  buttons.

To compare two folders in the difference viewer, perform these general steps:

- Configure the layout of the [Items List](#). Use the [toolbar buttons](#) to narrow down or widen the set of items to show. For example, show or hide files that are present only locally or remotely, equal files, different files, files [excluded from synchronization](#), etc.
- Specify the parameter for comparison. In the **Compare by** drop-down list, select one of the possible options (contents, size, or time stamp).
- Filter the folders' contents. To do that, type filtering string in the **Filter** text field, and press **EnterEnter** to apply it. Using the asterisk * wildcard to represent any number of characters is welcome.
- To switch to another pair of folders to compare, update the fully qualified paths to them. Click the **Browse** button  next to the **Paths** read-only fields and choose the required folders in the **Select Path** dialog box, that opens.
- Explore the detected differences between files in the **Differences Pane**.

To synchronize the contents of folders

The contents of the remote folder are always shown in the right pane of the [Items List](#), the contents of its local version are always shown in the left side pane.

1. For each pair of items, in the * field specify the action to apply. Click the icon in the field until the required action is set.
 -  - the file will be uploaded, possibly overwriting the remote version.
 -  - the file will be downloaded, possibly overwriting the local version.
 -  - the files are treated identical with regard to the selected criterion of comparison. No action will be performed by default.
 -  - the files differ with regard to the selected criterion of comparison. No action will be performed by default. Explore the differences in the [Differences Pane](#) and change the intended action by clicking the icon.
 -  - the file is present only locally or remotely and will be removed.
2. Do one of the following:
 - To synchronize the currently selected item, click the **Synchronize Selected** button  on the toolbar.
 - To synchronize all the items, click the **Synchronize All** button  on the toolbar.

See Also

Procedures:

- [Remote Hosts](#)
- [Comparing Folders](#)
- [Comparing Files](#)

Reference:

- [Differences Viewer](#)
- [Differences Viewer for Folders and DB Objects](#)
- [Remote Host Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
 - <http://youtrack.jetbrains.net/issues/WI> 
-

Using Local History

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

This section describes how to use [Local History](#), which is your personal real-time version control system. Local History is independent of external version control systems and works with the directories of your project even when they are not under any VCS control.

This section describes how to:

- [View local history of a file or folder.](#)
- [View recent changes.](#)
- [Restore files from local history.](#)
- [Mark local versions with labels.](#)

See Also

Concepts:

- [Local History](#)

Reference:

- [Show History for File / Selection Dialog](#)
- [Show History for Folder Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
 - <http://youtrack.jetbrains.net/issues/WI> 
-

Putting Labels

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: 

Before embarking on a risky change to your source code, it is a good idea to mark the stable version with some meaningful label. This will help you quickly roll back to a safe version.

Labels apply to a whole project.

To add a label to a local version

1. Select a file or folder in the Project tool window, or open a file in the editor.
2. Do one of the following:
 - On the main **VCS** menu, or on the context menu of the selection, choose **Local History | Put Label**.
 - Press **Alt+BackQuoteCommand V**. and choose **Put Label** command from the VCS Operations quick list.
3. In the Put Label dialog box, type the label name.

See Also

Concepts:

- [Local History](#)

Procedures:

- [Shelving and Unshelving Changes](#)

Reference:

- [Show History for File / Selection Dialog](#)
- [Show History for Folder Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
 - <http://youtrack.jetbrains.net/issues/WI> 
-

```
ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js'; var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s);})();
```



PhpStorm 3.0.0 Web Help

Restoring a File from Local History

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Rolling back changes from the local history works same way as in the regular version control.

To roll back changes in the local history

1. [Open the Local History view](#).
2. Select the version you want to roll back to.
3. On the context menu of the selection, choose **Revert**.



[Click thumbnail to view larger image.](#)

See Also

Concepts:

- [Local History](#)

Reference:

- [Show History for File / Selection Dialog](#)
- [Show History for Folder Dialog](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Viewing Local History of Source Code

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Besides [file history](#), you can track local changes to a [selected block of source code](#). This history shows only those changes that affect the code fragment.

To view local history of a source code block

1. In the editor, select a fragment of source code.
2. Do one of the following:
 - o On the main VCS menu, or on the context menu of the selection, choose **Local History | Show History for Selection**.
 - o Press **Alt+BackQuoteCommand** ∇ , and choose the desired command from the VCS Operations quick list.

See Also

Concepts:

- [Local History](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

```
ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js'; var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s);})();
```



PhpStorm 3.0.0 Web Help

Viewing Local History of a File or Folder

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Local History makes it possible to view changes made to a certain file or a whole directory. Each entry in the Local History dialog box is displayed with its time stamp, action and optional label. So doing, the local history for a file includes all changes that affect both the selected file and the whole project; local history for a folder shows changes to the source code tree in general. You can explore changes, selecting the respective row in the Local History dialog box.

To view local history

1. Select a folder or file in the Project tool window, or open a file in the editor.
2. Do one of the following:
 - o On the main VCS menu, or on the context menu of the selection, choose **Local History | Show History**.
 - o Press **Alt+BackQuoteCommand V**, and choose the desired command from the VCS Operations quick list.
 - o Use [View Recent Changes](#) that shows a summary of recent changes in a single pop-up list. Clicking an entry in this list shows the respective Local History.
3. Use the Local History view to compare local versions and revert changes.



Click thumbnail to view larger image.

See Also

Concepts:

- [Local History](#)

Procedures:

- [Differences Viewer](#)
- [Differences Viewer for Folders and DB Objects](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Viewing Recent Changes

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

PhpStorm enables you to view the summary of the recent changes to a project. From the **Recent Changes** pop-up window, you can browse through the history of changes, navigate to a particular change, compare versions, and roll changes back.

To view recent changes

1. On the main menu, choose **View | Recent Changes**, or press **Alt+Shift+Command Shift C**.
2. In the **Recent Changes** pop-up window, select the change you are interested in:

Recent Changes	
External change	21.01.09 16:38
Deleting	21.01.09 16:21
External change	21.01.09 16:12
Renaming file gendoc.html to package.html	21.01.09 16:11
External change	21.01.09 16:04
Create HTML	21.01.09 16:02

3. In the dialog box that opens, review differences and discard changes, if necessary.

See Also

Concepts:

- [Local History](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Version Control with PhpStorm

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

In this part you can find descriptions of procedures that are common to all supported version control systems, or specific to certain VCS integrations:

- [Accessing VCS Operations](#)
- [Enabling Version Control](#)
- [Configuring Version Control Options](#)
- [Common Version Control Procedures](#)
- [VCS-Specific Procedures](#)

See Also

Concepts:

- [Version Control with PhpStorm](#)

Reference:

- [Version Control Reference](#)
- [Version Control](#)
- [Changes Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Accessing VCS Operations

Previous | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

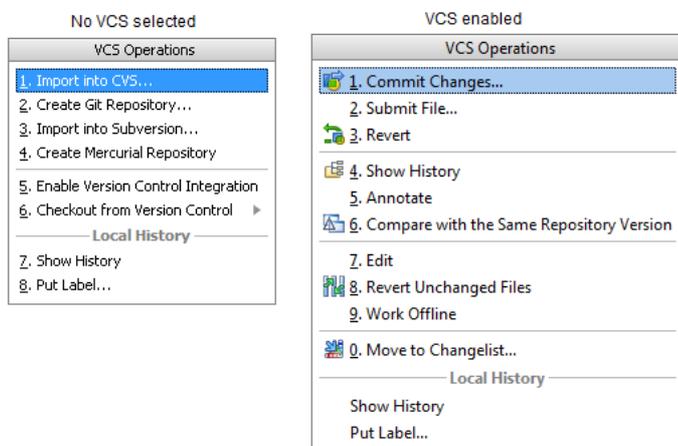
Tip

Use the VCS Operations Pop-up to quickly invoke most required commands.

To quickly invoke a VCS command using VCS Operations Pop-up

1. Open VCS Operations pop-up, in one of the following ways:
 - o On the main menu, choose VCS | VCS Operations Pop-up.
 - o Press `Alt+BackQuoteCommand V`.

Note that the composition of VCS commands available on the pop-up menu, depends on the specific VCS.



2. Choose command from the VCS Operations list. To do that, perform one of the following actions:
 - o Click the desired command in the list.
 - o Use up and down arrow keys to select the desired command, and then press `Editor.EnterEditor.Enter`
 - o Press number key that corresponds to the desired command in the list.

See Also

Reference:

- [Alt](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Enabling Version Control

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm supports version control integration at two levels:

- At the IDE level, VCS integration is provided through a set of [bundled plugins](#) that are enabled by default.
- At the project level, VCS integration is enabled by associating project folders with one or several version control systems.

This section describes how to:

- [Associate a project root](#) with version control system.
- [Associate a directory](#) with version control system.
- [Change the associated VCS](#) for the project root or directory.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Version Control with PhpStorm](#)

Reference:

- [Version Control](#)
- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Associating a Project Root with Version Control System

Previous | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

PhpStorm suggests a command that allows one-click enabling of version control integration. So doing, the selected VCS becomes associated with the project root. To learn how to associate the other content roots with the different version control systems, refer to the section [Associating a Directory with a Specific Version Control System](#).

To assign a version control system to a project root

1. Choose **VCS | Enable Version Control Integration**, or press **Alt+BackQuoteCommand V**, and choose **Enable Version Control Integration**. The [Enable Version Control Integration](#) dialog box opens.
2. In the **Please select version control system to make your <Project Root>** under drop-down list, select the version control system that you want to assign to your [project root](#).

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Version Control with PhpStorm](#)

Reference:

- [Enable Version Control Integration Dialog](#)
- [Version Control](#)
- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Associating a Directory with a Specific Version Control System

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm supports version control in a project [by directories](#). This means that each directory can be versioned under its own version control system.

To associate a directory with a version control system

1. Open the [Version Control](#) dialog box. The dialog box displays a table of directories and the version control systems associated with them.

Note

By default, the table contains only one row for the **<Project Root>**, with no version control system assigned to it.

2. Click the **Add** button. The **Add VCS Directory Mapping** dialog box opens.
3. Type the path to the desired directory, or click  to the right of the **Directory** field and navigate to the desired directory.
4. From the **VCS** drop-down list, select the version control system to be used for controlling files in this directory.

Tip

The drop-down list displays only the version control systems that are [installed and enabled](#) in the PhpStorm plugin repository.

Tip

Optionally, click the **Configure VCS** button. The **Version Control Configurations** dialog box opens where you can configure settings for the selected version control system (see the corresponding configuration settings in the [Version Control](#) dialog box reference for details).

5. Click **OK** to save the mapping and return to the **Version Control** dialog box.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Reference:

- [Version Control](#)
- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Changing VCS Associations

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

To change the version control system associated with a directory

1. Open the [Version Control](#) dialog box. The dialog box displays a table of directories with associated version control systems.
2. In the table, locate the row that corresponds to the directory which you want to put under another version control system.
3. Click the VCS column. From the drop-down list that appears, select a new version control system.

Tip

Optionally, click the **Configure VCS** button. The **Version Control Configurations** dialog box opens where you can configure settings for the selected version control system (see the corresponding configuration settings in the [Version Control](#) dialog box reference for details).

4. Click **OK** to save the new mapping and close the **Version Control** dialog box.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Reference:

- [Version Control](#)
- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Configuring Version Control Options

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Configuring version control options involves:

- [Configuring General VCS Settings](#)
- [Configuring VCS-Specific Settings](#)

See Also

Concepts:

- [Version Control with PhpStorm](#)

Reference:

- [Version Control Reference](#)
- [Version Control](#)
- [Ignored Files](#)
- [Ignore Unversioned Files](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Configuring General VCS Settings

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

General version control settings apply to all version control systems integrated with PhpStorm. General settings are specified on the [Version Control](#) page of the [Settings](#) dialog box, and include defining actions that require confirmation, background operations, ignored files, issue navigation, and depth of history.

To configure general version control settings, follow these general steps

1. Open the [Settings](#) dialog box and click **Project Settings**, then click the **Version Control** node.
2. Specify which version control related actions should require [confirmation](#).

3. Specify which operations should be performed in the [background](#).
4. Create a list of [files to be ignored](#) by version control systems.
5. Configure [history cache handling](#).
6. Define [issue navigation](#) rules to switch from check-in comments to corresponding issues in a bug tracking system.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Reference:

- [Version Control Reference](#)
- [Version Control](#)
- [Ignored Files](#)
- [Confirmation](#)
- [Background](#)
- [Issue Navigation](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Specifying Actions to Confirm

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

You can define that certain version control related activities should be performed only upon confirmation from your side. The activities for which confirmation is required are specified in the [Confirmation](#) dialog box.

To specify which actions should require confirmation

1. Below the [Version Control](#) node, click [Confirmation](#). The [Confirmation](#) dialog box opens.
2. In the [Files Creation/Deletion](#) area, define how files created or deleted in PhpStorm should be added to or removed from a version control system. The available options are:
 - Show options before adding to version control
 - Add silently
 - Do not add
3. In the [Display options dialog when these commands are invoked](#) area, specify the commands, for which you want PhpStorm to display the [Options](#) dialog box. The available options are:
 - Check Out
 - Status
 - Get Latest Version
 - Update
 - Undo Check Out
4. Configure additional settings for working with changelists by selecting or clearing the corresponding check boxes:
 - Specify whether the read-only state of files should require explicit cancellation.
 - Specify whether meaningful comments on committing files to the repository should be required.
 - Specify whether uncommitted changes should be moved to another changelist.
 - Specify whether and how to create a changelist if the commit operation fails.

See Also

Concepts:

- [Version Control with PhpStorm](#)
- [Configuring Version Control Options](#)

Reference:

- [Version Control Reference](#)
- [Confirmation](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Specifying Actions to Run in the Background

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

You can enable background execution of certain version control related activities. These activities are specified in the [Background](#) dialog box.

To specify the operations to run in the background

1. Below the [Version Control](#) node, click [Background](#). The [Background](#) dialog box opens.

2. Enable background execution of the necessary actions by selecting the corresponding check boxes. Background execution can be set for the following actions:
 - o Update
 - o Commit
 - o Checkout
 - o Edit/Checkout
 - o Add/Remove
 - o Revert
 - o [History Cache Handling](#)
 - o Detecting "changed on server" conflicts

See Also

- Concepts:
- [Version Control with PhpStorm](#)

- Procedures:
- [Configuring Version Control Options](#)
 - [Configuring History Cache Handling](#)

- Reference:
- [Version Control Reference](#)
 - [Background](#)

- Web Resources:
- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>

Configuring Ignored Files

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Sometimes it is helpful to leave files of certain types unversioned. These can be VCS administration files, artifacts of utilities, backup copies etc. You can create a global ignore list that will be stored in the workspace file and applied to all supported version control systems.

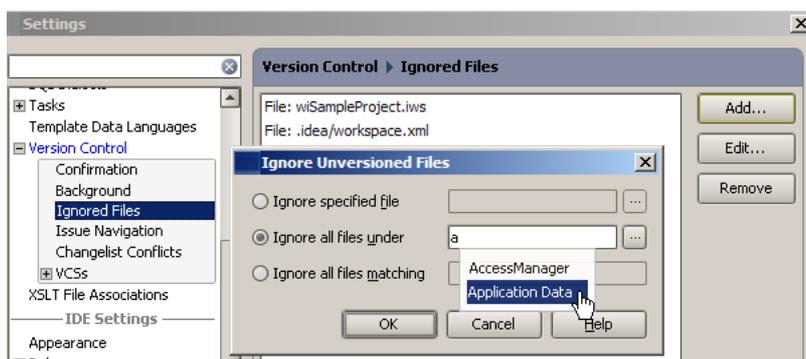
Tip

If the version control system that you are using has its own ignore facilities, use the corresponding native command provided by the version control integration.

Use the [Ignored Files](#) dialog box to specify which unversioned files a version control system should exclude from processing.

To define a list of ignored files

1. Under the [Version Control](#) node, of the [Settings](#) dialog box, click [Ignored Files](#). The [Ignored Files](#) dialog box opens.



2. Click the [Add](#) button to create a new entry or select an existing entry and click the [Edit](#) button. The [Ignore Unversioned Files](#) dialog box opens.
3. Explicitly specify the files/directories to be ignored or define file name patterns. Do one of the following:
 - o Select the [Ignore specified file](#) option and type the fully qualified name of the file to be ignored.
 - o Select the [Ignore all files under](#) option and type the path to the directory whose contents should be ignored.
 - o Select the [Ignore all files matching](#) option and type the pattern that defines the names of files to be ignored.

Tip

When typing a file name or a path to a directory, use [basic code completion](#): start typing, press `Ctrl+SpaceCommand Space`, and select the desired completion from the suggestion list.

4. Create as many entries as you need and close the dialog box.

Tip

You can also add files to ignore list on-the-fly. A new file under the [Unversioned Files](#) change list, has [Ignore](#) command on its context menu.

See Also

Concepts:

- [Version Control with PhpStorm](#)
- [Directory-Based Versioning Model](#)

Reference:

- [Version Control Reference](#)
- [Version Control](#)
- [Ignored Files](#)
- [Ignore Unversioned Files](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
 - <http://youtrack.jetbrains.net/issues/WI> 
-

Configuring History Cache Handling

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

You can configure handling of the history cache in the [Background](#) dialog box.

To configure history cache handling

1. Below the [Version Control](#) node, click **Background**. The [Background](#) dialog box opens.
2. To have PhpStorm handle history cache handling in the background, select the **Enable background processes**
3. Set the cache scope. Depending on the version control system you are using, do one of the following:
 - Specify the maximum number of changelists to be stored in the cache.
 - Specify the maximum number of days for a changelist to be stored in the cache.
4. Specify whether and how often (in minutes) you want your version control system to refresh the cache.

See Also

Concepts:

- [Version Control with PhpStorm](#)
- [Configuring Version Control Options](#)

Reference:

- [Version Control Reference](#)
- [Background](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
 - <http://youtrack.jetbrains.net/issues/WI> 
-

Configuring VCS-Specific Settings

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Certain settings are applicable to the version control systems assigned to the [whole project, or its directories](#). The others are related to the selected version control systems. Use the respective VCS pages under the **VCSs** node of the [Settings](#) dialog box to define VCS-specific settings.

To configure VCS-specific settings, follow these general steps

1. Open the [Settings](#) dialog box and click **Project Settings**, then click the **Version Control** node.
2. Click a page that corresponds to the VCS to be configured.
3. Set up options as required. For detailed information, see [VCS-specific](#) pages of the Version Control settings.

See Also

Concepts:

- [Version Control with PhpStorm](#) 

Procedures:

- [VCS-Specific Procedures](#)

Reference:

- [Version Control Reference](#)
- [VCSs](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Common Version Control Procedures

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm makes it possible to interact with the different version control systems (VCS). Regardless of which VCS you use, certain basic operations are common to all (or almost all) of them. This section covers these basic operations and explains how to perform them from within PhpStorm:

- [Adding Files to Version Control](#)
- [Browsing Contents of the Repository](#)
- [Getting Local Working Copy of the Repository](#)
- [Changing Read-Only Status of Files](#)
- [Checking in Files](#)
- [Copying, Renaming and Moving Files](#)
- [Deleting Files from the Repository](#)
- [Handling Differences](#)
- [Handling Issues](#)
- [Managing Changelists](#)
- [Refreshing Status](#)
- [Reverting Local Changes](#)
- [Reverting to a Previous Version](#)
- [Shelving and Unshelving Changes](#)
- [Updating Local Information](#)
- [Using Patches](#)
- [Viewing Changes Information](#)
- [Accessing the Authentication to Server Dialog](#)

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [VCS-Specific Procedures](#)

Reference:

- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Adding Files to Version Control

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

If a new file is created with PhpStorm in a directory that is already under version control, it automatically adds to the active changelist with the status Added. All you have to do, is to commit this change.

PhpStorm's behavior on adding files is configurable in the [General Settings tab](#) of the Version Control dialog.

If a new file is created outside of PhpStorm, or silent adding is disabled, you have to explicitly add it to the version control.

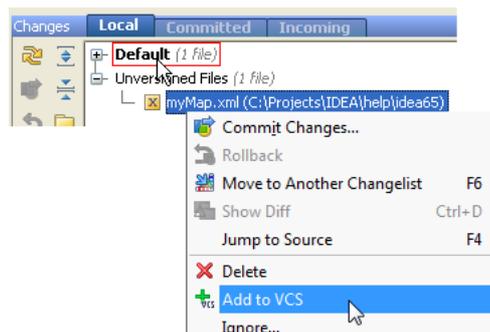
Another approach is VCS-specific. You can import an entire directory to your [CVS](#) or [Subversion](#) repository, provided that you have the corresponding access rights. This action is helpful for putting a whole project under version control.

To explicitly add a file to version control

1. Select file in the Project tool window.
2. On the main **Version Control** menu or on the context menu of the selected file, choose **<VCS> | Add**.

Tip

Alternatively, use the Changes tool window. Select the desired files in the Unversioned files changelist, and choose **Add to VCS** on the context menu, or just drag it to the target changelist.



3. Select the added file in the changelist and [check in \(commit\) changes](#).

See Also

Procedures:

- [Importing a Local Directory to CVS Repository](#)
- [Importing a Local Directory to Subversion Repository](#)
- [Sharing Directory](#)

Reference:

- [Version Control Reference](#)
- [Version Control](#)
- [Changes Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Browsing Contents of the Repository

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Prior to checking files out, you can browse the contents of the repository. This action is available for CVS and SVN integrations:

- [Browsing CVS Repository](#)
- [Browsing Subversion Repository](#)

Note

Browsing contents of Subversion or CVS repository is always available, even though the respective VCS is not enabled in project. All you need is a valid user account.

See Also

Procedures:

- [Browsing Changes](#)
- [Working with Annotations](#)
- [Viewing Changes History for a File or Selection](#)

Reference:

- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Getting Local Working Copy of the Repository

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

As soon as [version control support is enabled](#) in PhpStorm, you can retrieve the data from the repository. Depending on your purpose and workflow, choose one of the following approaches:

- [Check out](#) the repository sources to a [content root](#) of an [existing project](#). Then set up version control in the project, if it has not been done before, and put the downloaded sources under control of the VCS used.
- [Check out](#) the repository sources to a location of your choice and have PhpStorm [create a project around them](#). This does not require any extra steps from your side. PhpStorm suggests to set up a project based on the downloaded sources itself, as soon as the check-out is completed. Upon your consent, the [New Project from Existing Code Wizard](#) starts. When the project is created, set up version control in it.

Warning

This approach is not available for Perforce integration.

The check-out procedure depends on the type of VCS you use. Refer to the following sections for details:

- [Retrieving Files from a CVS Repository](#)
- [Retrieving Files from an SVN Repository](#)
- [Cloning a Remote Git Repository](#)
- [Cloning a Remote Mercurial Repository](#)
- [Using Perforce Integration](#)

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Version Control with PhpStorm](#)
- [Changing Read-Only Status of Files](#)

Reference:

- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Changing Read-Only Status of Files

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Different version control systems have different semantics for the action of removing read-only status from a file so that you can edit it. Some systems never put read-only status on local files at all unless specifically configured to do so (i.e. the system is configured to support the file locking model).

Different version control systems use different names for this action: `check out`, `edit`, `Open for Edit`, or `Get`. Regardless of the terminology used by your VCS, if it sets read-only status on your local working files, you can remove read-only status and make files writeable from within PhpStorm, which will also take care of setting a lock on the server, or take whatever other action is required by the VCS, via the respective VCS integration.

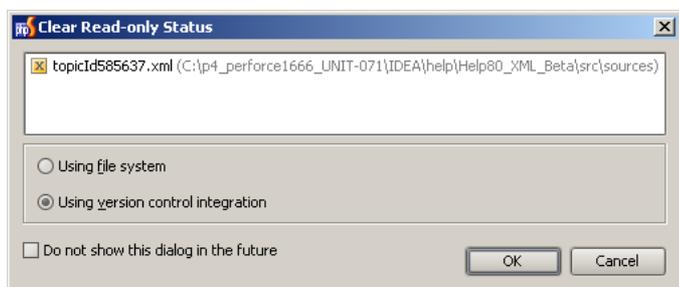
This behavior is configurable in the [General Settings tab](#) of the Version Control dialog.

To enable explicit removal of read-only status

- In the [Confirmation page](#) of the [Version Control settings](#), check the option Show "Clear Read-Only Status" Dialog.

For example, removing read-only status in Perforce looks as follows

- With the Show "Clear Read-Only Status" Dialog option enabled, an attempt to edit a file brings up the respective dialog box.



If you click the radio button **Using file system**, the file will not be added to the default changelist. If you click radio button **Using version control integration**, the file is added to the default changelist.

Tip

You can make a file writable using the lock icon  in the status bar. Open the desired file in the editor, and double-click the lock, as shown below:



See Also

Procedures:

- [Toggling Writable Status](#)

Reference:

- [Version Control Reference](#)
- [Version Control](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

2.0+

Checking in Files

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: 

Different version control systems have different semantics for the action of uploading changed files to the repository. Two common terms are `check in` and `commit`.

Note

In those version control systems, for example, [Git](#), that distinguish between local and remote repositories, the term `commit` denotes uploading changes to a `local` repository. Uploading changes to a remote repository is referred to as `push`.

Regardless of the terminology, you can perform this operation with the VCS configured for a directory from within PhpStorm.

To check in (commit) changed files, perform these general steps

- Open the [Changes](#) tool window (Alt+9Meta 9).
- Select one or more files you want to check in (commit) to version control.
- Open the [Commit Changes](#) dialog box by doing one of the following:
 - On the [Changes](#) tool window toolbar, click the [Commit Changes](#) button .
 - Press Ctrl+KCommand K.
 - On the main menu, choose **VCS | Commit Changes**.
- Review the changes to be committed in the [Details](#) pane. To do that, unfold the [Details](#) pane if it is hidden, and select the file in question in the [Changed Files](#) area.

The [Details](#) pane shows the base version and the local copy of the selected file. Examine the details of each change:

- To move to the next updated piece of code, click the [Next Change](#) button .
- To return to the previous updated code fragment, click the [Previous Change](#) button .
- To expand or narrow the context of an updated code fragment, position the cursor at the change in question, click the [More/Less Lines](#) button , and then specify the number of lines to be shown above and below the current code fragment.
- Add a commit comment. As you type, PhpStorm checks the spelling and highlights words in question.

Note

This functionality is available if the [Spelling code inspection](#) is enabled.

- Specify which actions should be performed on the files before and after submitting them to the repository.
- Click the [Submit/Commit](#) button to launch the [Check-in Changes](#) operations.
- To have the changes immediately pushed to your Git or Mercurial repository, hover the mouse pointer over the [Submit/Commit](#) button and select **Commit and Push** on the context menu.

Tip

Alternatively, use the [Submit/Commit](#) drop-down list to select the **Commit and Push** item.

- To save the changes as a patch in a text file, hover the mouse pointer over the [Submit/Commit](#) button and select **Create Patch** on the context menu.

Tip

Alternatively, use the [Submit/Commit](#) drop-down list to select the **Create Patch** item.

In the [Create Patch](#) dialog box, that opens, [configure the patch creation](#).

- If any error occurs when trying to commit, PhpStorm displays an error message. For example, you might have changed a file that has been already edited by another team member, or you might run into a branch conflict. In these cases, you need to [merge edits](#), or [update your local copy](#). The error messages are VCS-specific.

Tip

Users of [JetBrains TeamCity](#)  can obtain the TeamCity plugin for PhpStorm. Among the features of this plugin is Remote Run, which enables you to create a special *personal* build that does not affect the *real* build. Your changes are built and run through your test suite. If all tests are passed, your changes are automatically committed to version control.

See Also

Procedures:

- [Resolving Commit Errors](#)

Reference:

- [Version Control Reference](#)
- [Commit Changes Dialog](#)
- [Changes Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Copying, Renaming and Moving Files

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

There is no automatic renaming or moving files in the version control systems. Working with the clients, you have to manually copy a file to a new location, change its name, add to VCS and remove the old file. PhpStorm's refactoring features make it possible to easily move and rename files under version control, performing all actions involved in these functions.

The resulting files are specially marked in the [Changes](#) tool window, as shown in the image below:



To copy, move or rename a file under version control

1. Select the desired file in the Project tool window, and perform refactoring procedures, as described in the sections:
 - o [Copy](#)
 - o [Move](#)
 - o [Rename](#)
 All added and deleted files are placed to a changelist.
2. [Commit changes](#) to the repository.

Tip

You can revert refactorings, using the **Undo** command.

See Also

Procedures:

- [Copy/Clone](#)
- [Move Refactorings](#)
- [Rename Refactorings](#)

Reference:

- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

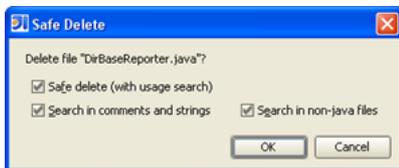
Deleting Files from the Repository

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

If you delete a file under version control, it still exists in the repository until deletion is committed. A deleted file is placed to the active changelist, and is displayed in grey font.

To delete a file

1. Select a file in any navigation view, and press **Delete**, or choose **Delete** on the context menu.
2. In the dialog that opens, you can opt to delete file without search for usages, or perform safe delete, to make sure that you are deleting an unused file, by checking **Safe delete** option. In this case, specify the refactoring options.



The encountered usages are reported in the Usages Detected dialog box. You can view and correct these usages in the Safe Delete tab of the Find tool window. The deleted files are added to changelist.

3. [Commit changes](#) to the repository.

See Also

Procedures:

- [Checking in Files](#)
- [Safe Delete](#)

Reference:

- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Handling Differences

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm enables you to examine the differences between two revisions of a file, or between the current local copy and any repository version. Differences display in the [Differences viewer](#). This window enables you to compare files and versions, navigate and search through the changes, copy and edit source code. Use the legend of the viewer to recognize modified, deleted and added lines of code.

This section describes how to:

- [Compare file versions](#).

- [Integrate contents of different versions into a file.](#)
- [Resolve conflicts between versions.](#)
- [Integrate your local copy of a project into a revision in the repository.](#)

See Also

Reference:

- [Differences Viewer](#)
- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Comparing File Versions

Previous | [Next](#) | [See Also](#) | [Comments](#)

VCS-specific integrations enable you to compare your local copy of a file with the versions stored in the repository. PhpStorm suggests several options:

- [Compare your local copy with the repository version, to which you have last synchronized](#)
- If somebody else has committed changes since your last update, [compare your local copy with newest version](#)
- [Compare your local version with any repository version of a file.](#)

For some VCS, it is possible to compare with a branch version. The differences display in the [Differences viewer](#).

To compare with a repository version, to which you have last synchronized

1. Select a file in the Project tool window, or open it in the editor.
2. Do one of the following:
 - On the main VCS menu, or on the context menu of a file, choose **<VCS> | Compare with the same repository version**.
 - Select a file in the Local tab of the Changes tool window, and choose **Show Diff** on the context menu.

To compare with the latest repository version

1. Select a file, or open it in the editor.
2. On the main VCS menu, or on the context menu of a file, choose **<VCS> | Compare with the latest repository version**.

To compare with the specified version of a file

1. Select a file, or open it in the editor.
2. On the main VCS menu, or on the context menu of a file, choose **<VCS> | Compare with**.
3. In the File Revision pop-up window, click the version to compare:

File Revisions			
160077	03.05.07 13:00		robert
160184	03.05.07 13:37		robert
159873	28.04.07 14:34		robert

Tip

Alternatively, use the History view of a file. Select the desired version, and choose **Compare with Local** on its context menu, or click the  button on the toolbar.

Tip

You can explore changes to binary files same way, as you do it for textual files. For example, use this feature to watch the changes made to images.

See Also

Procedures:

- [Using Change Markers to View and Navigate Through Changes in the Editor](#)

Reference:

- [Differences Viewer](#)
- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Integrating Differences

Previous | [Next](#) | [See Also](#) | [Comments](#)

You might need to integrate contents of the different versions of a file, for example integrate changes of a certain branch into your local copy.



To integrate changes into your working copy

1. Select the desired file and [compare it with a version](#).
2. In the Differences Viewer, use the *chevron* symbol for any block of lines that existed in the read-only copy and were changed or deleted in the local copy; and x symbol for the lines that were added in the local copy. Click these symbols to apply changes to the current local version of the file.

See Also

Reference:

- [Differences Viewer](#)
- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Integrating Project

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

You might need to integrate your local version of a project into a certain revision of that project in the repository. For this purpose PhpStorm provides Integrate Project command.

Integrating project is only available for the projects that contain directories associated with Perforce or Subversion. If neither of these two VCS is involved in project, the Integrate Project command is disabled in the menu.

To integrate different sources into one project

1. On the main menu, choose **VCS | Integrate Project**. The Integrate dialog appears.
2. Select [Perforce](#) or [Subversion](#) tab, and configure merge options as required. Refer to the respective procedural topics for details.

See Also

Procedures:

- [Integrating Perforce Files](#)
- [Integrating SVN Projects or Directories](#)

Reference:

- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Resolving Conflicts

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

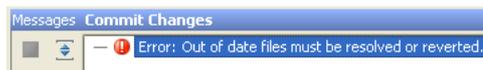
Working in a team, you might come across a situation, when somebody commits changes to a file you are currently working on. Conflicts can also arise on merging branches, applying patches or unshelving changes. If the changes are not overlapping, the files are merged automatically, but when there are changes to the same lines, you will need to resolve such conflict yourself.

When you try to edit a file that has a newer version on the server, PhpStorm informs you about that, showing a message pop-up window in the editor:

Outdated version. Modified by rayshade on 24.03.2009 15:33:18: New class added [Show Diff](#) [Update Project](#)

In this case you should update your local version prior to changing the file, or merge changes.

If you try to commit your changes that are in conflict with the repository version, an error is reported in the **Messages** tool window:

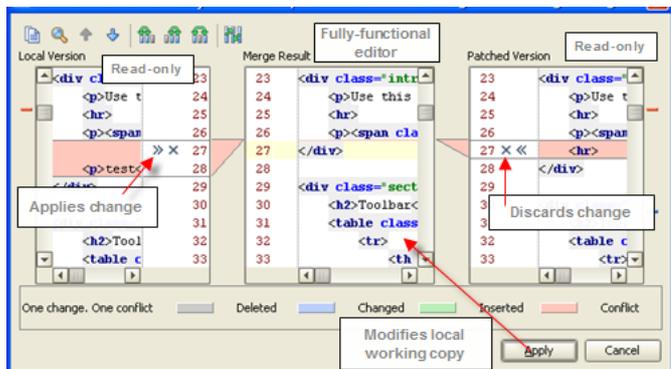


Depending on the behavior chosen from the **Create changelist on failed commit** drop-down list in the [Confirmation](#) page of the [Settings](#) dialog box, PhpStorm does one of the following:

- Creates a **Failed Commit** changelist, moves the conflicting files to it automatically, and highlights the names of the files red. This behavior takes place if the **Yes** option is chosen.
- Leaves the conflicting files in the changelist where they were but highlights them red. This behavior takes place if the **No** option is chosen.
- Displays a dialog box, where you can choose to move the conflicting files to a new changelist or leave them in the original one. In either case, the conflicting files will be highlighted red. This behavior takes place if the **Ask** option is chosen in the [Confirmation](#) page.

If you synchronize a file, that already has local changes, with a newer repository version, committed by somebody else, a conflict occurs. The conflicting file gets status *Merged with conflicts*. The file remains in the same changelist in the [Local](#) tab of the [Changes](#) tool window but its name is highlighted red. If the file is currently opened in the editor, the file name on the tab header is also highlighted red.

PhpStorm suggests a tool for resolving conflicts locally. This tool consists of three panes. The left pane shows the read-only local copy; the right pane shows the read-only version checked in to the repository. The central pane shows a fully-functional editor with the results merge and conflict resolving. Initially, the contents of this pane is the same as the local copy.



To resolve conflicts, follow these general steps

1. Select the conflicting file in the Changes, Update Version Control, or Project tool window, or open it in the editor.
2. On the context menu, under the node of your VCS integration, choose the VCS-specific merge command.
3. Perform edit in the merge tool to apply or discard changes, as described in the procedures below.

To apply or discard a change

1. In the left or right pane select the change to be merged. Use arrow and/or find buttons on the toolbar to navigate to the desired location.
2. Accept or reject changes, clicking the *chevron* or *X* button respectively.
3. Click **Apply**.

To apply all non-conflicting changes

- On the toolbar of the merge tool, click . It is useful to automatically accept any changes that are not in conflict, and spend your time working on conflicting changes.

See Also

Procedures:

- [Handling Differences](#)
- [Resolving Conflicts with Perforce Integration](#)
- [Resolving Text Conflicts](#)

Reference:

- [Changelist Conflicts](#)
- [Local Tab](#)
- [Changes Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Handling Issues

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm enables you to connect the checkin comments with the bug tracker or any issues data base.

In this section:

- [Creating Issue Patterns](#)
- [Navigating to Issues](#)

See Also

Reference:

- [Version Control Reference](#)
- [Version Control](#)
- [Changes Tool Window](#)
- [Regular Expression Syntax Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Creating Issue Patterns

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

To connect the changes information with an issue tracking database, you need to inform PhpStorm how to recognize a reference to an issue in the check-in comment. For this purpose, use the [Issue](#)

[Navigation](#) dialog box.

To define issue patterns and navigation links

1. Below the [Version Control](#) node, click [Issue Navigation](#). The [Issue Navigation](#) dialog box opens.
2. Click the [Add](#) button to create a new entry or select an existing entry and click the [Edit](#) button. The [Add Issue Navigation Link](#) dialog box opens.
3. In the [Add Issue Navigation Link](#) dialog box, specify the following:
 - [regular expression](#) to define the issue pattern
 - Replacement expression to define the link to the corresponding issue.
4. If you are using JIRA or [our bug tracking system YouTrack](#), click [Add JIRA pattern](#) or [Add YouTrack Pattern](#) respectively, and type the URL to the installation of bug tracking system in question.
The regular expression for such pattern is added automatically.

See Also

Concepts:

- [Version Control with PhpStorm](#)
- [Configuring Version Control Options](#)

Reference:

- [Version Control Reference](#)
- [Version Control](#)
- [Changes Tool Window](#)
- [Regular Expression Syntax Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Navigating to Issues

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Once the issue patterns and links to the issues databases are defined, you can navigate from a change in the Changes tool window to a CR in your bug tracker. To make it possible, your commit messages should contain matches with the specified patterns.

When such matches are encountered in the commit messages, they show up as hyperlinks in the Changes and Version Control tool windows.

To navigate from a change to the related issue

1. Open one of the following views:
 - Committed or Incoming tabs of the Changes tool window.
 - Browse Changes results tab.
 - History view.
2. Click the desired change, marked as a hyperlink to the related issue.

See Also

Reference:

- [Version Control Reference](#)
- [Changes Tool Window](#)
- [History Tab](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Managing Changelists

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

This section describes how to:

- [Create, delete](#) and [rename](#) changelists.
- [Assign active changelists](#).
- [Group items in a changelist](#).
- [Move files between changelists](#).
- [Jump from an item in a changelist to the corresponding source code in the editor](#).

See Also

Concepts:

- [Changelist](#)

Reference:

- [Version Control Reference](#)

- [Changes Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Assigning an Active Changelist

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Active changelist is the one to which the changed files are added automatically. The name of the active changelist is highlighted in bold font.

To assign an active changelist

1. Select a changelist in the Changes tool window.
2. On the context menu of the selected changelist, click **Set Active Changelist**.

See Also

Concepts:

- [Changelist](#)

Reference:

- [Version Control Reference](#)
- [Changes Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Creating a New Changelist

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: 

To create a new changelist

1. In the toolbar of the Changes tool window, click  button.

Tip

You can also use one of these alternatives:

- Right-click anywhere in the Local tab of the Changes tool window and choose **New Changelist** on the context menu.
 - Press `Alt+InsertCommand N`.
2. In the New Changelist dialog, specify the name of the new changelist, and optional comment.
 3. Click **OK**.

See Also

Concepts:

- [Changelist](#)

Reference:

- [Version Control Reference](#)
- [Changes Tool Window](#)
- [New Changelist Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Deleting a Changelist

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: 

When a [changelist](#) is deleted, all changes are moved to the active changelist.

To delete a changelist

1. In the [Changes tool window](#), select a changelist to be deleted.
2. In the toolbar of the Changes tool window, click . Alternatively, right-click the changelist node and choose **Delete Changelist** on the context menu, or just press `DeleteDelete` key.
3. If the changelist is not empty, you are prompted to confirm deletion and move uncommitted items to the active changelist. If you attempt to delete an active changelist, you are prompted to specify another

Note

If Perforce is used for a certain directory, deleting the default changelist is not allowed.

See Also

Concepts:

- [Changelist](#)

Reference:

- [Version Control Reference](#)
- [Changes Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Grouping Changelist Items by Folder

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts:

Within each node of the Changes tool window, you can display the modified files as a flat list, or as a directory tree.

To toggle grouping items by directories, do one of the following

- Click the Folder icon.
- Use `Ctrl+Meta P` keyboard shortcut.

See Also

Concepts:

- [Changelist](#)

Reference:

- [Version Control Reference](#)
- [Changes Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Moving an Opened File to Another Changelist

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

To move the currently opened file to another changelist

1. On the context menu of the active editor, choose **Move to Another Changelist**.
2. In the **Choose Changelist** dialog box, specify the changelist to move the file to:
 - If the target changelist exists, click the **Existing Changelist** option and select the desired changelist from the drop-down list.
 - To create a changelist, click the **New Changelist** option, type the name of the new changelist, and optionally provide a description.

See Also

Concepts:

- [Changelist](#)

Procedures:

- [Moving an Opened File to Another Changelist](#)
- [Moving Items Between Changelists in the Changes Tool Window](#)

Reference:

- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Moving Items Between Changelists in the Changes Tool Window

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts:

To move items between changelists in the changes tool window

1. In the [Changes](#) tool window, select one or more desired items in a changelist. Use the `Ctrl` and `Shift` keys for multiple selection.
2. Choose **Move to Another Changelist** on the context menu of the selection.

Tip

You can also use one of these alternatives:

- Click the **Move to Another Changelist** button  on the toolbar of the tool window.
 - Press `F6`.
 - Drag the selected items to the target changelist.
3. In the **Choose Changelist** dialog box, specify the changelist to move the selected items to:
 - If the target changelist exists, click the **Existing Changelist** option and select the desired changelist from the drop-down list.
 - To create a changelist, click the **New Changelist** option, type the name of the new changelist, and optionally provide a description.

See Also

Concepts:

- [Changelist](#)

Procedures:

- [Moving an Opened File to Another Changelist](#)
- [Moving Items Between Changelists in the Changes Tool Window](#)

Reference:

- [Version Control Reference](#)
- [Changes Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Navigating to Source Code

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

To navigate from an entry in the changelist to the source code

1. In the Changes Tool Window, expand a changelist and select the desired entry.
2. On the context menu of the selection, choose **Jump to Source**, or press `F4`.

See Also

Concepts:

- [Changelist](#)

Reference:

- [Version Control Reference](#)
- [Changes Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Renaming a Changelist

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

To rename a changelist

1. Select a changelist in the Changes Tool Window.
2. On the context menu, choose **Rename Changelist**.
3. In the Edit Changelist dialog, specify the new name and optional description, and click **OK**.

Tip

Alternatively, use the `Shift+F6` keyboard shortcut.

See Also

Concepts:

- [Changelist](#)

Reference:

- [Version Control Reference](#)
- [Changes Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Refreshing Status

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

This feature is helpful if a server, that holds the sources of the project files, is down. The statuses of files and directories become irrelevant, and you are unable to work with the local copy of the project. To avoid such situation, you need to refresh the status of your source files.

To refresh status of files in your project, do one of the following

- On the main menu, choose **VCS | Refresh File Status**.
- In the Changes tool window, press `Ctrl+F5` / `Command F5`.
- In the toolbar of the Changes tool window, click 

See Also

Concepts:

- [Version Control with PhpStorm](#)

Reference:

- [Version Control Reference](#)
- [Changes Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Reverting Local Changes

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

When you modify, add, or delete files, which are under version control, you are always able to revert such changes, rolling back the file's contents to what it was before the last successful update, check out, or commit.

Note

The exact name of the command (revert or roll back), and the type of the action performed when you revert changes, depend on the file status and VCS used for a particular directory.

To revert local changes, do one of the following

- In the **Local** tab of the Changes tool window, select one or more items in the relevant [changelist](#), then choose **Revert** on the context menu of the selection or click the **Revert** button  on the toolbar of the Changes tool window.
- Select the file to be reverted and choose the relevant VCS-specific revert (roll back) command on the **VCS | <VCS>** menu.

See Also

Procedures:

- [Version Control with PhpStorm](#)
- [Using Change Markers to View and Navigate Through Changes in the Editor](#)

Reference:

- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Reverting to a Previous Version

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

You can restore any previous version of a file, using the History view of a file. So doing, the current content of the file is replaced with the copy of the older content. After such operation, you have to commit the file to bring the repository up to date.

Note

The exact name of the command (revert or roll back) depends on the specific VCS.

To revert a file to its previous version

1. Select the desired file in the Project tool window, and [open its history](#).
2. In the History tab, select the desired revision.
3. On the context menu of the selected entry, choose **Get**, or click the  button on the toolbar.

See Also

Procedures:

- [Viewing Changes History for a File or Selection](#)
- [Rollback Actions with Regards to File Status](#)

Reference:

- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi> 
 - <http://youtrack.jetbrains.com/issues/WI> 
-

Shelving and Unshelving Changes

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Shelving is temporarily storing not yet committed changes in a dedicated [shelf](#).

Unshelving is returning postponed changes from a shelf to a pending changelist.

With PhpStorm, you can shelve and unshelve both separate files and entire changelists or shelves.

Unshelved changes can be [filtered out](#) from the view or [removed](#) from the shelf. Once shelved, a change can be applied as many times as you need by [unshelving](#) and subsequently [restoring](#) it on the shelf.

Note

In the [Git integration](#), in addition to shelving and unshelving "[stashing](#)" and "[unstashing](#)" are supported respectively. These features have much in common, the only difference is in the way patches are generated and applied.

- Patches with **stashed** changes are generated by Git itself. To apply them later, you do not need PhpStorm.
- Patches with **shelved** changes are generated by PhpStorm. Normally, they are also applied through the IDE. Applying shelved changes outside PhpStorm is also possible but requires additional steps.

In this section:

- [Shelving Changes](#)
- [Unshelving Changes](#)
- [Restoring Unshelved Changes](#)
- [Filtering Out and Removing Unshelved Changes](#)

See Also

Concepts:

- [Shelved Changes](#)

Reference:

- [Version Control Reference](#)
- [Shelve Changes Dialog](#)
- [Unshelve Changes Dialog](#)
- [Changes Tool Window](#)
- [Stash Dialog](#)
- [Unstash Changes Dialog](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi> 
 - <http://youtrack.jetbrains.com/issues/WI> 
-

Shelving Changes

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Shelving is temporarily storing not yet committed changes in a dedicated [shelf](#). This may be helpful, for example, when you need to switch to another urgent task, but you are short of time to bring your current code to a certain condition.

With PhpStorm, you can shelve both entire changelists or separate files.

Once shelved, a change can be applied as many times as you need by [unshelving](#) and subsequently [restoring](#) it on the shelf.

When using [Git](#) or [Mercurial](#) integration, it may be useful to have PhpStorm always shelve the base revisions of files that are under Git or Mercurial version control.

By default, PhpStorm always "remembers" the last [commit hash](#). However, this information is not sufficient if the history has been changed since the last commit as a result of running the **Rebase** operation. In this case, having a copy of the base revision may help.

- [Shelving changes](#)
- [Shelving base revision automatically](#)

To shelve changes

1. In the Local tab of the [Changes](#) tool window, select the files or changelist to put to a [shelf](#).
2. On the main **Version Control** menu or on the context menu of the selection, choose **Shelve changes**.
3. In the [Shelve Changes](#) dialog box, review the list of changed files and make sure that the files to be put to a shelf are checked out.

Tip

You can specify the desired changelist immediately in the dialog box: from the **Changelist** combo box, select the desired changelist.

4. In the **Comment** text box, type the comment to identify the shelf where the changes will be stored.
5. Click **Shelve changes**.

To have base revision shelved automatically

1. [Open the project settings](#) and click **Version Control**.
2. On the [Version Control](#) page, select the **Store on shelf base revision texts of files under DVCS** check box.

See Also

Concepts:

- [Shelved Changes](#)

Procedures:

- [Unshelving Changes](#)
- [Stashing and Unstashing Changes](#)

Reference:

- [Version Control Reference](#)
- [Shelve Changes Dialog](#)
- [Changes Tool Window](#)
- [Stash Dialog](#)
- [Unstash Changes Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

```
ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js'; var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s); })(0);
```



PhpStorm 3.0.0 Web Help

Unshelving Changes

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Unshelving is moving postponed changes from a shelf to a pending changelist.

With PhpStorm, you can unshelve both entire shelves and separate files.

Unshelved changes can be [filtered out](#) from the view or [removed](#) from the shelf. Once shelved, a change can be applied as many times as you need by [unshelving](#) and subsequently [restoring](#) it on the shelf.

To unshelve changes

1. In the **Shelf** tab of the [Changes](#) tool window, select the files or a whole shelf to unshelve.
2. On the context menu of the selection, choose **Unshelve Changes**.



3. In the [Unshelve Changes to Changelist](#) dialog box box that opens, specify the changelist you want to add the unshelved changes to. Select one of the existing changelists, or create a new one

4. Click **OK**.
5. If the conflicts occur between the patched version and the current version, resolve them as described in the section [Resolving Conflicts](#):



[Click thumbnail to view larger image.](#)

6. Select the changes in both versions to be merged to the resulting file and click **Apply**.

See Also

Concepts:

- [Shelved Changes](#)

Procedures:

- [Shelving Changes](#)
- [Stashing and Unstashing Changes](#)

Reference:

- [Version Control Reference](#)
- [Unshelve Changes Dialog](#)
- [Changes Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

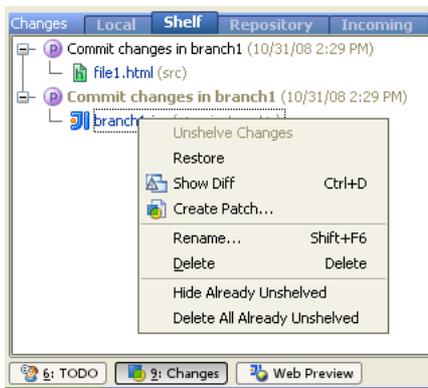
Restoring Unshelved Changes

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

With PhpStorm, you can apply changes as many times as you need by restoring them to the shelf upon unshelving. You can restore both separate files and entire shelves.

To restore unshelved changes

1. In the **Shelf** tab of the [Changes](#) tool window, select the desired files or shelf to restore.
2. On the context menu of the selection, choose **Restore**.



See Also

Concepts:

- [Shelved Changes](#)

Reference:

- [Version Control Reference](#)
- [Changes Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Filtering Out and Removing Unshelved Changes

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Most shelved changes are applied only once. You can make your list of available shelved changes easier to use by removing extraneous unshelved changes or at least filtering them out from the view

To filter out unshelved changes

1. In the Shelf tab of the [Changes](#) tool window, select the desired shelf.
2. On the context menu of the selection, choose **Hide Already Unshelved**.

To remove unshelved changes

1. In the Shelf tab of the [Changes](#) tool window, select the desired shelf.
2. On the context menu of the selection, choose **Delete Already Unshelved**.

See Also

Concepts:

- [Shelved Changes](#)

Reference:

- [Version Control Reference](#)
- [Changes Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Updating Local Information

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

The Update command enables you to synchronize the contents of your local files with the repository. You can invoke this command on:

- [Single or multiple files and directories](#).
- [Entire project](#).

Depending on the updating options, the update procedure may take place silently. If all files are up-to-date, you are notified about that. Otherwise, the Update Info tab opens in the Version Control tool window. You can [group the update information](#) as required.

To update files and folders

1. Select one or more files and folders to be updated in any navigation view (for example, Project tool window or Commander).
2. On the main Version Control menu, or on the context menu of the selection, choose **<VCS> | Update**.
3. In the Update dialog specify the update options, which are different for the supported version control systems.
4. Click **OK**.

To update a project

1. On the main menu, choose **VCS | Update project** (or use the **Ctrl+T** keyboard shortcut).
2. In the Update dialog, click the tab for your version control system (CVS, Perforce or Subversion).
3. Specify the update options, which are different for the supported version control systems.
4. Click **OK**.

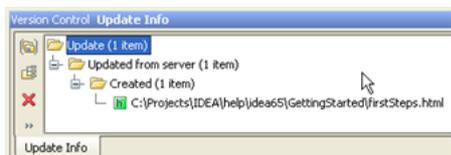
To group update information by package or changelist

- Use **Group By Package** and **Group By Changelist** buttons on the toolbar of the Update Info tab.

Note

Note the difference in the appearance of the tab:

- Grouped by packages



- Grouped by changelists



See Also

Concepts:

- [Version Control with PhpStorm](#)

Reference:

- [Version Control Reference](#)
- [Update Info Tab](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Using Patches

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

This section describes how to:

- [Create patches](#).
- [Apply patches](#).

See Also

Concepts:

- [Patches](#)

Reference:

- [Version Control Reference](#)
- [Changes Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

```
ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js'; var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s); }());
```



PhpStorm 3.0.0 Web Help

Applying Patches

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Postponed changes, stored in a patch file, can be applied to the target file or directory later. If source code has been edited after creating the patch, conflicts can arise. PhpStorm suggests a handy way to resolve such conflicts and merge patch with the results of editing.

When a patch is opened, PhpStorm detects files with the same names as the names of the modified files. For each detected file, PhpStorm compares the paths to it relative to the base directory, with the path from the patch, and chooses the closest match. If no matching path is found, the file is considered located in the project base directory and highlighted red.

You can apply changes to the files stored in the other locations than those specified in the patch, by mapping an arbitrary directory as the base one, or stripping off the leading directories.

To apply a patch

1. On the main menu, choose **VCS | Apply patch**.
2. In the [Apply Patch](#) dialog box that opens, specify the fully qualified name of the patch file. Type the name manually in the **Patch file name** text box or click the **Browse** button  and locate the desired patch file using [Select Patch File](#) dialog box.
3. Configure the patch presentation layout. To have changes shown in a flat view, press the **Group by Directory** toolbar button . Release the button to have changes shown in a directory tree view.
4. To have a change applied, select the check box next to it.
5. To have a change applied to a modified file that has been moved to another location, specify the new file location.
 - To map another base directory, select the desired file, directory, or a group files/directories and click the **Map base directory** toolbar button . In the [Select Path](#) dialog box that opens choose the directory relative to which file names will be interpreted.
 - To remove leading directories from the path, click the **Strip Directory** toolbar button  as many times as many leading directories you need to strip. The number of removed slashes is indicated in square brackets.
 - To revert the last `strip directory` action, click the **Restore Directory** toolbar button . Click the button as many times as many previously stripped leading directories you need to restore.
 - To revert all the `strip directory` actions in the selection, click the **Reset Directories** toolbar button .
 - To have all the leading directories stripped and have the changes applied to the file with the specified name in the base directory, click the **Remove Directories** toolbar button .
6. To view the differences and possible conflicts between your local working copy, the repository version, and the patch in the [Differences Viewer](#), select the desired change and click the **Show Differences** button .
7. To [resolve conflicts](#) between the patched and the current versions, if any, in both versions select the changes to be merged to the resulting file, and then click **Apply**.

See Also

Concepts:

- [Patches](#)
- [Changelist](#)

Procedures:

- [Create Patch Dialog](#)
- [Resolving Conflicts](#)

Reference:

- [Apply Patch Dialog](#)
- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
 - <http://youtrack.jetbrains.net/issues/WI> 
-

Creating Patches

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm suggests two ways of creating patches:

- On the base of the revisions, either local or committed to the repository, or
- On the base of revisions stored in the local history.

To create a patch file

1. In the [Local tab](#) or [Repository tab](#) of the Changes tool window, select a change or changelist you want to create a [patch](#) for. Alternatively, click  on the toolbar of the [Repository](#) tab.
2. On the main [Version Control](#) menu or on the context menu of the selection, choose [Create patch](#).
3. In the [Create Patch](#) dialog box that opens, review the list of changed files, and make sure that the files to be included in the patch are selected.

Tip

You can specify the desired changelist immediately in the Create Patch dialog box: click [Changelist](#) combo box in the upper right corner, and select the desired changelist.

4. Add a commit comment. As you type, PhpStorm checks the spelling and highlights words in question.

Note

This functionality is available if the [Spelling code inspection](#) is enabled.

5. Click [Create patch](#).

Tip

You can also create patch on the base of your local history. To do that, open the local history view for the desired directory, file or code fragment, as described in the section [Using Local History](#), right-click the desired revision, and choose the [Create Patch](#) command on the context menu, or click the create patch button  on the lower toolbar.

See Also

Concepts:

- [Patches](#)
- [Changelist](#)

Procedures:

- [Create Patch Dialog](#)
- [Using Local History](#)

Reference:

- [Create Patch Dialog](#)
- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
 - <http://youtrack.jetbrains.net/issues/WI> 
-

Viewing Changes Information

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm provides various ways to keep track of the changes that you and your team mates introduce to the source code.

• Viewing local changes:

Use the editor, where you create your source files, to [view, explore and navigate](#) through your local changes within a single file. Use the Local tab of the [Changes tool window](#) to [view and work with the local changes](#) throughout the entire project.

• Viewing changes of the other team members:

Use the Committed and Incoming tabs to view the changes committed to the repository [by the other team members](#).

• Viewing history:

[Browse through the changes](#), filtering the range you are interested in.
[Explore history of a file](#) in a dedicated view.
[Show annotated view of a file](#) to drill deeper into the file history and find changes information for each line of code.

See Also

Reference:

- [Version Control Reference](#)
- [Changes Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Browsing Changes

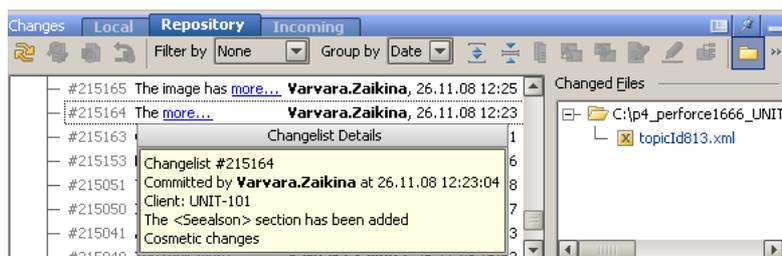
Previous | [Next](#) | [See Also](#) | [Comments](#)

Browsing changes feature helps you [see all changes](#) during the history cache period, or [confined to a certain period, user, or changelist](#).

All changes for the history cache period are displayed in the [Repository tab](#) of the Changes tool window, [Changelists](#) pane.

To browse all changes in the history cache

1. In the Changes tool window, switch to Repository tab.
2. Click refresh button  on the toolbar, or choose Refresh on the context menu.



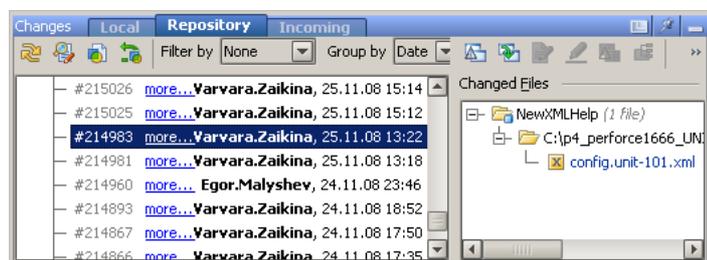
To browse changes to files or folders

1. Select the desired file or folder in the Project window.
2. On the Version Control menu or on the context menu of the selection, choose <VCS> | Browse Changes. The Change Search Criteria dialog box is opened.
3. Specify the search criteria (type the necessary filter value or press the ellipsis button to set the value for Author, Date, and Change filters) and click OK.

If you do not specify any filtering criteria, you will be prompted either to show recent changes only, or view all history of the entire project.

Note

Setting too wide filter values for a large project might require considerable time for the changes to be found.



See Also

Procedures:

- [Creating Patches](#)
- [Handling Differences](#)
- [Viewing Changes History for a File or Selection](#)

Reference:

- [Version Control Reference](#)
- [Changes Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Using Change Markers to View and Navigate Through Changes in the Editor

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

As you modify a file that is under version control, all changes are highlighted in the editor with the `change markers` that appear in the left gutter next to the modified lines, and show changes introduced since the most recent synchronization with the repository. On committing a file to the repository, the change markers disappear.

The changes you introduce to the text are color-coded:

-  line added.
-  line changed.
-  line deleted.

A special toolbar is associated with each change marker, enabling you to perform the following operations:

- [Navigate between changes](#) using the  and  buttons.
- [Compare the current version with the last committed one](#) using the  button.
- [Revert changes](#) using the  button.
- Put the previous version of the changed lines to the clipboard using the copy button .

To show the change marker toolbar

1. Point to the desired change marker. The mouse pointer changes its shape: 
2. Click the change marker. The toolbar and the previous contents of the line are displayed:

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Reverting Local Changes](#)
- [Navigating to Next/Previous Change](#)
- [Comparing File Versions](#)

Reference:

- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Viewing Details of Changes

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts:  

Besides [navigating](#) through your local changes within a file in the editor, you can view and examine these changes compared to the base revision of the file.

This functionality is provided through a dedicated `Change Details` pane in the [Local](#) tab of the [Changes](#) tool window.



The pane consists of two areas:

- Area (1) shows the affected code as it was in the base revision.
- Area (2) shows the affected code as it is after the change is introduced.

The pane can be [split horizontally or vertically](#).

In each area, PhpStorm numbers both changes themselves (4) and the lines (3) affected by them. The changes introduced to the code are color-coded:

- line changed (5).
- line added (6).
- line deleted (7).

A special toolbar (8) enables you to perform the following operations:

- [Navigate between changes](#) using the Next Change and Previous Change buttons.
- [Expand or narrow the context of a change](#) using the More/Less Lines button .
- [Configure the appearance](#) of the pane using the Settings button .

In this section:

- [Opening the dedicated Changes Details pane](#)
- [Examining details of changes in a file](#)
- [Configuring the appearance of the pane](#)
- [Closing the Change Details pane](#)

To open the dedicated change details pane

1. Open the [Changes](#) tool window (**Alt+9Alt 9**), and then switch to the [Local](#) tab.
2. Click the Change Details toggle button on the toolbar. The Change Details pane opens.

Note

The Change Details toggle button remains pressed until the pane is closed.

To examine details of changes in a file

1. Open the [Changes](#) tool window (**Alt+9Alt 9**), and then switch to the [Local](#) tab.
2. Select the file in question, and then open the [Change Details](#) pane, if it is not opened. The base revision and the local version of the selected file are opened with the differences numbered and highlighted.
3. Examine the details of each change:
 - To move to the next updated piece of code, click the [Next Change](#) button .
 - To return to the previous updated code fragment, click the [Previous Change](#) button .
 - To expand or narrow the context of an updated code fragment, position the cursor at the change in question, click the [More/Less Lines](#) button , and then specify the number of lines to be shown above and below the current code fragment.

To configure the appearance of the change details pane

1. Click the [Settings](#) button on the toolbar of the pane.
2. On the context menu, that opens, specify how you want the pane split:
 - To have the pane split horizontally, so the base revision is shown in the upper part and the locally updated version is shown in the bottom part of the pane, select the [Top/Bottom](#) option.
 - To have the pane split vertically, so so the base revision is shown in the left-hand part and the locally updated version is shown in the right-hand part of the pane, select the [Left/Right](#) option.

3. To have the `soft wraps` (or `word wraps`) used, select the **Use soft wraps** option.

To close the change details pane

- Click the pressed **Change Details** toggle button  on the toolbar.

See Also

Reference:

- [Local Tab](#)
- [Changes Tool Window](#)

Version Control:

- [Viewing Changes Information](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wiki/>
- <http://youtrack.jetbrains.com/issues/WI>

Working with Annotations

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Annotation is a form of a file presentation that shows detailed information for each line of code. In particular, for each line you can see the version from which this line originated, user ID of the person who has committed that line, and the commit date. Shortly, annotated view of a file helps you find out who did what, and, moreover, trace back the changes.

Annotating lines of code is available for ClearCase, TFS, Mercurial, git, CVS, Perforce and Subversion.

Note

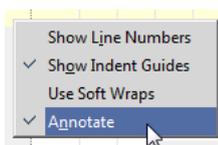
The **Annotate** command appears in the VCS-specific nodes of the Version Control menu, file context menus, and the File History view. This command toggles between plain and annotated view of a file.

This section describes how to:

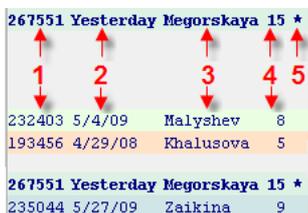
- [Show or hide annotations.](#)
- Show or hide [details of annotations.](#)
- View [revision number and comment](#) in the tooltip.
- [Highlight revisions within the specified range.](#)
- [Show differences](#) between revisions of the file (tracing back).
- [Show annotations for a specific revision of a file.](#)

To show or hide file annotations

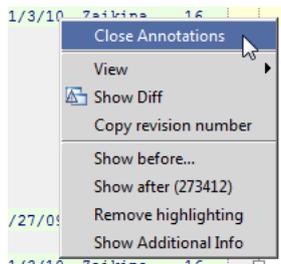
1. [Open](#) the desired file in the editor.
2. To show annotations, right-click the left gutter, and select the **Annotate** option:



When the **Annotate** option for a file is selected, the file changes its view as shown below:



1. The number of the changelist within which the annotated change has been checked in.
 2. The date when the annotated change has been checked in.
 3. The person who has checked the annotated change in.
 4. The revision number of the current file.
 5. An asterisk * indicates the changes checked in within the latest revision of the file.
3. To hide annotations, right-click the annotations gutter, and choose **Close Annotations** on the context menu:



To show or hide annotation details

1. [Show annotations](#) and switch to the **Annotations** gutter.
2. On the context menu, choose **View**, then select or clear the following options to have the corresponding information displayed or hidden:

- Revision**
The numbers of the changelists within which the annotated changes were checked in.
- Date**
Check-in date
- Author**
The names of the persons who checked in the annotated changes
- Commit Number**
Commit numbers of particular files
- Colors**
Background colors of annotations for each person who checked in a particular annotated change
- Names**
Choose the way of displaying the author name (first name, last name, or full name).

To view basic information in the tooltip

- Hover the mouse pointer over an annotation. The following tooltip appears:



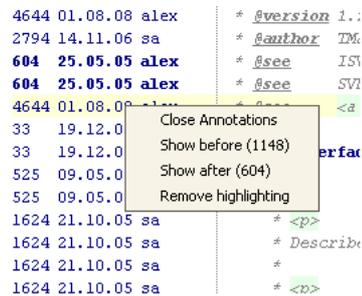
- The tooltip shows:
1. The number of the changelist within which the annotated change was checked in.
 2. The check-in message provided with the changelist.

Note

The [annotation settings](#) do not affect the range of [information displayed in the tooltip](#).

To highlight revisions within the specified range

1. [Show annotations](#) and switch to the **Annotations** gutter.
2. To specify the range, select the **Show before/Show after** in the context menu and specify the required revision numbers. The revisions with numbers within the specified range are highlighted.
3. To cancel highlighting, select **Remove highlighting** in the context menu.

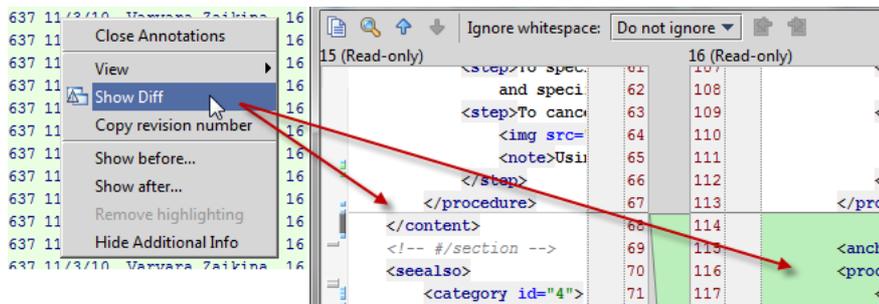


Note

Using annotations to highlight a limited set of revisions is available for Subversion integration.

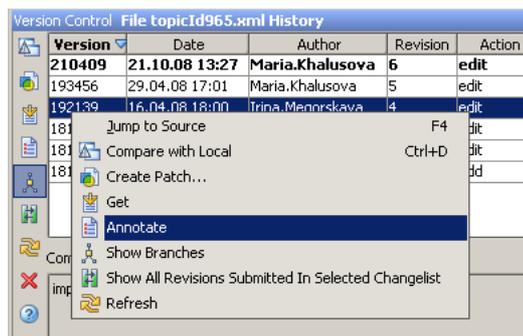
To view differences between revisions

1. [Show annotations](#) and switch to the **Annotations** gutter.
2. Position the cursor on the annotation in question and choose **Show Diff** on the context menu. PhpStorm opens the [Differences Viewer for Files](#) that shows the deviations between the annotated revision of the file and its previous revision.



To show annotation for a specific revision of a file

1. [Open the file history view.](#)
2. In the History tab, select the version you want to review and select Annotate on the context menu of the selected line.



See Also

Procedures:

- [Viewing Changes History for a File or Selection](#)

Reference:

- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

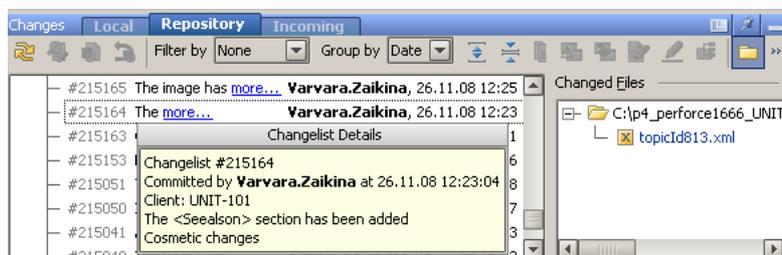
Viewing Changes Made by Other Team Members

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Use Repository and Incoming tabs of the Changes tool window to view changes introduced by the team. [Repository tab](#) shows all changes that you and your colleagues have committed to the repository; the [Incoming tab](#) shows only those committed changes, which are not yet taken to your local working copy. PhpStorm informs you about new changes in the repository by the  icon in the status bar.

To view all changes committed to the repository

1. In the Changes tool window, select Committed tab.
2. Click refresh button  on the toolbar, or choose Refresh on the context menu.



To view incoming changes

1. In the Changes tool window, select Incoming tab.
2. Click refresh button  on the toolbar, or choose Refresh on the context menu. Alternatively, click the incoming changelists icon in the status bar:



See Also

Reference:

- [Version Control Reference](#)
- [Changes Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Viewing File Status

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm provides facilities for monitoring the status of project files relative to the repository. The status of a file reflects which operations have been performed on the file locally since the last synchronization with the repository. Using this functionality requires the following prerequisites:

- [Integration](#) with a version control system is enabled.
- The entire project or its specific directories are associated with this version control system.

You can monitor the status of a project in the following ways:

- In any interface component by the [color](#) used to display the name of a file.
- In the [Changes](#) tool window, [Local](#) tab, which by default shows files grouped according to their status. The name of nodes depend on the VCS you are using.

Tip

To have files grouped by directories, click the  icon.

- For Git and CVS integrations, in the [Version Control](#) tool window, [Check Project Status Info](#) tab, via the [Check Project Status](#) command.

See Also

Procedures:

- [Version Control with PhpStorm](#)
- [VCS-Specific Procedures](#)
- [Checking Git Project Status](#)
- [Checking SVN Project Status](#)
- [Checking Perforce Project Status](#)

Reference:

- [Version Control Reference](#)
- [Local Tab](#)
- [Version Control Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Accessing the Authentication to Server Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

For a number of reasons of various nature you may be not authenticated to the remote server while the authentication dialog box does not appear, as you might expect. Instead, PhpStorm displays the corresponding message in a pop-up window in the bottom-left corner of the editor.

Tip

The authentication dialog appears immediately only when you attempt to access a URL address outside your working copies. This feature applies to Perforce, Subversion, and CVS.

To access the authentication dialog box

1. Click the **Notifications Pending** button  on the Status bar.
2. In the **Notifications** pop-up window that opens, click the link **Click to fix**.



3. In the authentication dialog box that opens, specify your credentials.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Checking in Files](#)

Reference:

- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

VCS-Specific Procedures

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

In most cases, PhpStorm provides a unified approach to version control operations, as described in the previous sections. Nevertheless, there are certain VCS-specific features and peculiarities that the user should be aware of. Find helpful tips and notes in the following sections:

- [Using CVS Integration](#)
- [Using Git Integration](#)
- [Using Perforce Integration](#)
- [Using Mercurial Integration](#)
- [Using Subversion Integration](#)
- [Using TFS Integration](#)

See Also

Concepts:

- [Version Control with PhpStorm](#) 

Procedures:

- [Configuring General VCS Settings](#)

Reference:

- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Using CVS Integration

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm integrates with the Concurrent Version System (CVS) plugin that helps you managing revisions of your source files. The plugin is bundled and turned on by default.

PhpStorm's CVS integration does not require a standalone CVS client. All you need is an account in your CVS repository.

When CVS is selected as a version control system for a module, the **CVS** menu item appears on the main VCS menu, and the CVS submenu appears on the context menus of the Editor and Project views.

See Also

Concepts:

- [Supported Version Control Systems](#)

Reference:

- [Version Control Reference](#)
- [File Status Highlights](#)
- [CVS Roots Dialog](#)
- [CVS Global Settings Dialog](#)

External Links:

- <http://ximbiot.com/cvs/cvshome/> 

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Browsing CVS Repository

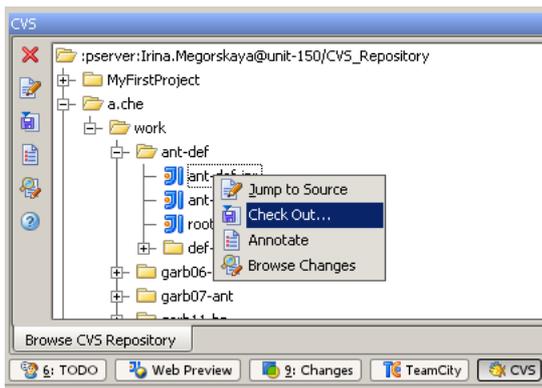
[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

You can browse any CVS repository and modify the structure of the currently open project, or a different one. Browsing contents of a repository is always available, even when CVS is not enabled in project. All you need is a valid user account.

To browse the CVS repository and modify its structure

1. Open a project. Then, choose **VCS | Browse CVS Repository** on the main menu. The Select CVS Root Configuration dialog is opened.

2. Select a root from the list of configured CVS roots, or click [Configure to specify a new one](#), and then click OK. The Browse CVS Repository tab opens in the CVS tool window at the bottom of the Editor.
3. Browse the desired CVS repository and perform jump to source, checkout, browse changes and annotate operations for files and folders:



Tip

Icons for the tree nodes denote their respective types. For example:

- means a CVS directory
- denotes a CVS module.

Icons appearing next to the files will denote the corresponding file types.

See Also

Reference:

- [Version Control Reference](#)
- [CVS Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Checking Out Files from CVS Repository

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Checking out action helps you obtain a writable copy of the repository, which you can edit as required. After making the necessary changes, you can publish results by committing, or checking in, to the repository. This section describes CVS-specific checkout procedure.

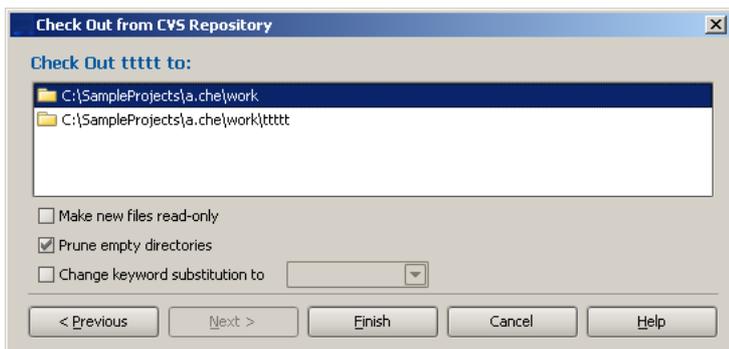
To check out files from a CVS repository

1. On the main menu, choose **VCS | Checkout from Version Control**.
2. On the submenu, choose **CVS**.
3. In the [Checkout From CVS Repository dialog](#), select the desired CVS configuration, and click **Next**. If the necessary CVS configuration is not available, click [Configure to create a new one](#).
4. Select the elements to check out, and click **Next**.
5. Specify the checkout destination directory where the local copy of the repository files will be created and click **Next**.

Note

If you are checking out sources for an existing project, the destination folder should be below a project [content root](#).

6. If you have selected an element of a CVS module to check out, the last page of the wizard suggests you to add the module name to the local path:



7. Set up CVS checkout options, or accept defaults, and click **Finish**.
8. PhpStorm suggests to create a project created based on the checked out sources. If you accept the suggestion, the [New Project from Existing Code Wizard](#) starts.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Version Control with PhpStorm](#)
- [Creating New Project from Existing Source Code](#)

Reference:

- [Version Control Reference](#)
- [Changes Tool Window](#)
- [Version Control](#)
- [CVS](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
 - <http://youtrack.jetbrains.net/issues/WI> 
-

Configuring CVS Roots

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts:  

CVS roots are global PhpStorm properties. Once configured, a CVS root is available regardless of which project is currently opened. This is helpful if you need to check out an entire project from CVS.

You can define multiple CVS root configurations for future use and edit them whenever necessary. Alternatively, you can configure CVS roots when [checking out](#) files or directories from or [importing](#) them to a CVS repository.

From this section you will learn how to:

- Configure a [new CVS root](#)
- [Modify](#) a CVS root
- Configure a CVS root [based on an existing](#) configuration
- [Remove](#) a CVS root configuration

To configure a CVS root, follow these general steps:

1. [Open the CVS Roots](#) dialog box.
2. Click the **Add** button  on the toolbar.
3. Specify the [CVS root string](#).
4. Specify the [version to work](#) with.
5. Specify [additional connection settings](#).
6. Click the **Test connection** button to check that the specified settings ensure establishing successful connection to the CVS server.
7. Click **OK** to apply the specified settings and close the dialog box.

To modify an existing CVS root configuration

1. [Open the CVS Roots](#) dialog box.
2. Select the CVS root to modify.
3. Edit the settings as while [configuring a new](#) CVS root.

Note

Changes made here apply to the current CVS root configuration only.

To configure a new CVS root based on an existing configuration

1. [Open the CVS Roots](#) dialog box.
2. Select the CVS root to be used as the basis for a new configuration.
3. Click the **Copy** button  on the toolbar or press `Ctrl+O` or `Command O`. The selected configuration is copied as a new CVS root.
4. [Edit](#) the newly created CVS root configuration, as required.

To remove a CVS root configuration

1. [Open the CVS Roots](#) dialog box.
2. Select the CVS root you want to remove and click the **Remove** button  on the toolbar.

See Also

Reference:

- [Version Control Reference](#)
- [CVS Roots Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Accessing the CVS Roots Dialog Box

Previous | [Next](#) | [See Also](#) | [Comments](#)

You can access the CVS Roots dialog box in several ways, depending on the general task you are currently performing.

To open the CVS roots dialog box, do one of the following:

- When [checking out](#) files or directories from a CVS repository: on the main menu, choose **VCS | Checkout from Version Control | CVS** and in the dialog box that opens click the **Configure** button.
- When [importing](#) files or directories to a CVS repository: on the main menu, choose **VCS | Import into CVS** and in the dialog box that opens click the **Configure** button.
- **When defining a CVS root to use in the future:** open a file which is already under CVS control and choose **VCS | CVS | Configure CVS Roots** on the main menu.

See Also

Procedures:

- [Configuring CVS Roots](#)

Reference:

- [Version Control Reference](#)
- [CVS Roots Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Assembling a CVS Root String

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

CVS root strings are specified in the **CVS Root** text box of the [CVS Roots](#) dialog box.

The CVS Root string syntax is:

```
[ :method: ] [ [user] [ :password ] @ ] hostname [ : [port] ] / path / to / repository .
```

You can obtain the valid string from your system administrator, or assemble the CVS root parameters into a correct string manually, or use the **Edit by Field** functionality, as described below.

To assemble the CVS root parameters

1. In the [CVS Roots](#) dialog box, click the **Edit by Field** button next to the **CVS Root** text box. The [Configure CVS Root Field by Field](#) dialog box opens.
2. Choose the connection method and specify the user name, port, host, and repository. See the [Configure CVS Root Field by Field](#) dialog box reference for details.
3. Click **OK**. The dialog box closes and you return to the [CVS Roots](#) dialog box. The **CVS Root** text box displays the specified parameters assembled in a valid string.

See Also

Procedures:

- [Configuring CVS Roots](#)

Reference:

- [CVS Roots Dialog](#)
- [Version Control Reference](#)
- [Configure CVS Root Field by Field Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Specifying a Version to Work with

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

By default, PhpStorm suggests you to check out the latest (HEAD) revision to work with. However, you can synchronize your local working copy with any previous revision from the repository.

Tip

Make sure that the **CVS root** field is filled in with valid data.

To specify the version to work with

1. In the **Use Version** section of the [CVS Roots](#) dialog box, select the criterion to search for the desired version. The available options are:
 - **By tag**

- **By date**

2. Click the **Browse** button  next to the selected option and do one of the following depending on the selected search criterion:

- If you have selected **By tag**, select the desired tag from the list of tags that opens. The list displays the tags obtained from the CVS server according to the specified CVS root string.
- If you have selected **By date**, select the date and time in the calendar that opens.

Note

You can also type the desired revision number, tag, or date in the text box next to the selected option.

See Also

Procedures:

- [Configuring CVS Roots](#)

Reference:

- [Version Control Reference](#)
- [CVS Roots Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Specifying Additional Connection Settings

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

You can flexibly configure connection to the CVS server using additional settings. The set of relevant options depends on the [connection method](#) specified in the CVS root string. The available additional options for each of the supported connection methods are displayed in the lower part of the [CVS Roots](#) dialog box. See the [CVS Roots](#) dialog box reference for details.

See Also

Procedures:

- [Configuring CVS Roots](#)

Reference:

- [Version Control Reference](#)
- [CVS Roots Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Configuring Global CVS Settings

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

For a module associated with CVS you can specify global CVS settings, which includes character set, location of the password file, connection timeout etc.

To configure CVS global settings for a directory associated with CVS

1. On the main menu, choose **VCS | CVS | Global Settings**.
2. In the [Global CVS Settings](#) dialog box that opens, specify the global settings and click **OK**.

Tip

Alternatively, you can define global CVS settings when [configuring a CVS root](#). To invoke the [Global CVS Settings](#) dialog box, click the **Global Settings** button in the [CVS Roots](#) dialog box.

See Also

Procedures:

- [Configuring CVS Roots](#)

Reference:

- [CVS Roots Dialog](#)
- [CVS Global Settings Dialog](#)
- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Ignoring Files

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: 

If you want your CVS integration to ignore certain unversioned files under CVS-associated directories, and skip them when performing update, import etc., add these files to the `CVS ignore list`, which is stored in the `.cvsignore` file.

The way CVS integration handles unversioned files depends on the [general settings for file creation](#). If the new files, created with PhpStorm, are not put under version control automatically, you can add them to the ignore list using CVS command. You can put the `.cvsignore` file under CVS version control, and it will be recognized by all CVS clients.

To include an unversioned file to the ignore list

1. Select an unversioned file under a CVS-associated directory (by default, such files are brown).
2. On the main **VCS** menu, or on the context menu of the selection, choose **CVS | Ignore**. If `.cvsignore` file is not under CVS version control, proceed to the next step. If `.cvsignore` already exists and is under version control, PhpStorm adds the file in question to the ignore list silently.
3. If you want to put the ignore list under version control, in the Add File `.cvsignore` to CVS dialog box, click **Add to CVS** and optionally specify the desired [keyword substitution](#).

Tip

If you want to remove a file from the ignore list, you can only do it manually. Open the `.cvsignore` file for editing by pressing **F4F4**, and remove the lines for the files that should not be ignored by CVS.

See Also

Concepts:

- [Supported Version Control Systems](#)

Reference:

- [Version Control Reference](#)
- [Version Control](#)
- [CVS](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Importing a Local Directory to CVS Repository

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

You can import an entire directory to your CVS repository, provided that you have the corresponding access rights. This action is helpful for putting a whole project under version control.

Note

Import to the repository is always available, even though CVS is not enabled in project.

To import a directory into CVS repository

1. On the main menu, choose **VCS | Import into CVS**.
2. On the first page of the [Import into CVS](#) wizard, select the target CVS root. If the desired target repository does not exist, you can create a new one. To do that, click **Configure**, and [define the desired CVS root](#). Click **Next**.
3. On the second page of the wizard, select the target directory in the repository, and click **Next**.
4. On the third page of the wizard, select the local directory that will be imported into the CVS repository. Click **Next**.
5. In the **Customize Keyword Substitution** page, specify the [keyword substitution rule](#) for the files imported into the repository, and click **Next**.
6. Specify the required [import settings](#). The fields of this page correspond to CVS command-line arguments for `import`, and additional options that define the checkout status of the imported directory.
7. Click **Finish**.

See Also

Concepts:

- [Supported Version Control Systems](#)

Procedures:

- [Adding Files to Version Control](#)
- [Configuring CVS Roots](#)

Reference:

- [Version Control Reference](#)
- [Import Into CVS](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Resolving Commit Errors

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

In the section [Checking In Files](#), you have learnt how to check in (commit) your changes to the repository. In this section you can find examples of CVS-specific error messages and suggestions on resolving conflicts.

If any error occurs when trying to commit, PhpStorm displays an error message. For example:

- If you have a modified file, which has been already changed on the server by someone else, since your last synchronization, you will get the following error:

```
Error: cvs server: Up-to-date check failed for 'source/com/...'
```

In this case you would need to [merge](#) your local copy with the current revision in the repository. When the copies are merged, and all possible conflicts are [resolved](#), so that the file is assigned the merged status, you can safely commit it to the repository.

- If you try to commit a file marked with a sticky tag, or sticky date, the CVS server will detect an attempt to *change the past*, and the error looks as follows:

```
Error: cvs server: sticky tag '1.1' for file 'source/com/impl/ManagerImpl.java&' is not a branch
```

To solve the problem, you need to update with resetting sticky data; in this case your changes will be merged with the most recent revision of the file. After resolving possible conflicts (by calling the Merge command) you will be able to commit the files.

See Also

Procedures:

- [Integrating Differences](#)
- [Resolving Conflicts](#)

Reference:

- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Updating Local Information in CVS

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

From the [Updating Local Information](#) topic, you have learnt the general procedure. CVS integration provides a special [Update File / Directory](#) dialog box with the options that map directly to the corresponding CVS command-line options of the `update` command. Refer to the [CVS documentation](#) for details.

This section will consider the CVS-specific procedure and several of the options in terms of their presentation in PhpStorm.

To update local information

1. Select one or more files and/or directories in any navigation view (for example, Project tool window or Commander).
2. On the main menu, select **VCS | Update Project** or on the context menu of the selection choose **CVS | Update File (Directory)**. The Update dialog for a project or file opens. In case of project update, select CVS tab.
3. Specify the following options:

Branch Merging

You can choose to merge your local file(s) with the counterpart(s) in one or two CVS branches. In the CVS command-line interface, this is the `-j` option.

The option *Don't Merge* is selected by default, as merging across branches is not commonly needed. If you choose one of the other options, one or both of the text fields are enabled (depending on the choice of options). Clicking the **Browse** button  next to each field opens the **Select Tag** dialog box which lists all the branch tags maintained by the repository on the CVS server. Locate the branch you want to merge with, select it in the list, and click **OK**.

Use Version

You can optionally update your local system from some different revision. You can choose a revision by its tag or by its date. The Default option synchronizes with your file's current revision, as this is the most common synchronization. The other options for Use Version are:

- **By tag (-r):** When updating a single file, you can choose the revision either by Revision or Tag. When you choose this option, the text field and corresponding ellipsis button are enabled. Enter a revision number or tag in the text field, or click the ellipsis button and select either a revision or a tag in the resulting dialog.

When updating multiple files (selected individually, or when invoking update on a directory), you can only select the revision by Tag.

- **By date (-d):** You can update from the revision of a specific date. This is possible whether you are invoking update on one file or multiple files or directories. When you choose this option, the date defaults to the current date. To specify a different date, click the ellipsis button and specify the desired date in the Choose Date dialog which appears.

Reset sticky data (-A)

If the last checkout or update of the selected file(s) was from some revision that was specified by tag or date, the tag/date information is *sticky* for the file(s). If you now want to update these sticky-tagged files from the *HEAD* revision, select this option in combination with selecting the Default option in *Use Version* so that the sticky information is removed.

Change keyword substitution to

If checked, this is converted to the `-k` CVS parameter. Refer to [CVS Options: Default keyword substitution for text files](#) for details.

Do not show this dialog in the future

Select this option if you want the update operation take place silently. For details see CVS Options.

See Also

Reference:

- [Update Directory / Update File Dialog \(CVS\)](#)
- [Version Control Reference](#)

External Links:

- <http://www.nonqu.org/cvs/> 

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Using CVS Watches

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

CVS integration of PhpStorm helps coordinate the activities of team members, who concurrently work on the same files or directories.

`CVS watches` enable you to notify the users of a CVS repository whenever a file has been opened for editing, or committed. If you watch a file or directory for changes and commits, you are added to the list of watchers.

Usage the commands related to watches depends on configuration of the `$CVSROOT/CVSROOT/notify` file.

`Edit` and `Unedit` commands change read-only status of the files or directories under CVS. If the sources were checked out with the option `-c`, you can apply `Edit` command to make them writable. In this case you are added to the list of editors. When you are done with editing, use `Unedit` command to restore read-only status. So doing, you are removed from the list of editors. If watching is configured, the watchers will receive email notification about these events.

Note

In fact, the option **Use read-only flag for not edited files** in the [CVS](#) provides the same functionality automatically. If this option is checked, `Unedit` always applies to the source files after commit.

`Edit` and `Watch` commands apply to all the files you have selected in the current view (including all the files in any selected directory), or to the current file in the editor if you invoked the command there.

This section describes how to:

- [Access Edit and Watch commands](#)
- [Change read-only status of a file or directory](#)
- [View the other persons who edit the same file or directory](#)
- [Set watch on a certain event for a file or directory and thus add yourself to the list of wanchers](#)
- [Enable or disable watching](#)
- [View the other persons who watch the same file or directory](#)

To access edit and watch commands

1. In one of the tool windows, select the desired files or directories, or open a file in the editor.
2. Do one of the following:
 - On the main menu, choose **VCS | CVS | Edit and Watch**
 - On the context menu, choose **CVS | Edit and Watch**

To get write access to a file or directory

1. [Open Edit and Watch menu.](#)
2. Choose `Edit` on the submenu. `Edit Options` dialog box is displayed.
3. If you want to gain exclusive write access, check the option **Reserved edit (-c)**. Click **OK**.

To restore read-only status of a file or directory

1. [Open Edit and Watch menu.](#)
2. Choose `Unedit` on the submenu.

To view the other persons who edit the same file or directory

1. [Open Edit and Watch menu.](#)
2. Choose **Show Editors** on the submenu. This will display the list of all users who have run the `Edit` command on the same file or directory.

To set watch on a file or directory

1. [Open Edit and Watch menu.](#)
2. Choose **Add Watch** on the submenu.
3. In the dialog box that opens, select the type of action you would like to be notified about:

- **Edit**: you will notified whenever `E`d*i*t is applied to a watched file or directory.
- **Unedit**: you will notified whenever `U`n*e*d*i*t is applied to a watched file or directory.
- **Commit**: you will notified whenever `C`o*m*m*i*t is applied to a watched file or directory.
- **All**: you will notified whenever any of the above commands is applied to a watched file or directory.

To remove watch from a file or directory

1. [Open Edit and Watch menu](#).
2. Choose **Remove Watch** on the submenu.
3. In the dialog box that opens, select the type of action for which you would like to skip notification (Edit, Unedit, Commit or All).

To suspend or resume watching

1. [Open Edit and Watch menu](#).
2. Choose **Watch Off** or **Watch On** on the submenu.

To view the list of users who are watching the same files or directories

1. [Open Edit and Watch menu](#).
2. Choose **Show Watchers** on the submenu.

See Also

Concepts:

- [Supported Version Control Systems](#)

Reference:

- [Version Control Reference](#)
- [CVS](#)

External Links:

- <http://ximbiot.com/cvs/cvshome/> 

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

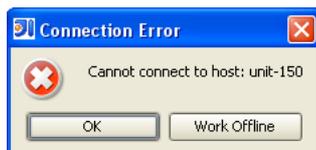
Working Offline

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Offline mode in CVS makes it possible to ignore network errors. When this mode is enabled, you will not receive any notifications about connection problems.

You can go to offline mode in two ways

- When a connection error occurs, you are presented with an option to work offline:



With the first successful transaction, offline modes automatically turns off.

- Using the CVS menu command on the main Version Control menu, or a context menu: **VCS | CVS | Work Offline**.

See Also

Concepts:

- [Supported Version Control Systems](#)

Reference:

- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Working with Tags and Branches

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

From within PhpStorm, you can create and delete CVS tags and branches. The names of the tags and branches must start with a letter, and contain only alphanumeric characters.

Branch and tag commands apply to all the files you have selected in the current view (including all the files in any selected directory), or to the current file in the editor if you invoked the command there.

This section describes how to:

- [Access the tags and branches commands.](#)
- [Create a branch](#) in the repository on the base of the current revision.
- [Tag a revision in you local working directory.](#)
- [Delete a tag.](#)

To access tags and branches commands

1. In one of the tool windows, select the desired files or directories, or open a file in the editor.
2. Do one of the following:
 - On the main menu, choose **VCS | CVS**
 - On the context menu, choose **CVS**
3. On the submenu, choose the appropriate command (**Create Branch**, **Create Tag**, or **Delete Tag**)

To create a branch

1. [Invoke](#) the **Create Branch** command.
2. In the Create Branch dialog box, specify the name of the new branch. To do that, type the name in the **Branch name** field, or click the **Browse** button  and select the desired name from the list of existing CVS tags.
3. Optionally, specify the following:
 - Select the **Override existing** check box, if you want to move an existing tag to a new branch, as defined by the option `-F` of `rtag` CVS command.
 - Select the **Switch to this branch** check box to switch your local working copy to the branch specified in the **Branch name** field.

To create a new tag

1. [Invoke](#) the **Create Tag** command.
2. In the Create Tag dialog box, specify the new tag name. To do that, type the name in the **Tag name** field, or click the **Browse** button  and select the desired name from the list of CVS tags.
3. Optionally, specify the following:
 - Select the **Override existing** check box, if you want an existing tag to point to the current revision, as defined by the option `-F` of `rtag` CVS command.
 - Select the **Switch to this tag** check box to switch your local working copy to the tag specified in the **Tag name** field.

To delete a tag

1. [Invoke](#) the **Delete Tag** command.
2. In the Delete Tag dialog box, specify the name of the tag you want to delete. To do that, type the name in the **Tag name** field, or click the **Browse** button , and select tag name from the list of the current tags in the repository.

See Also

Concepts:

- [Supported Version Control Systems](#)

Reference:

- [Version Control Reference](#)

External Links:

- <http://ximbiot.com/cvs/cvshome/> 

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Using Git Integration

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

With the [Git](#)  integration enabled, you can perform basic Git operations from inside PhpStorm.

Using the Git integration requires the following prerequisites:

- Git is installed on your computer.

Note

It is strongly recommended that you use version 1.6.0 or higher.

- The location of the Git executable file is correctly specified on the [Git](#) page of the **Settings** dialog box.
- Git [integration is enabled](#) for the current project root or directory.
- If you are going to use a remote repository, create a Git hosting account first. You can access the remote repository through the username/password and keyboard interactive authentication methods supported by the Git integration or through a pair of `ssh` keys.

Tip

1. `ssh` keys are generated outside PhpStorm. You can follow the instructions from <http://inchoo.net/tools/how-to-generate-ssh-keys-for-git-authorization/> or look for other guidelines.
2. Store the `ssh` keys in the `home directory` `\.ssh\` folder. The location of the `home directory` is defined through `environmental variables`:
 - `HOME` for `Unix-like` operating systems.
 - `userprofile` for the Microsoft Windows operating system.
3. Make sure, the keys are stored in files with correct names:
 - `id_rsa` for the private key.
 - `id_rsa.pub` for the public key.

Note

When the Git integration with PhpStorm is enabled, the Git item appears on the VCS menu.

Tip

When using the Git integration, it is helpful to open the `Version Control` tool window. Its `Console` tab displays the following data:

- All the commands generated based on the settings you specify through the PhpStorm user interface.
- Information messages concerning the results of executing generated Git commands.
- Error messages.

In this section:

- [Adding Files to a Local Git Repository](#)
- [Adding Tags](#)
- [Checking Git Project Status](#)
- [Committing Changes to a Local Git Repository](#)
- [Fetching Changes from a Remote Git Repository](#)
- [Using GitHub Integration](#)
- [Handling Passwords for Git Remote Repositories](#)
- [Managing Branches](#)
- [Setting up a Local Git Repository](#)
- [Stashing and Unstashing Changes](#)
- [Updating a Local Git Repository \(Pull\)](#)
- [Uploading a Local Git Repository \(Push\)](#)

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Version Control with PhpStorm](#)

Reference:

- [Version Control Reference](#)
- [Git Reference](#)
- [Git](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi/>
- <http://youtrack.jetbrains.net/issues/WI>

Adding Files to a Local Git Repository

Previous | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

After a Git repository for a project is [initialized](#), you need to add the project data to it.

Tip

If you have specified Git as the version control system for your project in the `Settings` dialog box, tab `Version Control`, PhpStorm suggests to put each new file under Git control during the file creation.

To have Git ignore some types of files, [configure files to ignore](#).

You can [add all unversioned](#) files to Git control or [select files to add](#).

To add all currently unversioned files to git control

1. Switch to the `Changes` tool window.
2. Navigate to the `Unversioned Files` node and choose `Add to VCS` from the context menu.

To add specific file(s) to a local git repository, do one of the following:

- Switch to the `Changes` tool window, expand the `Unversioned Files` node, and select the files to be added. From the context menu, choose `Add to VCS`.

- Switch to the [Project](#) tool window and select the files to be added. From the context menu, choose **Git | Add Snapshot** or press `Git.AddGit.Add`.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Version Control with PhpStorm](#)
- [Adding Files to Version Control](#)

Reference:

- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Adding Tags

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

The Git integration supports [tagging](#) a particular commit or object to refer to it in the future. Basically, by adding a tag you create a branch that never moves.

You can also create [annotated tags](#) and have annotations displayed to facilitate viewing and navigating through tagged commits.

To assign a tag to a commit

1. On the main menu, choose **VCS | Git | Tag Files**. The [Tag](#) dialog box opens.
2. From the **Git Root** drop-down list, select the required local repository.

Note

The **Current Branch** read-only field shows the branch you are currently working with in the selected repository. The contents of the read-only field change as you change the selection in the **Git Root** drop-down list.

3. In the **Tag Name** text box, type the name of the new tag.

Tip

If you specify a name that already exists, PhpStorm displays an error message. To override the error and re-assign the tag, select the **Force** check box.

4. In the **Commit** text box, specify the commit or object which you want to tag:
 - To tag the latest commit in the branch (HEAD), leave the text box empty.
 - To tag a particular commit, specify its commit hash or use an expression, for example, of the following structure: `<branch>~<number of commits backwards between the latest commit (HEAD) and the required commit>`.

Refer to the Git [commit naming](#) conventions for details.

5. To check that the specified commit exists and is the one you actually need, click the **Validate** button. The **Paths affected in commit** dialog box shows the files that were affected in the specified commit. View the information and click **OK**.

Note

If you specify a commit that does not exist, PhpStorm displays an error message.

To create an annotated tag

1. [Create](#) a tag.
2. In the **Message** text box, provide a description of the commit.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Committing Changes to a Local Git Repository](#)
- [Updating a Local Git Repository \(Pull\)](#)
- [Uploading a Local Git Repository \(Push\)](#)
- [Version Control with PhpStorm](#)

Reference:

- [Version Control Reference](#)
- [Tag Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

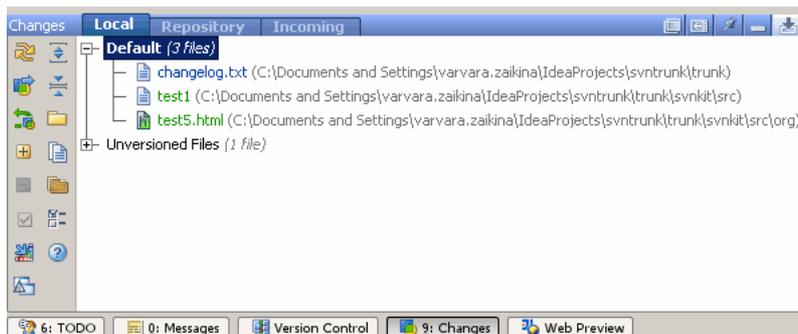
Checking Git Project Status

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Besides indicating the [current file status](#) relative to the repository, PhpStorm integration with Git provides you with the accumulated view of the project files' statuses.

To view differences between the current state of the project files and the repository

1. Open the required project.
2. On the main menu, choose **VCS | Refresh File Status**.
3. Switch to the Changes tool window, tab [Local](#).



The status of each file is indicated by the [color](#) in which the path to the file is displayed.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Refreshing Status](#)
- [Viewing File Status](#)
- [Version Control with PhpStorm](#)

Reference:

- [Version Control Reference](#)
- [File Status Highlights](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Committing Changes to a Local Git Repository

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: ⌘ Mac ⇧

This topic describes how to commit changes to a [local Git repository](#). For information on uploading changes to a remote repository, see [Uploading a Local Git Repository \(Push\)](#).

To commit changes to a local repository

1. Open the [Changes](#) tool window by pressing **Alt+Meta 9** or choosing **View | Tool Windows | Changes**.
2. Select the files or folders you want to commit.
3. Open the [Commit Changes](#) dialog box by doing one of the following:
 - On the Changes tool window toolbar, click the **Commit Changes** button .
 - On the main menu, choose **VCS | Commit Changes**.
 - On the main menu, choose **VCS | Git | Commit File**.
 - Press **Ctrl+Command K**.
4. In the **Changed Files** area, view the files changed since the last commit. Select the check boxes next to the files you want to commit now.
5. In the **Summary** area, view statistics on changes made since the last commit.
6. Add a comment and select the **Before Commit** options you want PhpStorm to perform on the files before committing them to the local repository.

Tip

To upload the committed changes to the remote repository immediately, select the **Push Changes** check box. The changes will [pushed](#) to the preconfigured origin.

7. Use the **Author** drop-down list to specify the person who created the changes to be committed.

Tip

This may be necessary when you are committing changes made by another person.

8. Click **Commit**.

Tip

To save the changes in a text file, click the **Create Patch** button.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Checking in Files](#)
- [Uploading a Local Git Repository \(Push\)](#)

Reference:

- [Version Control Reference](#)
- [Commit Changes Dialog](#)
- [Changes Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Fetching Changes from a Remote Git Repository

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

The `Fetch` operation involves downloading changes from a remote repository without applying them locally.

Tip

Basically, the `Fetch` operation is intended for downloading changes from a remote repository that is different from the [Origin](#).

To fetch changes from a remote repository

- On the main menu, choose **VCS | Git | Fetch**.
PhpStorm retrieves the changes from the repository silently.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Updating Local Information](#)
- [Version Control with PhpStorm](#)

Reference:

- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Using GitHub Integration

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm supports integration with the [GitHub](#) remote storage.

To use GitHub integration, perform these general steps

- [Enable the GitHub bundled plugin](#) to get access to GitHub integration.
- [Register your GitHub account](#) in PhpStorm.
- [Clone repositories](#) from GitHub.
- [Publish your projects](#) on GitHub.
- [Share code snippets](#) through Git gists.

See Also

Reference:

- [GitHub Integration Reference](#)
- [GitHub](#)
- [Git Reference](#)
- [Version Control Reference](#)

Version Control:

- [Using Git Integration](#)
- [Version Control with PhpStorm](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Registering GitHub Account in PhpStorm

Previous | [Next](#) | [See Also](#) | [Comments](#)

To retrieve data from GitHub repositories and share your projects in right from PhpStorm, you need to [register](#) your GitHub account credentials in the IDE. You can also [create an account](#) on the GitHub free hosting without leaving PhpStorm.

In either case, PhpStorm remembers the login and password so you do not need to specify them while [retrieving](#) or [uploading](#) data.

To register GitHub account credentials

1. [Open the IDE settings](#) and click **GitHub**.
2. On the [GitHub](#) page that opens, specify its credentials in the **Login** and **Password** text boxes and click **OK**.

To create a GitHub account from PhpStorm

1. [Open the IDE settings](#) and click **GitHub**.
2. On the [GitHub](#) page that opens, click the **Sign up** link.
3. On the **Sign up for GitHub** page that opens in the browser, choose the account plan (free or paid) and specify the requested information. When you are through with creating the account, the **GitHub Welcome Page** is displayed.

Note

Free accounts do not support `private` repositories. `Private` repositories are available only for you, for other users even `READ` access is prohibited.

4. Return to the [GitHub](#) page and specify your GitHub account credentials in the **Login** and **Password** text boxes and click **OK**.

See Also

Procedures:

- [Using GitHub Integration](#)
- [Using Git Integration](#)
- [Version Control with PhpStorm](#)

Reference:

- [GitHub Integration Reference](#)
- [GitHub](#)
- [Git Reference](#)
- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Cloning a Repository from GitHub

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Besides the standard procedure for cloning a remote repository, PhpStorm provides facilities to clone a repository located on the [GitHub](#)  remote storage.

Tip

Make sure, you have [registered your GitHub account credentials](#) in PhpStorm.

To clone a repository from the GitHub storage

1. Choose **VCS | Checkout from Version Control | GitHub** on the main menu. PhpStorm establishes connection with GitHub using the [login and password you registered](#). Upon establishing connection, the [Select Git Hub Repository to Clone](#) dialog box opens.

2. From the **Repository** drop-down list, select the source repository to clone the data from.
3. To view the details and description of the selected repository in the browser, click the **Details on <repository name> here** link.
4. In the **Folder** text box, specify the directory where the local repository for cloned sources will be set up. Type the path to the directory manually or click the **Browse** button  and choose the desired directory in the **Select project destination folder** dialog box that opens.
5. In the **Project name** text box, specify the name of the project to be created based on the cloned sources.
6. Click the **Clone** button to start cloning the sources from the specified remote repository.

See Also

Procedures:

- [Publishing a Project on GitHub](#)
- [Using GitHub Integration](#)
- [Using Git Integration](#)
- [Version Control with PhpStorm](#)

Reference:

- [Select Repository to Clone Dialog](#)
- [GitHub](#)
- [Share Project on GitHub Dialog](#)
- [GitHub Integration Reference](#)
- [Git Reference](#)
- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

2.1+

Viewing the GitHub Version of a File

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

You can view the GitHub copy of a file right from the IDE. PhpStorm detects which branch is currently active and opens the GitHub copy of the file in the corresponding branch.

To view the remote copy of a file

1. Open the file in the editor or select it in the [Project](#) tool window.
2. On the context menu, choose **Open in Browser**. The GitHub copy of the file in the remote version for the current branch opens.

Note

If the command is invoked on a folder, the page with the information on the corresponding remote folder is displayed.

See Also

Procedures:

- [Using GitHub Integration](#)
- [Using Git Integration](#)
- [Version Control with PhpStorm](#)

Reference:

- [GitHub](#)
- [GitHub Integration Reference](#)
- [Git Reference](#)
- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Publishing a Project on GitHub

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

You can [publish your project sources](#) in a GitHub Repository right from PhpStorm.

Tip

Make sure that you have [registered your GitHub account credentials](#) in PhpStorm.

To publish your project sources on GitHub

1. On the main menu, choose **VCS | Import into Version Control | Share Project on GitHub**.

- If you [have registered login and password](#), PhpStorm establishes connection with GitHub using these credentials.
 - If you have not registered your GitHub credentials in PhpStorm, the [Login to GitHub](#) dialog box. Specify your GitHub login and password or create an account there.
2. Upon establishing connection, the [Share Project on GitHub](#) dialog box opens. Specify the name of the repository to store your project sources in. By default, PhpStorm suggests the name of the current project. Provide a brief description of the project functionality.

Click the Share button. PhpStorm initiates creation of the new repository on the GitHub and [uploads the project sources](#) to it.

See Also

Procedures:

- [Using GitHub Integration](#)
- [Using Git Integration](#)
- [Version Control with PhpStorm](#)

Reference:

- [GitHub](#)
- [Share Project on GitHub Dialog](#)
- [GitHub Integration Reference](#)
- [Git Reference](#)
- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Creating Git Gists

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

To share your code snippets, you can create [gists](#)  right from PhpStorm.

Note

Creation of gists in PhpStorm requires that the following preconditions are fulfilled:

1. The [GitHub bundled plugin](#) is enabled.
2. You have an account on GitHub remote storage, and this [GitHub account is registered](#) in PhpStorm.

To create a git gist

1. In the editor, open the file that contains the snippet to be shared.
2. Select the code snippet to be shared. If no selection is made, the gist will contain the entire file contents.
3. Click the right mouse button anywhere in the editor and choose **Create gist** on the context menu.
4. In the [Create Gist](#) dialog box, that opens, provide a brief description of the gist to be created and configure the gist's visibility.
 - By default, the new gist will be **public**, that is, visible for all registered users with your login displayed as owner. To accept this default behaviour, just click **OK**.
 - To create a **private** gist that will be available for you only, select the **Private** check box.
 - To make the gist available for all registered users without displaying your login, select the **Anonymous** check box.
5.
 - To have the gist opened in the [default PhpStorm browser](#) so you can edit it right now, select the **Open in browser** check box. When you click **OK**, the <https://gist.github.com/>  page that opens, shows the new gist with the selection or the entire contents of the current file. By default, the gist is named after the file the gist originates from. To update the code snippet, if necessary, click **Edit**.
 - If you do not want to open the gist right now, clear the **Open in browser** check box. PhpStorm displays the URL address of the gist so you can access it later.

See Also

Reference:

- [Create Gist Dialog](#)
- [GitHub Integration Reference](#)
- [GitHub](#)
- [Git Reference](#)
- [Version Control Reference](#)

Version Control:

- [Using GitHub Integration](#)
- [Using Git Integration](#)
- [Version Control with PhpStorm](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Handling Passwords for Git Remote Repositories

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Every time you interact with a remote Git repository, for example, during **Pull**, **Update**, or **Push** operations, you are requested to specify the private `ssh` key or passphrase to identify yourself. You may happen to use a number of remote repositories and accordingly need to remember a number of passwords.

PhpStorm provides the possibility to [configure a password policy](#) according to which passwords are either never saved, or saved during one session and cleared upon the session end, or stored in a special `passwords` database.

The `passwords` database is under protection of a `master password`. Once [specified](#), the `master password` can be [changed](#) or [reset](#) at any time. PhpStorm stores the history of all updates to the `master password` until it is reset.

Warning

When you reset the `master password`, the history of all the previously used `master passwords` is removed.

To configure the password policy

1. [Open the IDE settings](#), then click **Passwords**.
2. On the [Passwords](#) page that opens, specify how you want PhpStorm to process passwords for Git remote repositories.

Do one fo the following:

- If you do not need PhpStorm to save passwords at all, select the **Do not remember passwords** option.
- To have passwords stored in the memory during a session, select the **Remember passwords until closing of the application** option.
- To have passwords stored in a `passwords` database, select the **Remember on disk (protected with master password)** option and [specify the master password](#) for the storage.

To set the master password

1. On the [Passwords](#) page of the **Settings** dialog box, select the **Remember on disk (protected with master password)** option.
2. Click the **Reset** button.
3. In the [Master Password](#) dialog box that opens, specify the password to use. Type the desired password once again to confirm your setting.

To change the master password

1. On the [Passwords](#) page of the **Settings** dialog box, click the **Change Password** button.
2. In the [Change Master Password](#) dialog box that opens, type the currently used `master password` and the password to change it to. Confirm the new password by typing it once again.

To reset the master password

1. Open the [Reset Master Password](#) dialog box. Do one of the following:
 - On the [Passwords](#) page of the **Settings** dialog box, click the **Reset Master Password** button.
 - In the [Change Master Password](#) dialog box, click the **Reset Password** button.
2. In the **New Password** text box, type the `master password` to replace the current one with.
3. In the **Confirm Password** text box, type the new `master password` once again to confirm your setting.

See Also

Reference:

- [Passwords](#)
- [Git Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Managing Branches

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm supports processing of multiple Git branches using sophisticated checkout and merge strategies and provides interface for configuring operations on branches.

In this part:

- [Creating a New Branch](#)
- [Managing Branch Tracking](#)
- [Merging Branches](#)
- [Rebasing Branches](#)
- [Switching Between Branches \(Checkout\)](#)
- [Resetting Head Commit](#)
- [Applying Changes from a Specific Commit to Other Branches \(Cherry Picking\)](#)

See Also

Reference:

- [Version Control Reference](#)
- [Checkout Dialog](#)
- [Merge Branches Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Creating a New Branch

Previous | [Next](#) | [See Also](#) | [Comments](#)

With the Git integration, you can create branches according to the following scenarios:

- Check out a branch into a [new branch](#).
- [Overwrite an existing branch](#) by checking out another branch into it.

To check out a branch into a new branch

1. On the main menu, choose **VCS | Git | check out Branch**. The [Checkout](#) dialog box opens.
2. From the **Git Root** drop-down list, select the required local repository.

Note

The **Current Branch** read-only field shows the branch with which you are currently working in the selected repository. The contents of the read-only field change as you change the selector in the **Git Root** drop-down list.

3. In the **Checkout** drop-down list, select the branch which you want to check out into a new branch.

Note

If you do not specify any particular commit the HEAD of the selected branch will be checked out.

4. Specify which commit of the selected branch you need:
 - To check out a tagged commit, select the **Include Tags** check box - tags will be also available in the **Checkout** list. Choose the required tag.
 - To check out a commit for which no tag is assigned, type its commit hash or use an expression, for example, of the following structure: `<branch>~<number of commits backwards between the latest commit (HEAD) and the required commit>`. Refer to the Git [commit naming](#) conventions for details.
5. To check that the specified commit exists and is the one you actually need, click the **Validate** button. The **Paths affected in commit** dialog box shows the files that were affected in the specified commit. View the information and click **OK**.

Note

If you specify a commit that does not exist, PhpStorm displays an error message.

6. In the **As New Branch** text box, type the name of the new branch.
7. Specify additional parameters of the new branch:
 - To have a reference log for the new branch created where all changes made to the branch references will be recorded, select the **Create Ref Log** check box.
 - To have the parent of the new branch tracked, select the **Track Branch** check box.

To overwrite an existing branch

1. [Open the Checkout](#) dialog box, select the required [local repository](#), select the [source branch](#), and specify the [required commit](#).
2. In the **As New Branch** text box, type the name of the existing branch which you want to overwrite. PhpStorm displays a message which warns you that the branch with the specified name already exists.
3. Select the **Override** check box.
4. Specify the additional parameters of the branch using the **Create Ref Log** and **Track Branch** check boxes.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Switching Between Branches \(Checkout\)](#)

Reference:

- [Version Control Reference](#)
- [Checkout Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Managing Branch Tracking

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

With the Git integration, you can find out which [branch is currently being tracked](#) in any of your repositories. You can also [switch tracking](#) to another branch in any repository.

To find out which branch is currently being tracked

1. On the main menu, choose **VCS | Git | Current Branch**. The [Current Branch](#) dialog box opens.
2. In the **Tracked Branch** area, select the desired repository from the **Repository** drop-down list. The **Branch** field shows the branch which is currently tracked in the selected repository.

To have another branch tracked

1. In the [Current Branch](#) dialog box, select the desired repository from the **Repository** drop-down list.
2. From the **Branch** drop-down list select the desired branch and click the **Change Tracked Branch** button.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Switching Between Branches \(Checkout\)](#)
- [Creating a New Branch](#)

Reference:

- [Version Control Reference](#)
- [Current Branch Dialog](#)
- [Checkout Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
 - <http://youtrack.jetbrains.net/issues/WI> 
-

Merging Branches

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

The Git integration provides interface for selecting branches to merge and specifying the merge strategy.

The target branch is the [currently checked out](#) branch, the changes are merged into it.

The number of source files from which commits are merged into the current branch is unlimited. The source branches remain unchanged and all the changes are applied to the target branch only.

The merge affects both the set of files in the target branch and the contents of the files. After merge some new files may appear in the target branch while some existing files may be removed.

If any conflicts arise, PhpStorm attempts to resolve them only if you are merging two branches, that is you have only one source branch. When two or more source branches are involved, any conflict results in a merge failure.

To merge branches

1. On the main menu, choose **VCS | Git | Merge Changes**. The [Merge Branches](#) dialog box opens.
2. From the **Git Root** drop-down list, select the required local repository.
3. Make sure the **Current Branch** read-only field shows the branch to which you actually want to apply the merge results. If necessary, [switch](#) to the correct branch.
4. In the **Branches to Merge** list, select the branches that you want to involve in the merge.

Tip

The list shows only branches that contain applicable commits. A commit is applicable if it was made after the branch was separated from the target branch.

5. Specify the [merge strategy](#)  and additional settings using the dialog box controls described in the [Merge Branches](#) dialog reference.

Tip

When you have selected two or more source branches in the **Branches to Merge** list box:

- Only the **Octopus** and **Ours** merge strategies are available. This prevents potentially harmful attempts to resolve conflicts.
- The merge results will be automatically committed if no conflicts arise. The **No Commit** check box is disabled

6. If you want to commit the merge result to the local repository, provide a commit description in the **Commit Message** text box.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Version Control with PhpStorm](#)

Reference:

- [Version Control Reference](#)
- [Merge Branches Dialog](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Rebasing Branches

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

The Git integration with PhpStorm supports the `Rebase` operation and provides interface that ensures high flexibility in setting rebase arguments.

The following functionality is supported:

- The [basic use case](#), which involves [applying a branch on top](#) of the current HEAD of the master after synchronization with the upstream.
- Rebasing a branch entirely or partially to a [specific commit](#) in any branch or tag.
- Running rebase on several local repositories simultaneously.
- Selecting a merge strategy to apply, with the possibility to use no merging strategy at all.
- Running [rebase interactively](#) with control over preserving/squashing merges.
- [Resuming interrupted rebase](#) after merge conflicts are resolved.
- [Cancelling](#) rebase.

To initiate rebasing

1. On the main menu, choose **VCS | Git | Rebase**. The [Rebase Branches](#) dialog box opens.
2. From the **Git Root** drop-down list, select the relevant local repository.
3. From the **Branch** drop-down list, select the branch to rebase.

Tip

By default, the current branch is selected. If you specify another branch, it will be checked out.

4. Specify the new base and commits to apply.

To resume interrupted rebase

- On the main menu, choose **VCS | Git | Continue Rebasing**.

Tip

Before resuming rebase, view the log in the [Version Control](#) tool window.

Note

If rebase has been initiated and interrupted on two or more local repositories, the **Continue Rebasing** dialog box is displayed. Use the **Git Root** drop-down list, to specify the repository to resume rebase on.

To cancel rebase

- On the main menu, choose **VCS | Git | Abort Rebasing**.

Note

If rebase has been initiated on two or more local repositories, the **Abort Rebasing** dialog box is displayed. Use the **Git Root** drop-down list, to specify the repository to cancel rebase on.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Applying a Branch Entirely on Top of Master](#)
- [Rebasing a Branch to a Specific Commit](#)
- [Interactive Rebase](#)

Reference:

- [Version Control Reference](#)
- [Rebase Branches Dialog](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Applying a Branch Entirely on Top of Master

Previous | [Next](#) | [See Also](#) | [Comments](#)

Suppose you have a branch `branch1` based on `master`. While you are working in `branch1`, some updates have been committed to `master`. The diagram below illustrates rebasing `branch1` so that it applies on top of the current HEAD of `master`.



By default, the commits 1, 2, and 3 are applied one after another in the chronological order. To skip, edit, squash commits or change their order, run rebase in the [interactive mode](#).

To apply a branch entirely on top of the current head of the master

1. [Initiate the rebase](#) procedure.
2. Clear the **Preserve Merges** check box.
3. Clear the **Interactive** check box.
4. From the **Onto** drop-down list, select the `master` branch.
5. Clear the selection in the **From** drop-down list, if anything is selected.
6. From the **Merge Strategy** drop-down list, select **Default**.
7. Click the **Rebase** button. The rebase process starts. View the rebase log in the [Version Control](#) tool window, [resolve conflicts](#) that arise, and [resume](#) rebasing.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Rebasing Branches](#)
- [Rebasing a Branch to a Specific Commit](#)
- [Interactive Rebase](#)

Reference:

- [Version Control Reference](#)
- [Rebase Branches Dialog](#)
- [Rebasing Commits Dialog](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wiki/>
- <http://youtrack.jetbrains.com/issues/WI>

Interactive Rebase

Previous | [Next](#) | [See Also](#) | [Comments](#)

Rebasing branches interactively provides you with the possibility to apply commits in the necessary order, squash or edit commits before applying, and skip commits that contain extraneous changes.

Interactive mode is available for rebasing branches entirely or partially on top of the HEAD or to any specific commit.

To rebase a branch in the interactive mode

1. [Initiate the rebase](#) procedure.
2. Select the **Interactive** check box.

Tip

To have PhpStorm try to recreate merges instead of ignoring them, select the **Preserve Merges** check box.

Warning

Git does not support squashing commits when the **Preserve merges** option in enabled.

3. Specify the [new base](#), the [range of commits](#) to apply, and the [merge strategy](#).
4. Click OK. The [Rebasing Commits](#) dialog box opens displaying a list of all the commits within the specified range in the chronological order. For each commit its hash and comment are shown.

Tip

To view which files are affected in a commit, select the commit and click the **View** button.

5. Define the order of processing the commits by selecting the relevant lines and clicking the **Move Up** and **Move Down** buttons.
6. Use the **Action** drop-down list to define how each commit should be processed:
 - o To apply a commit as is, select the **Pick** option.
 - o To update a commit before applying, select the **Edit** option.
 - o To ignore a commit, select the **Skip** option.
 - o To combine a commit with the previous commit, select the **Squash** option.

Tip

After you start rebasing, you will be asked to supply additional information on the squashed commits.

Note

If the affected commits have different authors, the squashed commit will be attributed to the author of the first commit.

7. Click the **Rebase** button. The rebase process starts. View the rebase log in the [Version Control](#) tool window, [resolve conflicts](#) that arise, and [resume](#) rebasing.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Rebasing Branches](#)
- [Applying a Branch Entirely on Top of Master](#)
- [Rebasing a Branch to a Specific Commit](#)

Reference:

- [Version Control Reference](#)
- [Rebase Branches Dialog](#)
- [Rebasing Commits Dialog](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wiki/>
- <http://youtrack.jetbrains.com/issues/WI>

Rebasing a Branch to a Specific Commit

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

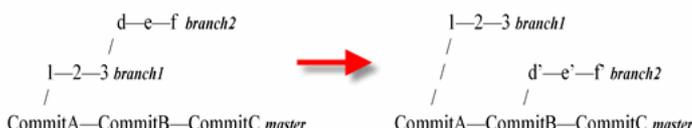
Suppose you have a branch `branch1` based on `master`. While you are working in `branch1`, some updates have been committed to `master`. For some reason, you do not want to rebase `branch1` to the current HEAD of `master` but to commit `CommitB`. Moreover, you do not want to apply commit 1 but want to start with commit 2.

The diagram below illustrates the described rebasing operation.



Transplanting a branch to a branch different from its base is also supported. Suppose your `branch2` is based on `branch1`, which, in its turn, is based on `master`.

The diagram below illustrates rebasing `branch2` to a specific commit in `master`.



To rebase a branch to a specific commit

1. [Initiate the rebase](#) procedure.
2. Configure the rebase mode:
 - o To [interactively](#) handle the list of commits to apply, select the **Interactive** check box.
 - o To have PhpStorm try to recreate merges instead of ignoring them, select the **Preserve Merges** check box.

Note

This option is available only in the interactive mode.

3. Define the contents of the **Onto** and **From** drop-down lists. By default, tags and remote branches are not available from them. To expand the lists with tags or remote branches, select the **Show Tags** and **Show Remote Branches** check boxes respectively.

4. From the **Onto** drop-down list, select the required branch and specify the commit to move the base to. Type the desired commit hash or use an expression, for example, of the following structure:

```
<branch>~<number of commits backwards between the latest commit (HEAD) and the required commit>
```

Refer to the [Git commit naming](#) conventions for details.

Tip

If you select a tag, no commit specification is required.

- Click the **Validate** button and view view which files are affected in the specified commit in the **Paths affected in commit...** dialog box that opens.
- From the **From** drop-down list, select the current branch and specify the commit, starting from which you want to apply commits on top of the new base. Type the desired commit hash or use an expression, for example, of the following structure:

```
<branch>~<number of commits backwards between the latest commit (HEAD) and the required commit>
```

Refer to the Git [commit naming](#) conventions for details.

Tip

To apply all commits, leave the field empty.

- Specify the merge procedure to be applied, when necessary:
 - To perform all merges manually, select the **Do not use merging strategies** check box.
 - To have PhpStorm apply one of the available merging methods, select the relevant option from the **Merge Strategy** drop-down list. See the [Rebase Branches](#) dialog box reference for description of available options.
- Click the **Rebase** button and proceed depending on the chosen rebase mode:
 - If you have specified the [interactive](#) mode, configure the list of commits to apply in the [Rebasing Commits](#) dialog box, that opens.
 - If you have specified the automatic mode, view the rebase log in the [Version Control](#) tool window, [resolve conflicts](#) that arise, and [resume](#) rebasing

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Rebasing Branches](#)
- [Applying a Branch Entirely on Top of Master](#)
- [Interactive Rebase](#)

Reference:

- [Version Control Reference](#)
- [Rebase Branches Dialog](#)
- [Rebasing Commits Dialog](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Switching Between Branches (Checkout)

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

With the Git integration, you can [switch](#) between existing branches within a local repository and get any tag or any commit of the required branch.

You can also find out which [branch is currently checked out](#) in any of your existing local repositories.

To find out which branch is currently checked out in a local repository

- On the main menu, choose **VCS | Git | Current Branch**. The [Current Branch](#) dialog box opens.
- In the **Git Root** drop-down list, select the local repository you need. The **Current Branch** read-only field shows the branch with which you are currently working in the selected repository.

To switch to another existing branch

- On the main menu, choose **VCS | Git | Checkout Branch**. The [Checkout](#) dialog box opens.
- From the **Git Root** drop-down list, select the local repository you need.

Note

The **Current Branch** read-only field shows the branch with which you are currently working in the selected repository. The contents of the read-only field change as you change the selection in the **Git Root** drop-down list.

- In the **Checkout** drop-down list, select the branch you want to switch to.

Note

If you do not specify any particular commit you will switch to the HEAD of the selected branch.

- Specify which commit of the selected branch you need:

- To switch to a tagged commit, select the **Include Tags** check box - tags will be also available in the **Checkout** list. Choose the required tag.
- To switch to a commit for which no tag is assigned, type its commit hash or use an expression, for example, of the following structure:

```
<branch>><number of commits backwards between the latest commit (HEAD) and the required commit>.
```

Refer to the Git [commit naming](#) [↗] conventions for details.

5. To check that the specified commit exists and is the one you actually need, click the **Validate** button. The **Paths affected in commit** dialog box shows the files that were affected in the specified commit. View the information and click **OK**.

Note

If you specify a commit that does not exist, PhpStorm displays an error message.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Reference:

- [Version Control Reference](#)
- [Current Branch Dialog](#)
- [Checkout Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> [↗]
 - <http://youtrack.jetbrains.net/issues/WI> [↗]
-

Resetting Head Commit

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Suppose, you notice a small error in a recent commit or a set of commits and want to redo that part without showing the undo in the history. In this case resetting the latest commit in the current branch (**HEAD**) to an earlier commit is helpful. The **Reset HEAD** operation sets the current **HEAD** to the specified commit and optionally resets the index and working tree to match.

To reset the current head to an earlier commit

1. On the main menu, choose **VCS | Git | Reset HEAD**. The **Reset Head** dialog box opens.
2. From the **Git Root** drop-down list, choose the required local repository.
3. Make sure the **Current Branch** read-only field shows the branch in which you actually want to reset the **HEAD**. If necessary, [switch](#) to the correct branch.
4. From the **Reset Type** drop-down list, select the desired reset strategy to apply:
 - To have the changed files preserved but not marked for commit, select the **Mixed** option. The index will be reset while the working tree will not and you will get a report of what has not been updated.

Tip

This is the default strategy.

- To have only the **HEAD** pointer moved without updating the index and the working tree, select the **Soft** option. Your current state with any changes will remain different from the commit you are switching to.
 - To have both the working directory and the index changed to the specified commit, select the **Hard** option.
5. In the **To Commit** text box, specify the commit you want to reset the current **HEAD** to. Type its commit hash or use an expression, for example, of the following structure:

```
<branch>><number of commits backwards between the latest commit (HEAD) and the required commit>.
```

Refer to the Git [commit naming](#) [↗] conventions for details.

6. To check that the specified commit exists and is the one you actually need, click the **Validate** button. The **Paths affected in commit** dialog box shows the files that were affected in the specified commit. View the information and click **OK**.

Note

If you specify a commit that does not exist, PhpStorm displays an error message.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Committing Changes to a Local Git Repository](#)

Reference:

- [Version Control Reference](#)
- [Reset Head Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Applying Changes from a Specific Commit to Other Branches (Cherry Picking)

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Suppose, you have two diverged branches, `master` and `experimental`. One day, you decide to integrate changes from one specific commit in `experimental` to `branch_2`. With PhpStorm, you can cherry pick between branches right from the [Changes](#) tool window.

Note

Unlike the native [git cherry-pick](#) action, PhpStorm does not commit the selected changes to the other branch automatically but stores them in a new change list.

To cherry pick changes between two branches

1. [Switch to the target branch](#) the changes will be integrated to.

Warning

It is required that your working tree should be clean, that is, there should be no modifications from the `HEAD` commit.

2. Open the [Changes](#) tool window and switch to the [Log](#) tab.
3. From the **Branch** drop-down list on the toolbar, select the source branch from which you want to integrate the changes.

Tip

The **Commits** area lists all the commits performed in the selected branch. To reduce the number of items in the list, specify the author of the required commit in the **User** drop-down list.

4. From the **Commits** list, select the required commit. Use the commit information below, if necessary.
5. Click the **Cherry-pick** button  on the toolbar. PhpStorm displays a dialog box that informs you that changes are going to be cherry-picked to the current branch. Click **Continue** to launch cherry picking. When cherry picking is completed, PhpStorm displays the corresponding information message.
6. Switch to the [Local](#) tab where the cherry picked changes are grouped in a new change list. The name of this change list is the commit message of the commit the changes are cherry picked from.
7. [Commit the changes](#) from the new change list.

See Also

Reference:

- [Commit Changes Dialog](#)
- [Changes Tool Window](#)
- [Log Tab](#)
- [Git Reference](#)

External Links:

- <http://www.kernel.org/pub/software/scm/git/docs/git-cherry-pick.html>

Version Control:

- [Managing Branches](#)
- [Committing Changes to a Local Git Repository](#)
- [Using Git Integration](#)
- [Checking in Files](#)
- [Version Control with PhpStorm](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Setting up a Local Git Repository

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Although Git provides high flexibility in arranging data and your work with repositories, the following scenarios are most commonly used for setting up a local Git repository:

- [Clone](#) an existing remote repository and create a new project with the downloaded data.
- [Create a local repository](#) which you can [push](#) to a remote location later, if necessary.

To clone a remote git repository

1. On the main menu, choose **VCS | Checkout from Version Control | Git**. The [Clone Repository](#) dialog box opens.
2. In the **Git Repository URL** text box, type the URL of the remote repository which you want to clone.
3. Click the **Test** button next to the **Git Repository URL** text box to check that connection to the remote repository can be established successfully.

4. In the **Parent Directory** text box, specify the directory where you want PhpStorm to create a folder for your local Git repository. Use the **Browse** button  button, if necessary.
5. In the **Directory Name** text box, specify the name of the new folder into which the repository will be cloned. Click **Clone**.
6. Create a new project based on the cloned data by accepting the corresponding suggestion displayed by PhpStorm.

To create a local git repository

1. Open the project you want to store in a repository.
2. On the main menu, choose **VCS | Import into Version Control | Create Git Repository**. The **Select Path** dialog box opens.
3. Specify the directory where you want to create a new Git repository.

Tip

By default, PhpStorm suggests the root of the current project.

Warning

Git does not support external paths. So if you choose another directory, note that it must contain the tree where the project root resides.

4. Put the required [files under Git](#) version control. The files appear in the [Changes](#) tool window under the **Default** node.

Tip

If you specify Git as the version control system for a directory in the [Version Control](#) dialog box, PhpStorm will suggest to put each new file in this directory under Git control.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Version Control with PhpStorm](#)
- [Adding Files to Version Control](#)
- [Getting Local Working Copy of the Repository](#)

Reference:

- [Version Control Reference](#)
- [Clone Repository Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Stashing and Unstashing Changes

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Sometimes it may be necessary to revert your working copy to match the HEAD commit but you do not want to lose the work you have already done. This may happen if you learn that there are upstream changes that are possibly relevant to what you are doing or if you need to make some urgent fixes.

[Stashing](#) involves recording the difference between the HEAD commit and the current state of the working directory (stash).

Note

Changes to the index can be stashed as well.

[Unstashing](#) involves applying a stored stash to a branch.

You can apply a stash to an [existing branch](#) or create a [new branch](#) on its basis.

Note

A stash can be applied as many times as you need to any branch you need, just [switch](#) to the required branch. Keep in mind that:

- Applying a stash after a series of commits results in conflicts that need resolving.
- A stash cannot be applied to a "dirty" working copy, that is a working copy to which changes have been made since the latest commit.

You can store stashes in a list as long as you need and [remove](#) them from the list when necessary.

Note

As you might have noticed, [Stashing](#) and [Unstashing](#) have much in common with [Shelving](#) and [Unshelving](#) respectively.

The only difference is in the way patches are generated and applied:

- Patches with [stashed](#) changes are generated by Git itself. To apply them later, you do not need PhpStorm.

- Patches with shelved changes are generated by PhpStorm. Normally, they are also applied through the IDE. Applying shelved changes outside PhpStorm is also possible but requires additional steps.

To save modifications to a new stash

1. On the main menu, choose **VCS | Git | Stash Changes**. The [Stash](#) dialog box opens.
2. Select the relevant Git root and make sure that the correct branch is checked out.
3. In the **Message** text box, describe the changes to be stashed.

Tip

To stash the local changes and bring the changes staged in the index to your working tree for examination and testing, select the **Keep Index** check box.

To apply a stash to the same branch

1. On the main menu, choose **VCS | Git | Unstash Changes**. The [Unstash Changes](#) dialog box opens.
2. Select the Git root where you want to apply a stash and make sure that the correct branch is checked out.
3. In the **Stashes** list, select the relevant stash.

Tip

Examine the stash descriptions and the names of the branches where specific stashes were created to find the stash you need.

4. Click the **View** button to open the **Paths affected in commit** dialog box and find out which files are affected in the selected stash.
5. Specify additional unstash options:
 - To have the selected stash removed from the list after it is applied, select the **Pop stash** check box.
 - To have the stashed index modifications applied, select the **Reinstate Index** check box.

Warning

This operation may fail if you have conflicts. This happens because conflicts are stored in the index, where you can no longer apply the changes as they were originally.

To create a new branch on the basis of a stash

1. On the main menu, choose **VCS | Git | Unstash Changes**. The [Unstash Changes](#) dialog box opens.
2. Select the Git root where you want to apply a stash.
3. In the **Stashes** list, select the relevant stash.
4. In the **As new branch** text box, type the name of the new branch to be created.

To remove one or several available stashes

1. On the main menu, choose **VCS | Git | Unstash Changes**. The [Unstash Changes](#) dialog box opens.
2. In the **Stashes** list, select the stashes to be removed and click the **Drop** button.

Tip

To remove all stashes from the list, click the **Clear** button.

See Also

Concepts:

- [Shelved Changes](#)
- [Version Control with PhpStorm](#)

Procedures:

- [Shelving and Unshelving Changes](#)
- [Version Control with PhpStorm](#)

Reference:

- [Version Control Reference](#)
- [Stash Dialog](#)
- [Unstash Changes Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Updating a Local Git Repository (Pull)

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Refreshing a local Git repository with the changes from the remote repository (**Pull**) involves retrieving changes (**Fetch**) and applying them to the local data (**Merge**). The Git integration with PhpStorm provides interface for specifying the mandatory **Pull** settings and for customizing the **Pull** procedure.

To pull changes from a remote repository

1. On the main menu, choose **VCS | Git | Pull Changes**. The [Pull Changes](#) dialog box opens.
2. Specify the required Git root and the URL address or alias of the source remote repository.
3. From the **Branches to Merge** list, select the remote branches you want to merge to the current local branch.

Tip

To use the list of branches from the remote repository, click the **Get Branches** button.

4. Specify the pull strategy and additional settings using the dialog box controls described in the [Pull Changes](#) dialog box reference.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Updating Local Information](#)
- [Fetching Changes from a Remote Git Repository](#)
- [Merging Branches](#)
- [Version Control with PhpStorm](#)

Reference:

- [Version Control Reference](#)
- [Pull Changes Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Uploading a Local Git Repository (Push)

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

The Git integration in PhpStorm supports the following modes of uploading changes to a remote repository:

- Uploading changes from [one or several branches within one local repository](#).
- Uploading changes from the [active branches within one or more local repositories](#) through one **Push** operation.
- If you want to push active branches from several local repositories and skip a commit but include some commits performed later, PhpStorm will inform you that you need to [run rebase before push](#).

Note

Before uploading (**Pushing**) changes from your local Git repository to a remote repository make sure that:

- You have a Git hosting account.
- The target remote Git repository is accessible for your account.
- You have a pair of `ssh keys` to access the remote repository.

To push changes from one local git repository

1. On the main menu, choose **VCS | Git | Push Changes**. The [Push Changes](#) dialog box opens.
2. From the **Git Root** drop-down list, select the location of the local repository you want to push.
3. From the **Push** drop-down list, select the alias of the target remote repository.

Note

If you have accepted the default alias during [pull](#), the origin alias is displayed.

Tip

To check that the selected alias corresponds to the correct URL address, expand the list - the URL addresses will be displayed explicitly.

4. Specify the push strategy, objects to push, and additional settings using the **Push** drop-down list, the **Branches** list, and the **Push Tags**, **Use Thin Pack**, and **Force Update** check boxes. Refer to the [Push Changes](#) dialog box reference for a detailed description of the controls.

To push changes from active branches within several local git repositories

1. On the main menu, choose **VCS | Git | Push Active Branches**.
2. In the **Commits** list box of the [Push Active Branches](#) dialog box, configure the set of commits to be pushed. The list shows active branches in all your local repositories (roots). All the commit

performed in the active branch of a root since the last push are shown below the relevant root node. The latest commit is shown at the top of each list.

- o To find out which files contain the changes within a commit, select the desired commit and click the **View** button. The **Paths affected in commit** dialog box shows the files with the committed changes.
 - o To have a commit pushed, select the check box next to it.
3. To synchronize with the remote repository, click the **Fetch** button. PhpStorm will download remote changes without applying them automatically. [View the differences](#), if any, and apply them as necessary.
 4. Click the **Push** button, when ready.

To run rebase before push

1. To synchronize with the remote repository, click the **Fetch** button. PhpStorm will download remote changes without applying them automatically. [View the differences](#), if any, and apply them as necessary.
2. Specify how you want the [working tree cleaned](#) before rebase. The available options are:
 - o **Using Stash** - choose this option to have a patch with [stashed changes](#) generated.
 - o **Using Shelve** - choose this option to have a patch with [shelved changes](#) generated.

Note

- o Patches with [stashed](#) changes are generated by Git itself. To apply them later, you do not need PhpStorm.
 - o Patches with [shelved](#) changes are generated by PhpStorm. Normally, they are also applied through the IDE. Applying shelved changes outside PhpStorm is also possible but requires additional steps.
3. Do one of the following:
 - o To have the **Rebase** and **Push** operations performed explicitly:
 1. To initiate rebasing, click the **Rebase** button. If no conflicts are detected, rebasing is performed silently and the commit to be skipped is moved to the top of the list. Otherwise, [resolve](#) them using the Merge tool that opens.
 2. Click the **Push** button, when ready.
 - o To have rebasing and pushing performed in the background, click the **Rebase and Push** button.

Warning

It is strongly recommended that you do not perform any editing of the affected files until the process is completed successfully.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Committing Changes to a Local Git Repository](#)
- [Version Control with PhpStorm](#)

Reference:

- [Version Control Reference](#)
- [Push Changes Dialog](#)
- [Push Active Branches](#)

External Links:

- <http://www.kernel.org/pub/software/scm/git/docs/git-push.html>
- <http://www.kernel.org/pub/software/scm/git/docs/git-rebase.html>
- <http://www.kernel.org/pub/software/scm/git/docs/git-clean.html>

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Using Perforce Integration

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

When Perforce integration is enabled, the **Perforce** menu item is added to the main **Version Control** menu, to the context menus of the editor and Project tool window, and the project artifacts are highlighted according to their status. Results of modifications show in the Changes tool window.

Note

Using Perforce integration in PhpStorm, requires the following **PREREQUISITES**:

- A Perforce client is installed on your computer.
- You have an account with the Perforce depot.

To start using Perforce integration

1. Create a client spec using your Perforce client.
2. Get source files from the Perforce depot using your Perforce client.

- 3. As soon as the local working copy is on your computer, [associate your local directory with Perforce](#).

After that you will be able to open source files for edit, and perform the usual Perforce-related tasks using PhpStorm.

See Also

Concepts:

- [Supported Version Control Systems](#)

Reference:

- [Version Control Reference](#)
- [Perforce Options Dialog](#)

External Links:

- <http://www.perforce.com/perforce/doc.073/manuals/p4guide/index.html>

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

2.1+

Enabling and Configuring Perforce Integration

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

The Perforce Integration is disabled by default. If you want to perform Perforce-related operations right from PhpStorm, enable the integration at the IDE level and associate the project root or specific directories with Perforce. The general procedure is described in the section [Enabling Version Control](#).

In this section:

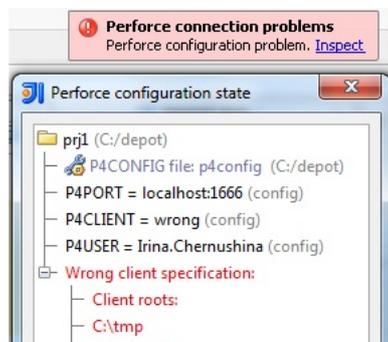
- [Enabling Perforce integration for a project or directory](#)
- [Configuring Perforce integration settings](#)

To enable Perforce integration for a project or directory

1. [Open the project settings](#) and click **Version Control**.
2. In the [Version Control](#) page, that opens, point to the desired root.
This can be **<Project Root>** or a [content root](#).
3. From the **VCS** drop-down list, choose **Perforce**.

Note

If you specify a wrong client workspace, and your project roots do not match with the workspace roots, PhpStorm displays a pop-up window with a warning:



Click **Inspect** to view and fix the discrepancies.

To configure Perforce integration

1. [Open the project settings](#), and then open the [VCSs](#) page by clicking **VCSs** under the **Version Control** node.
2. Open the [Perforce](#) page.
3. To establish live connection to the Perforce server, select the **Perforce is online** check box.
4. Specify which credentials you want to use for connecting to the Perforce server.
 - To use the connection settings from your **P4CONFIG** files, choose the **Use P4CONFIG or default connection** option.

Note

If you are using **P4CONFIG** files for configuration, PhpStorm shows what config files it has found and what other default settings are used. This way you can be sure that your **P4CONFIG** files are detected and taken into account.

- To configure connection manually, choose the **Use connection parameters** option and specify the following settings in the corresponding text boxes:
 1. The **Port** the Perforce client will listen to.

2. The **Client name**.
3. Your **User name** and **Password** to authenticate to the server.
5. To use ticket-based authentication, select the **Login authentication** check box. Otherwise, password-based authentication will be used. In either cases PhpStorm uses the login name and password specified in the dialog box or stored in the `P4CONFIG` files.
6. To attempt to log on the Perforce server without authentication, select the **Try to login silently** check box.
7. To have PhpStorm create a `P4.output` file and store the output of Perforce commands in it, select the **Dump Perforce Commands to:** check box.
8. Specify the path to the Perforce executable file. Click **Test Connection** to make sure your settings ensure successful connection.
9. In the **Timeout** field, specify the lapse of time in seconds during which PhpStorm waits for response from the server. If the server does not respond in due time, the user is prompted to disable integration.
10. To enable displaying the branch history of a specified file, including all file branch points, edits, and merges, select the **Show branching history** check box.
11. To have PhpStorm point at committed changes that are also integrated to other changelists and provide information on the target changelists that received the content in question, select the **Show integrated changelists in committed changes** check box.
12. To get the user interface for attaching and detaching Perforce jobs to changelists, select the **Enable Perforce Jobs Support** check box.

See Also

Concepts:

- [Supported Version Control Systems](#)

Reference:

- [Version Control Reference](#)
- [Version Control](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi> 
 - <http://youtrack.jetbrains.com/issues/WI> 
-

Handling Modified Without Checkout Files

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

If you are going to modify or delete a file under Perforce version control, the read-only status of such file should be removed. PhpStorm takes care of automatically making files writable. However, you can change read-only status manually, which may happen in a number of ways; for example:

- [With the Clear Read-Only Status dialog enabled](#), you make a file writable using file system.
- When a read-only file is opened in the editor, you double-click lock icon  in the status bar.
- You remove read-only attribute externally, using file properties.

In these cases, the file gets status `Modified without checkout` and appears in the [Local tab of the Changes tool window](#).

To resolve modified without checkout files

1. In the [Local tab of the Changes tool window](#), expand `Modified without Checkout` node, and select the desired file.
2. On the context menu of the file, choose **Check Out**. The file becomes writable, and moves to the active changelist.

See Also

Procedures:

- [Changing Read-Only Status of Files](#)

Reference:

- [Version Control Reference](#)
- [Version Control](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi> 
 - <http://youtrack.jetbrains.com/issues/WI> 
-

Integrating Perforce Files

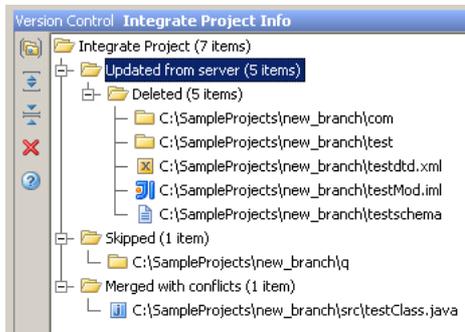
[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

You can merge changes between the branches into your local working copy, using the branch specification, or a changelist.

Note

Integrate Project command is available for both Perforce and Subversion integrations.

Integration results display in the **Integrate Info** tab of the Version Control tool window. Context menu of a file enables you to compare versions, view history and annotations, browse changes and more.



To integrate a Perforce branch into a project

1. On the main menu, choose **VCS | Integrate Project**.
2. In the Integrate dialog box, select the **Perforce** tab (if both Perforce and Subversion integrations are used in this project).
3. Select the sources to be merged, and the desired revision.
4. Define VCS-specific merge options.
5. Click **OK**.

See Also

Procedures:

- [Integrating SVN Projects or Directories](#)

Reference:

- [Version Control Reference](#)
- [Perforce Options Dialog](#)
- [Version Control Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Resolving Conflicts with Perforce Integration

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

As described in the section [Resolving Conflicts](#), the conflicts might occur in course of team work. Perforce integration makes use of the following commands:

- **Resolve** enables you to resolve a conflict to a specific file.
- **Resolve All** applies to all files in a changelist that have `merge` status.

To resolve conflicts for the files under Perforce version control

1. In the [Local tab of the Changes tool window](#), select a conflicting file, or a whole conflict node.
2. On the context menu of the selection, choose **Resolve** or **Resolve All**. Files Merged with Conflicts dialog box appears.
3. If you want to accept server version and overwrite your local changes, click **Accept Theirs**. If you want to force your changes to the repository, click **Accept Yours**. Clicking **Merge** opens the merge tool, where you can [accept or discard](#) each change individually.
4. Once the conflicts are successfully resolved, [commit](#) your local version to the repository.

See Also

Procedures:

- [Resolving Conflicts](#)
- [Checking in Files](#)

Reference:

- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Showing Revision Graph and Time-Lapse View

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Using Perforce integration, you can view the `Revision Graph` or `Time-lapse View` for a file without leaving the IDE, provided that `p4v.exe` is installed on your computer.

To show revision graph or time-lapse view for a file

1. Select the desired file in any navigation tool window, or open it in the editor.

2. On the main Version Control menu, or on the context menu of the selection, choose **Perforce**, and then select **Revision Graph**, or **Time-Lapse View** on the submenu.
3. With the first invocation, enter password in the login dialog box.

See Also

Reference:

- [Version Control Reference](#)
- [Version Control](#)

External Links:

- <http://www.perforce.com/perforce/products/p4v.html> 

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Using Multiple Perforce Depots with P4CONFIG

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

If your project contains directories that are stored in the different Perforce depots, you might need to switch between them. PhpStorm uses P4CONFIG to automatically switch to the respective depot as you use a Perforce-versioned directory.

P4CONFIG is an environment variable that contains the name of P4CONFIG file without a path. If a certain directory is associated with Perforce, PhpStorm seeks for P4CONFIG file in this directory and its parents; if the file is not found, it is sought in the bin directory of PhpStorm installation. When a P4CONFIG file is found, PhpStorm uses the settings contained therein, to connect to the respective Perforce depot.

A sample P4CONFG file might consist of such lines:

```
P4CLIENT=MyClient
P4USER=MySelf
P4PORT=ida:3456
```

To use multiple Perforce depots in a project, follow these general steps

1. Create a P4CONFIG file in each directory associated with Perforce.
2. Create environment variable P4CONFIG that contains file name without a path.

See Also

Reference:

- [Version Control Reference](#)
- [Version Control](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Working Offline

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

The Perforce plugin keeps a log of VCS operations performed while offline, and replays the log when the user comes back online. The log of operations is stored in the `.iws` file and persists between PhpStorm restarts.

While offline, you can perform the following operations, which will be automatically replayed in online mode:

- Edit
- Add/Copy
- Delete
- Move/Rename
- Revert
- Move to another changelist
- View Committed/Incoming changes (displaying cached information only).

The following operations are not supported in offline mode: update, commit, integrate, tracking of the unversioned, locally deleted and *modified without checkout* files (unversioned files are shown as *unchanged*), and any other operations that require server connection.

You can go to the offline mode in two ways

- **Automatically**, when the Perforce server becomes unavailable. PhpStorm switches to the offline mode automatically, and displays an offline notification in a pop-up window. To enable this behaviour, select the **Switch to offline mode automatically if Perforce is unavailable** check box in the [Perforce](#) page of the [Settings](#) dialog box.
- Manually at anytime, by choosing **VCS | Perforce** and select **Work Offline** on the main menu.

To return to the online mode, do one of the following

- Choose **VCS | Perforce** and clear **Work Offline**.
- In the offline notification pop-up window, click the **Go online** link.

Tip

The performance of Perforce integration in the offline mode is considerably better than in the online mode (because no server calls are required), so you might want to use offline mode even if you are not actually offline.

See Also

Reference:

- [Version Control Reference](#)
- [Changes Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

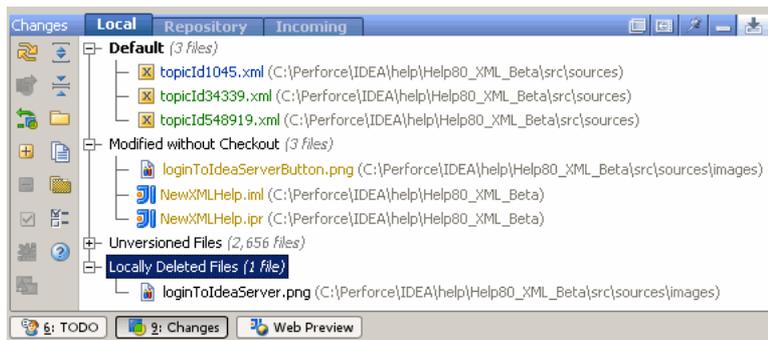
Checking Perforce Project Status

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Besides indicating the [current file status](#) relative to the repository, PhpStorm integration with Perforce provides you with the accumulated view of the project files' statuses.

To view differences between the current state of the project files and the repository

1. Open the required project.
2. On the main menu, choose **VCS | Refresh File Status**.
3. Switch to the **Changes** tool window, tab [Local](#).



The status of each file is indicated by the [color](#) in which the path to the file is displayed.

See Also

Procedures:

- [Viewing File Status](#)

Reference:

- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Attaching and Detaching Perforce Jobs to Changelists

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

The Perforce integration with PhpStorm provides you with a user interface for attaching and detaching Perforce [jobs](#) to changelists.

Tip

To get access to this functionality, select the **Enable Perforce Jobs Support** check box on the [Perforce](#) page of the [Settings](#) dialog box.

Note

The integration does not support creation of Perforce jobs.

From this topic you will learn how to:

- Attach jobs to changelists:
 - [At any stage](#) of your work from the [Local](#) tab.
 - [During commit](#) from the [Commit Changes](#) dialog box.
- Find the desired job to attach to a changelist using:
 - The [standard search](#) functionality, which enables you to specify numerous search criteria and thus to flexibly configure the search procedure.
 - The [quick search](#) functionality, which enables you to have the found job linked to the changelist automatically.

Tip

This functionality is helpful when you need to attach only one job to a changelist and you either know the exact name of the desired job or at least can specify a search pattern for the name.

- [Detach](#) jobs from changelists.

To find and link a job at any stage of your work

1. Open the [Local](#) tab of the [Changes](#) tool window.
2. Select the changelist you want to link a job to.
3. From the context menu of the changelist, choose **Edit Associated Jobs**. The [Edit Jobs Linked to Changelist](#) dialog box opens.
4. Find the desired job. Do one of the following:
 - To use the [standard search](#) functionality, click the  button.
 - To use the [quick search](#) functionality, click the  button.
5. View the details of the found job and click OK.

To find and link a job during commit

1. In the [Local](#) tab of the [Changes](#) tool window, select the changelist you want to link a job to and open the [Commit Changes](#) dialog box.
2. Find the desired job using the controls in the **Jobs** area. Do one of the following:
 - To use the [standard search](#) functionality, click the  button.
 - To use the [quick search](#) functionality, click the  button.
3. View the details of the found job and continue the [commit](#) procedure.

To find a job using the standard search functionality

1. In the [Edit Jobs Linked to Changelist](#) dialog box or in the **Jobs** area of the [Commit Changes](#) dialog box, click the  button.

Note

Which dialog box you are currently in depends on whether you are linking jobs [during commit](#) or at any [other stage](#) of work.

2. In the [Link Job to Changelist](#) dialog box that opens, specify the desired search parameters.

Tip

At least one of the fields should be filled in.

3. Click the **Search** button. The jobs that match the specified criteria are shown in the **Search Results** list. To view the details of a job, select it in the list.
4. Select the desired job and click OK. The dialog box closes and you return to the dialog box where you started the search:
 - The [Commit Changes](#) dialog box, if you are linking jobs [during commit](#).
 - The [Edit Jobs Linked to Changelist](#) dialog box, if you are linking jobs at any [other stage](#) stage of work.

To quickly find and link one job

1. Open the [Edit Jobs Linked to Changelist](#) dialog box or switch to the **Jobs** area of the [Commit Changes](#) dialog box.

Note

The dialog box to be used depends on whether you are linking jobs [during commit](#) or at any [other stage](#) of work.

2. In the text box, type the desired job name search pattern and click the  button. The job is found and automatically linked to the current changelist.

Note

If no job matching the specified pattern is found, the corresponding information message is displayed.

Warning

The details of jobs that are found and linked through the quick search functionality are available only in the [Edit Jobs Linked to Changelist](#) dialog box.

To detach a job from a changelist

- In the [Edit Jobs Linked to Changelist](#) dialog box or in the **Jobs** area of the [Commit Changes](#) dialog box, select the desired job and click the  button.

Note

The dialog box to be used depends on whether you are detaching jobs [during commit](#) or at any [other stage](#) of work.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Checking in Files](#)
- [Using Perforce Integration](#)

Reference:

- [Version Control Reference](#)
- [Local Tab](#)
- [Commit Changes Dialog](#)
- [Edit Jobs Linked to Changelist Dialog](#)
- [Link Job to Changelist Dialog](#)

External Links:

- <http://www.perforce.com/perforce/doc.081/manuals/cmdref/jobs.html> [↗]

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> [↗]
- <http://youtrack.jetbrains.net/issues/WI> [↗]

Using Mercurial Integration

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

With the [Mercurial](#) [↗] integration enabled, you can perform basic Mercurial operations from inside PhpStorm.

Using the Mercurial integration requires the following prerequisites:

- Mercurial is installed on your computer.
- The location of the Mercurial executable file is correctly specified on the **Mercurial** page of the **Settings** dialog box.
- Mercurial [integration is enabled](#) for the current project root or directory.

If you want to use a remote repository, create a Mercurial hosting account first. You can access the remote repository through a pair of `ssh` keys or apply the username/password and keyboard interactive authentication methods supported by the Mercurial integration.

Note

When the Mercurial integration with PhpStorm is enabled, the **Mercurial** item appears on the **VCS** menu.

Tip

When using the Mercurial integration, it is helpful to open the [Version Control](#) tool window. Its [Console](#) tab displays the following data:

- All the commands generated based on the settings you specify through the PhpStorm user interface.
- Information messages concerning the results of executing generated Mercurial commands.
- Error messages.

In this section:

- [Adding Files to a Local Mercurial Repository](#)
- [Setting up a Local Mercurial Repository](#)
- [Switching Between Working Directories](#)
- [Updating a Local Mercurial Repository \(Pull\)](#)
- [Uploading a Local Mercurial Repository \(Push\)](#)

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Version Control with PhpStorm](#)

Reference:

- [Version Control Reference](#)
- [Mercurial](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Adding Files to a Local Mercurial Repository

Previous | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts:  

After a Mercurial repository for a project is [initialized](#), you need to add the project data to it.

Tip

- If you have specified Mercurial as the version control system for your project in the **Settings** dialog box, tab [Version Control](#), PhpStorm suggests to put each new file under Mercurial control during the file creation.

To have Mercurial ignore some types of files, [configure files to ignore](#).

- You can [add all unversioned](#) files to Mercurial control or [select files to add](#).

To add all currently unversioned files to mercurial control

1. Switch to the [Changes](#) tool window.
2. Navigate to the **Unversioned Files** node and choose **Add to VCS** from the context menu.

To add specific file(s) to a local mercurial repository, do one of the following:

- Switch to the [Changes](#) tool window, expand the **Unversioned Files** node, and select the files to be added. From the context menu, choose **Add to VCS**.
- Switch to the [Project](#) tool window and select the files to be added. From the context menu, choose **Mercurial | Add to VCS** or press `Git.AddGit.Add`.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Version Control with PhpStorm](#)
- [Adding Files to Version Control](#)

Reference:

- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Setting up a Local Mercurial Repository

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Although Mercurial provides high flexibility in arranging data and your work with repositories, the following scenarios are most commonly used for setting up a local Mercurial repository:

- [Clone](#) an existing remote repository and create a new project with the downloaded data.
- [Create a local repository](#) which you can [push](#) to a remote location later, if necessary.

To clone a remote mercurial repository

1. On the main menu, choose **VCS | Checkout from Version Control | Mercurial**. The **Clone Mercurial Repository** dialog box opens.
2. In the **Mercurial Repository URL** text box, type the URL of the remote repository which you want to clone.
3. Click the **Test Repository** button next to the **Mercurial Repository URL** text box to check that connection to the remote repository can be established successfully.
4. In the **Parent Directory** text box, specify the directory where you want PhpStorm to create a folder for your local Mercurial repository. Use the **Browse** button  button, if necessary.
5. In the **Directory Name** text box, specify the name of the new folder into which the repository will be cloned. Click **Clone**.
6. Create a new project based on the cloned data by accepting the corresponding suggestion displayed by PhpStorm.

To create a local mercurial repository

1. Open the project you want to store in a repository.
2. On the main menu, choose **VCS | Import into Version Control | Create Mercurial Repository**. The [Create Mercurial Repository](#) dialog box opens.
3. Specify the location of the new repository.
 - To have the repository created in the project root, choose the **Create repository for the whole project** option. PhpStorm will create the `.hg` directory in the project root folder.

Note

This option is selected by default.

- To have a new repository created in another location, choose the **Select where to create repository** option and specify the path to the repository location in the text box below. Type the path manually or click the **Browse** button  and choose the relevant folder in the **Select directory for hg init** dialog box that opens.

Warning

1. Mercurial does not support external paths. So if you choose another directory, note that it must contain the tree where the project root resides.
 2. If you choose a directory which is already under Mercurial control, PhpStorm opens the Directory Is Under hg dialog box, where you can choose to create a repository in the specified location or to stay in the parent repository.
4. Put the required [files under Mercurial](#) version control. The files appear in the [Changes](#) tool window under the **Default** node.

Tip

If you specify Mercurial as the version control system for a directory in the [Version Control](#) dialog box, PhpStorm will suggest to put each new file in this directory under Mercurial control.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Version Control with PhpStorm](#)
- [Adding Files to Version Control](#)
- [Getting Local Working Copy of the Repository](#)

Reference:

- [Version Control Reference](#)
- [Clone Repository Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Switching Between Working Directories

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

The Mercurial integration with PhpStorm provides the possibility to switch update the repository's [working directory](#) to the specified [changeset](#) or a specific [line of development](#). Changesets can be identified by their hashes or by previously assigned [tag identifiers](#).

To switch to another working directory

1. On the main menu, choose **VCS | Mercurial | Update to**.
2. In the [Switch Working Directory Dialog](#) dialog box that opens, specify the target working directory:
 - To switch to another line of development, choose **Branch** and select the desired branch from the drop-down list.
 - To switch to a changeset to which you have previously assigned a tag identifier, choose **Tag** and select the desired tag from the drop-down list.
 - To switch to a changeset identified by a hash, choose **Revision** and type the desired revision number in the text box.
3. To have any local changes overwritten, select the **Overwrite locally modified files (no backup)** check box.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Using Mercurial Integration](#)
- [Version Control with PhpStorm](#)

Reference:

- [Switch Working Directory Dialog](#)
- [Mercurial Reference](#)
- [Version Control Reference](#)

External Links:

- <http://www.selenic.com/mercurial/hg.1.html#update>

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Updating a Local Mercurial Repository (Pull)

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Refreshing a local Mercurial repository with the changes from the remote repository (**Pu1.1**) involves retrieving changes and applying them to the local data. The Mercurial integration with PhpStorm

provides interface for specifying the mandatory `pull` settings and for customizing the `pull` procedure.

To pull changes from a remote repository

1. On the main menu, choose **VCS | Mercurial | Pull Changesets**. The `Pull` dialog box opens.
2. Specify the required URL address of the source remote repository.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Updating Local Information](#)
- [Version Control with PhpStorm](#)

Reference:

- [Version Control Reference](#)
- [Pull Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Uploading a Local Mercurial Repository (Push)

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Before uploading (`Pushing`) changes from your local Mercurial repository to a remote repository make sure that:

- You have a Mercurial hosting account.
- The target remote Mercurial repository is accessible for your account.
- You have a pair of `ssh` keys to access the remote repository.

To push changes from a local mercurial repository

1. On the main menu, choose **VCS | Mercurial | Push Changeset**. The `Push` dialog box opens.
2. In the **Destination Repository URL** text box, specify the location of the remote repository to push.
3. Specify the changes to upload:
 - To have a `branch` uploaded entirely, select the **Branch** check box and choose the relevant branch from the drop-down list.
 - To have a specific `changeset` uploaded, select the **Revision** check box and specify the desired revision number in the text box. By default, the latest revision (`tip`) is uploaded.
4. To push a locally created branch or to avoid updating before upload, select the **Force Push** check box.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Using Mercurial Integration](#)
- [Version Control with PhpStorm](#)

Reference:

- [Push Dialog](#)
- [Mercurial Reference](#)
- [Version Control Reference](#)

External Links:

- <http://www.selenic.com/mercurial/hq.1.html#push>

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Using Subversion Integration

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

When the Subversion integration is `enabled`, the `Subversion` menu item is added to the main `VCS` menu, to the context menus of the editor and Project tool window, and the project artifacts are highlighted according to their status. Results of modifications are shown in the `Changes` tool window.

Note

PhpStorm currently supports integration with Subversion 1.6. The main benefit is detecting `tree conflicts` of various types with the possibility to mark them as resolved.

Note

The information provided in this section assumes that you are familiar with the basics of Subversion.

See Also

Concepts:

- [Supported Version Control Systems](#)

Reference:

- [Version Control Reference](#)
- [File Status Highlights](#)
- [Changes Tool Window](#)

External Links:

- <http://subversion.tigris.org/>
- <http://svnbook.red-bean.com/en/1.1/svn-book.html>

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>
-

Authenticating to Subversion

Previous | [Next](#) | [See Also](#) | [Comments](#)

Subversion server does not require user authentication on every request. When you use the Subversion integration from within PhpStorm, you only need to answer the authentication challenge of the server, if it is required by the authentication and authorization policies. The successful authentication results in saving your credentials on disk, in `~/.subversion/auth/` on Unix systems or `<USER HOME>/.subversion_IDEA` on Windows and MacOS.

When an authentication challenge comes from the server, the credentials are sought for in the disk cache; if the appropriate credentials are not found, or fail to authenticate, the users is prompted to specify the login name and password.

If necessary, you can opt to delete all credentials stored in cache, for http, svn and ssh+svn protocols.

To delete credentials from disk

1. On the main menu, choose **File | Settings | Project Settings | Version Control for Windows and Linux or PhpStorm | Preferences | Project Settings | Version Control for Mac OS**.
2. In the [Version Control dialog box](#), click **Configure VCS**.
3. In the Version Control Configurations dialog, select **Subversion**.

Tip

Alternatively, In the [Version Control dialog box](#), select a directory associated with Subversion, and click the Browse button .

4. In the Subversion-related options page, click **Clear authentication cache**.

See Also

Concepts:

- [Supported Version Control Systems](#)

Reference:

- [Version Control Reference](#)
- [Authentication Required](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>
-

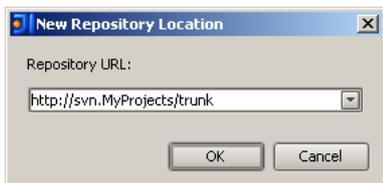
Browsing Subversion Repository

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

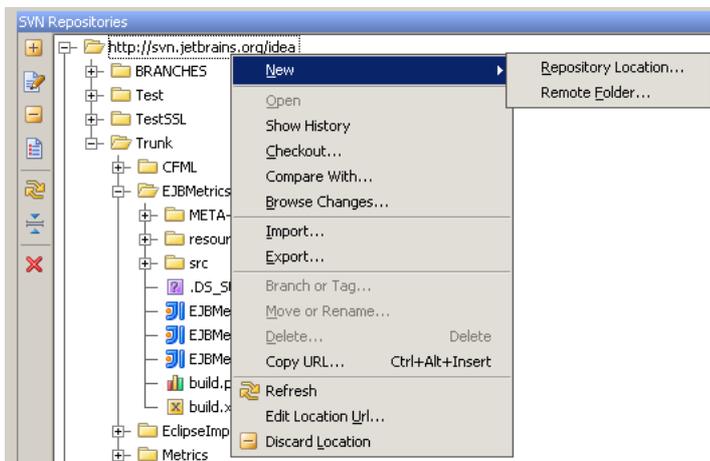
Prior to checking files out, you can browse the contents of the Subversion repository. Browsing contents of the repository is always available, even when Subversion is not enabled in project. All you need is a valid user account.

To browse contents of subversion repository

1. On the main menu, choose **VCS | Browse Subversion Repository**.
2. Specify repository URL in the New Repository Location dialog box:



3. View contents of the repository in the SVN Repositories tool window:



Tip

SVN Repositories browser enables you to add or discard repository locations, view history of files and folders, check out files and folders, navigate to source code, browse changes, create branches or tags etc.

See Also

Procedures:

- [Importing a Local Directory to Subversion Repository](#)
- [Exporting Information from Subversion Repository](#)
- [Checking Out Files from Subversion Repository](#)
- [Browsing Changes](#)
- [Viewing Changes History for a File or Selection](#)

Reference:

- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Checking Out Files from Subversion Repository

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Checkout helps you obtain a [local working copy of the repository](#), which you can edit as required. After making the necessary changes, you can publish results by [committing, or checking in](#) changes to the repository.

This section describes Subversion-specific checkout procedure.

To obtain a local working copy of a subversion repository

1. On the main menu, choose **VCS | Checkout from Version Control**.
2. On the submenu, choose **Subversion**.
3. In the [Check Out From Subversion dialog box](#), expand the desired repository location and select the element to be checked out.
4. Click **Checkout** button.

Tip

This action is also available in the SVN Repositories browser. Right-click the desired directory and choose the command on the context menu.

5. In the **Select Path** dialog box, specify the destination directory where the local copy of the repository files will be created and click **OK**.

Note

If you are checking out sources for an existing project, the destination folder should be below a project [content root](#).

6. In the [SVN Checkout Options](#) dialog box, specify the following settings:
 - Revision to be checked out (HEAD or a selected revision).
 - Whether you need to check out the nested directories.
 - Whether you need to include the external locations.

Click **OK**.

7. PhpStorm suggests to create a project created based on the checked out sources. If you accept the suggestion, the [New Project from Existing Code Wizard](#) starts.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Using Subversion Integration](#)
- [Version Control with PhpStorm](#)
- [Creating New Project from Existing Source Code](#)

Reference:

- [Version Control Reference](#)
- [Check Out from Subversion Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> [ⓘ]
- <http://youtrack.jetbrains.net/issues/WI> [ⓘ]

Cleaning up Local Working Copy

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

The `cleanup` command in Subversion can be helpful in the following cases:

- Your local working copy is in an inconsistent state because a Subversion command has been interrupted.
- The timestamp of a file has changed while its content remains intact.

To clean up the local working copy, do one of the following

- Select the desired file or directory in the [Project](#) tool window and choose **Subversion | Cleanup** on the context menu of the selection.
- Open the desired file in the editor and choose **VCS | Subversion | Cleanup** on the main menu.
- Select the desired file or directory in the [Local](#) tab of the [Changes](#) tool window and choose **Subversion | Cleanup** on the context menu of the selection.

See Also

Concepts:

- [Supported Version Control Systems](#)

Reference:

- [Version Control Reference](#)

External Links:

- <http://subversion.tigris.org/> [ⓘ]

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> [ⓘ]
- <http://youtrack.jetbrains.net/issues/WI> [ⓘ]

Comparing with Branch

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

In addition to the common file versions comparison operations, Subversion integration provides a special command that enables you to compare a file from your local working copy with the version in a certain branch.

To compare a file with the specified branch version

1. Select the desired file in the Project tool window, or open it in the editor.
2. On the main **VCS** menu, or on the context menu of the selection, choose **Subversion | Compare with Branch**.
3. In the Compare with Branch pop-up window, select the desired branch. The Compare with Branch dialog in the form of a [Differences Viewer](#) appears.

Tip

This action is also available in the SVN Repositories browser. Right-click the desired directory and choose the command on the context menu.

See Also

Reference:

- [Version Control Reference](#)
- [Differences Viewer](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Configuring Format of the Local Working Copy

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

A working copy is a directory that contains a collection of files, which you can use as your private work area, and some extra files, created and maintained by Subversion. In particular, each directory in your working copy contains an administrative directory named `.svn`.

You can have local working copies created with Subversion 1.3, 1.4, 1.5, or 1.6. PhpStorm handles all these formats, giving you a choice to upgrade to the new format or preserve the legacy one.

Switching to another Subversion format is always associated with a specific working copy (directory) under Subversion control. In other words, one cannot upgrade to a newer Subversion format at the IDE level but only for a directory under Subversion control.

Tip

To check the VCS association, open the [Settings - Version Control](#) dialog box where all the mappings between the project directories (or the entire project) and VCSs are listed.

To change the format of a working copy

1. Open the [Changes](#) tool window by doing one of the following:
 - On the main menu, choose **View | Tool Windows | Changes**.
 - Press `Alt+9Meta 9`.
2. Switch to the [Subversion Working Copies Information](#) tab.

Tip

This tab is only available, when the current project sources are entirely or partially under Subversion control.

3. Scroll to the information on the required directory and click the **Change** link next to the **Format** read-only text box.
4. In the **Convert Working Copy Format** dialog box, that opens, select the desired format option.

See Also

Concepts:

- [Supported Version Control Systems](#)

Reference:

- [Changes Tool Window](#)
- [Version Control Reference](#)
- [Subversion Working Copies Information Tab](#)

External Links:

- <http://subversion.tigris.org/>

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Configuring HTTP Proxy

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Subversion stores http proxy settings in the `servers` file in the user's runtime configuration area (`~/.subversion/auth/` in Unix systems or `<USER HOME>/Application Data/Subversion/auth/` in Windows).

You can configure Subversion proxy settings in two ways:

- [Edit the servers file](#) manually.
- Configure proxy directly from PhpStorm.

To configure proxy settings for subversion

1. Open the [Project settings](#) dialog box.
2. Below the **Version Control** node, click **VCSs**, then click **Subversion**.
3. In the **Subversion** dialog box that opens, click the **Edit Network Options** button and specify the proxy settings in the [Edit Subversion Options Related to Network Layers](#) dialog box that opens.

See Also

Reference:

- [Version Control Reference](#)
- [Edit Subversion Options Related to Network Layers Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>
-

Configuring Subversion Repository Location

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

The Subversion repository locations are global PhpStorm properties. It means that the configured repository location will be available regardless of a project being open, which is useful if you need to check out an entire project from Subversion. You can define multiple Subversion repository locations for future use.

To configure a subversion repository location

1. Open SVN Repositories tool window. You can do that in a number of ways, for example:
 - Browse repository using **VCS | Browse Subversion Repository** command.
 - Show the tool window using **View | Tool Windows | SVN Repositories** command.
2. In the SVN Repositories tool window, do one of the following:
 - On the context menu of the window, choose **New | Repository Location**.
 - On the toolbar, click add button.

Tip

You can also configure repository location from the [Import into Subversion](#), or from [Check Out From Subversion](#) dialog box. Use the add button, which is available on the toolbar of the dialog box.

3. In the New Repository Location dialog box, specify URL of the repository, and click OK.

See Also

Procedures:

- [Checking Out Files from Subversion Repository](#)

Reference:

- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>
-

Configuring Subversion Branches

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

PhpStorm integration with Subversion provides the possibility to compose a list of branches you work with. This list will be displayed every time you select a branch to perform any kind of operation on, for example, to synchronize your working copy, compare branches, etc. This saves your time because you no longer need to scroll through the entire tree of branches in the repository. Another advantage is that you can reference any branch from your list by the branch name instead of specifying its location through the entire URL address.

Branches are configured in the [Configure Subversion Branches](#) dialog box.

To open the configure subversion branches dialog box, do one of the following

- 1. In the [Changes](#) tool window, open the [Repository](#) tab, click the **Highlight Integrated** toolbar button  to have the **Merge Info** pane displayed.
 2. Click the **Browse** button  or press **Shift+EnterShift Enter**.
 3. In the [Select Branch](#) pop-up dialog box that opens, choose **Configure Branches**.
- 1. On the main menu, choose **VCS | Subversion | Update File/Directory**, or **VCS | Update Project**.
 2. In the [Update Project/Directory](#) dialog box that opens, click the **Browse** button  or press **Shift+EnterShift Enter** to select the path to the target branch.
 3. In the [Select Branch](#) pop-up dialog box that opens, choose **Configure Branches**.

To configure branches

1. Open the Configure Subversion Branches dialog box.
2. In the **Trunk Location** text box, specify the URL address that identifies the trunk of your repository. Type the URL address manually or click the **Browse** button  and appoint the trunk manually in the [Select Repository Location](#) dialog box that opens. The dialog box shows a tree of all branches and tags under the [repository root](#).
3. In the **Branch Locations** area, compose a list of branches that you need in your work.
 - To have an item included in the list, click the **Add** button and choose the required branch in the [Select Repository Location](#) dialog box that opens.
 - To exclude an item from the list, select the required item and click the **Remove** button.

Now you can reference any branch from the list by the branch name instead of specifying the entire URL address.

See Also

Concepts:

- [Supported Version Control Systems](#)

Procedures:

- [Creating Branches and Tags](#)
- [Using Subversion Integration](#)

Reference:

- [Configure Subversion Branches](#)
- [Repository and Incoming Tabs](#)
- [Subversion Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>
-

Creating Branches and Tags

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Subversion integration enables you to create branches or tags on the basis of your local working copies or repository versions.

To create branch or tag in a subversion repository

1. On the main VCS menu, choose **Subversion | Branch or Tag**. Alternatively, select the source folder and choose the command on the context menu. The [Create Branch or Tag](#) dialog box opens.
2. In the **Copy From** section, specify the source folder that will be copied to a branch or tag. You can opt to use your local working copy, or a repository location.
If a repository location is selected as the source:
 - Click to fill the **Repository Location** field with the path to the project location.
 - Specify the revision to base the new branch on. This can be the HEAD revision, or a revision with the specified number. If the **Specified** option is selected, type the revision number in the text field, or click  and find the desired revision in the Subversion Changes Browser.
3. In the **Copy To** section, specify the destination where the branch or tag will be created. If you use the base URL, specify the name of the new branch or tag. If you opt to create a branch or tag in another repository location, type its URL in the text field, or click  and select the destination from the [Select Repository Location](#) dialog box.
4. Optionally, type a comment message and click **OK**.

See Also

Concepts:

- [Supported Version Control Systems](#)

Reference:

- [Version Control Reference](#)
- [Create Branch or Tag Dialog \(Subversion\)](#)
- [Select Repository Location Dialog \(Subversion\)](#)

External Links:

- <http://svnbook.red-bean.com/en/1.1/svn-book.html>

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>
-

Exporting Information from Subversion Repository

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

You might need to obtain a *clean* local copy of the Subversion working tree without `.svn` catalogs. Instead of checking files out, and then manually deleting administrative directories, you can use `Export` command, which is available in the Subversion repository browser.

To export a directory from subversion repository

1. [Open SVN Repositories browser](#).
2. Right-click a directory to be exported, and choose **Export** on the context menu.
3. In the Select Path dialog specify the destination directory, and click **OK**.
4. In the SVN Export Options dialog box, specify the following options that directly map to the options of SVN export command:
 - Export recursively
 - Replace existing files
 - Include externals
 - Override native EOLs

See Also

Procedures:

- [Browsing Subversion Repository](#)

Reference:

- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Importing a Local Directory to Subversion Repository

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

You can import an entire directory to your Subversion repository, provided that you have access rights. This action is helpful for putting a whole project under version control.

Note

Import to the repository is always available, even though Subversion is not enabled in project.

To import a directory into a subversion repository

1. On the main menu, choose **VCS | Import into Subversion**.
2. In the [Import into Subversion](#) dialog box, select the target Subversion repository location. If the desired target repository location does not exist, you can [configure a new one](#).
3. Click **Import**.

Tip

This action is also available in the SVN Repositories browser. Right-click the target directory and choose the command on the context menu.

4. In the Select Path dialog box, specify the directory to be imported, and click **OK**.

Tip

If you are importing a PhpStorm project, make sure that the project file (.ipr) is located under that folder.

5. In the **SVN Import Options** dialog box, showing the read-only target repository location, and an editable field with the source directory, specify the following import options:
 - Check the option **Import directories recursively**, if you want to upload the nested directories to the repository location.
 - Check the option **Include ignored resources**, if you want to upload files from the ignored list.
 - Enter your comment in the **Commit Message** field, or select one from the history drop-down list.
6. Click **OK**.

See Also

Reference:

- [Version Control Reference](#)
- [Import Into Subversion](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Integrating SVN Projects or Directories

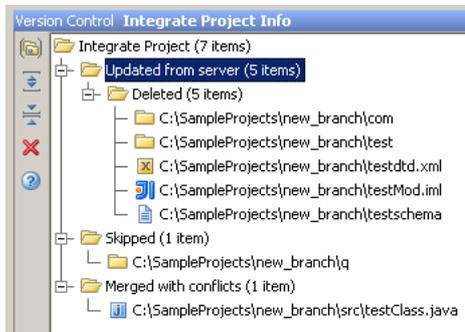
[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Integrating projects or directories in Subversion means merging differences between the two specified revisions into your working copy.

Note

Integrate Project command is available for both Subversion and Perforce integrations.

Integration results display in the **Integrate Info** tab of the Version Control tool window. Context menu of a file enables you to compare versions, view history and annotations, browse changes and more.



To integrate different sources into one subversion project

1. On the main menu, choose **VCS | Integrate Project**.
2. In the Integrate dialog box, select **Subversion** tab (if both Perforce and Subversion integrations are used in this project).
3. In the **Source 1** and **Source 2** fields, specify the sources to be merged, and the desired revision:
 - o In the text fields, type repository locations for both sources, or click the browse button to open the [Select Repository Location](#) dialog box.
 - o Specify the revisions to be integrated; if you click the **Specified** radio-button, type revision number, or click the browse button and select the desired revision from the Changes Browser.
4. Define the following merge options, which correspond to the Subversion merge options:
 - o **Try merge, but make no change**, which corresponds to the Dry Run option of Subversion. When this option is checked, you will be presented with the preview of diff operation to be used to perform integration.
 - o **Descend into child subdirectories** to perform operation recursively.
 - o **Run status after update** to perform status check.
5. Click **OK**.

See Also

Procedures:

- [Integrating Perforce Files](#)
- [Version Control Tool Window](#)

Reference:

- [Version Control Reference](#)
- [Integrate Project Dialog \(Subversion\)](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Locking and Unlocking Files and Folders

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Though Subversion integration enables you to successfully modify and merge files, changed by different team members, sometimes it makes sense to lock files (for example, images) to avoid overwriting changes.

To lock a file

1. Select the desired file or open it in the editor.
2. Choose **VCS | Subversion | Lock** on the main menu or **Subversion | Lock** on the context menu of the selection.
3. In the [Lock File](#) dialog box that opens specify the lock name.

Tip

To forcibly break the lock set by somebody else, select the **Steal Existing Lock** check box.

To remove lock from a file

1. Select the desired file or open it in the editor.
2. Choose **VCS | Subversion | Unlock** on the main menu or **Subversion | Unlock** on the context menu of the selection.

See Also

Procedures:

- [Handling Differences](#)

Reference:

- [Lock File Dialog \(Subversion\)](#)
- [Version Control Reference](#)
- [Changes Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Resolving Text Conflicts

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

If a conflict occurs in a file under Subversion, the conflict markers are added to the conflicting file, and three auxiliary unversioned files are created in your working copy:

- **filename.mine**: the copy of your local file without conflict markers.
- **filename.rOld**: the base revision you have last synchronized to.
- **filename.rNew**: the latest version on the server.

For resolving conflicts, Subversion integration suggests you to use **Resolve Text Conflict** command, or edit the conflicting file manually.

To resolve conflicts under subversion

1. In the [Local tab of the Changes tool window](#), select the conflicting file:



2. On the main **VCS** menu, or on the context menu of the selection, choose **Subversion | Resolve Text Conflict**. Files Merged with Conflicts dialog box appears.
3. If you want to accept the server version and overwrite your local changes, click **Accept Theirs**. If you want to force your changes to the repository, click **Accept Yours**. Clicking **Merge** opens the merge tool, where you can [accept or discard](#) each change individually. So doing, the file is automatically marked as resolved, and auxiliary files are deleted.
4. Once the conflicts are successfully resolved, commit your local version to the repository.

Tip

You can resolve a text conflict manually. Open conflicting file in the editor, and merge contents within the conflict markers as required, or copy one of the auxiliary files on top of your working file. After that, choose **Mark Resolved** on the Subversion menu, to clean up the auxiliary files from the working copy.

See Also

Procedures:

- [Resolving Conflicts](#)

Reference:

- [Version Control Reference](#)
- [Differences Viewer](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Sharing Directory

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm supports the useful sharing directories feature of Subversion. The Share Directory command applies to the content roots. When such content root is shared, a new directory is created in the specified location of the Subversion repository, checked out to the local directory you want to share, and the contents of the local directory is added to the repository. After that the local directory contains a valid working copy of the repository.

To share a directory via subversion repository

1. In the Project tool window, select an unversioned directory, which is a content root of a module associated with Subversion.
2. On the main menu, choose **Subversion | Share Directory**.
3. Specify the target repository location, where the directory should be shared.

See Also

Procedures:

- [Importing a Local Directory to Subversion Repository](#)

Reference:

- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Working with Subversion Properties for Files and Directories

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Subversion integration enables you to work with Subversion-specific properties without leaving PhpStorm.

Once defined, properties of a file or directory display in the SVN Properties view. In this view you can explore and change existing properties and their values, or create new ones, using the toolbar buttons or context menu commands.

This section describes how to:

- [View properties of a file or directory](#) from within PhpStorm.
- [Create a new property, or change value of an existing property](#).
- [Set up](#) the `keyword` property.
- [Delete property](#).

To view properties of a file or directory

1. In the [Project tool window](#), select the desired file or directory under SVN version control.
2. On the main **VCS** menu, or on the context menu of the selection, choose **Subversion | Edit Properties**. SVN Properties view displays properties of the selected file.
3. Use the toolbar buttons or context menu commands to create, edit or delete properties, as described in the procedures below.

Tip

If you want SVN Properties view to preserve its contents as you navigate through your project or edit files, make sure that the **Follow Selection** button  is not pressed; otherwise the view will show properties for the currently selected or edited file.

To create a new property, or set up value for an existing property

1. Open [SVN Properties view](#).

Tip

Use **Set property** command on the Subversion menu to define a single property.

2. Click add button  on the toolbar of SVN Properties view, or choose **Add property** command on the context menu. [Set Property](#) dialog box appears.
3. In the **Property name** field, type the name of the new property, or select one from the drop-down list.
4. Click **Set property value** radio-button, and specify the desired value in the text area below.
5. If you want to apply changes to all subdirectories of the selected directory, check the **Update properties recursively** option.
6. Click **OK**.

To set up svn:keywords property

1. In the SVN Properties view for a file, click .
2. In the SVN Keywords dialog box, check the keywords to be included in the property.
3. Click **OK**.

To delete a property in the svn properties view

1. Select a property to be deleted.
2. Click  on the toolbar.

To delete a property in the set property dialog box

1. In the **Property name** field, select the property to be deleted.
2. Click **Delete property** radio-button.
3. If you want this property to be deleted from all files and subdirectories of the selected directory, check the option **Update properties recursively**.
4. Click **OK**.

See Also

Reference:

- [Version Control Reference](#)
- [Set Property Dialog \(Subversion\)](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi> 
- <http://youtrack.jetbrains.com/issues/WI> 

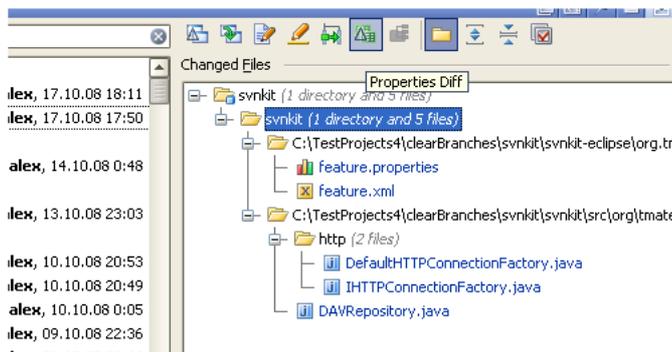
Viewing Differences in Properties

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

With PhpStorm, you can view differences in properties between a file in the [local copy and in the repository](#) or between [two revisions](#) of a file in the local copy.

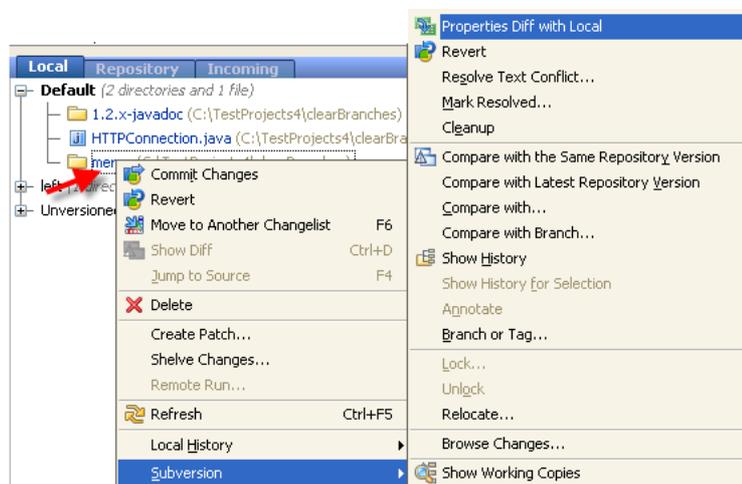
To view property difference between the local copy and the repository version

1. Open the Changes tool window and switch to the [Repository and Incoming](#) tab.
2. Select the file for which you want to view property differences and choose **Properties Diff** in the context menu or click the  button on the toolbar:

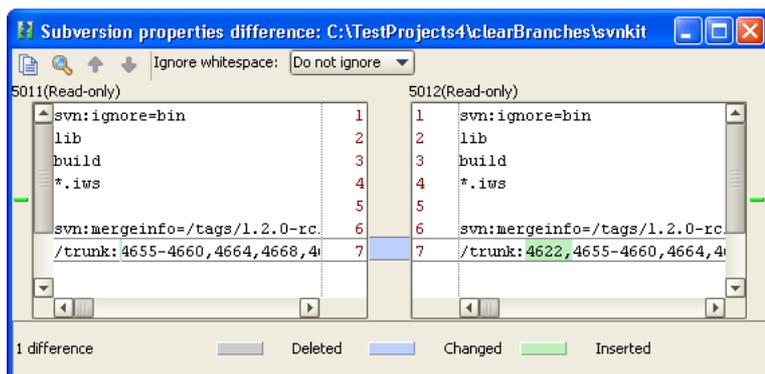


To view property difference between two revisions in the local copy

1. Open the Changes tool window and switch to the [Local](#) tab.
2. Select the file for which you want to view property differences between revisions and choose **Subversion | Properties Diff with Local** in the context menu.



The Subversion properties difference information is displayed:



See Also

Reference:

- [Version Control Reference](#)
- [Changes Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

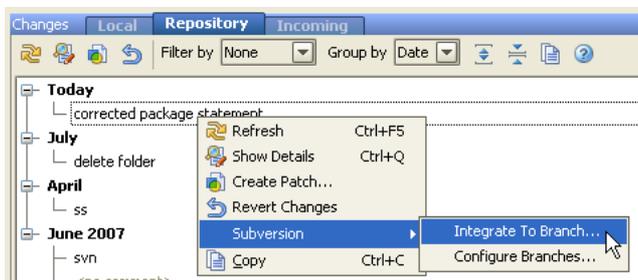
Integrating Changes to Branch

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Subversion integration with PhpStorm enables you to integrate changes into a branch of your choice and commit the integration results to the repository.

To integrate changes from one branch to another

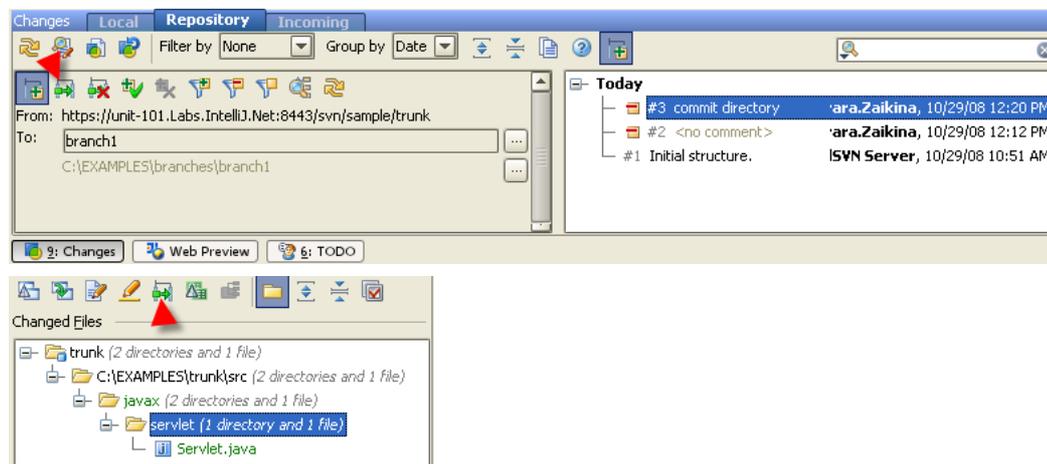
1. In the [Changes](#) tool window, switch to the [Repository tab](#).
2. Right-click the desired change list and select **Subversion | Integrate To Branch** in the context menu:



3. To integrate specific files from a changelist, select the file(s) in the **Changed Files** pane and select **Subversion | Integrate To Branch** in the context menu.

Note

If you are using SVN 1.5 or higher both on the server and in your local working copy, select the relevant changelist/file(s) in the **Changelists** or **Changed Files** pane and click the  button on the toolbar.



4. The [Integrate to Branch](#) dialog box opens displaying the URL addresses of the source and target branches and a list of available local working copies. In the **Integrate into Working Copy** list, select the path to the local working copy into which the changelist will be integrated.
5. To add a path to the list, click the  button.

Note

Make sure the specified working copy directory is under Subversion version control!

6. To remove a path from the list, click the  button.
7. To preview the merge result by enabling the `--dry-run` switch of the `svn` command, select the **Try merge, but make no changes** check box.

Note

If this check box is not selected, the sources are merged silently.

8. To view all the files with local modifications after update, select the **Run status after update** check box.

Note

Technically, this means that the `svn status` command will be applied after commit.

9. Click OK. The [Commit Changes](#) dialog box opens.
10. View the summary, specify the necessary option, and [run commit](#).
11. To view the integration results, open the [Version Control](#) tool window by choosing **View | Tool Windows | Version Control**.

Note

This information is available if you specified the **Run status after update** option in the **Integrate to Branch** dialog box.

See Also

Concepts:

- [Version Control with PhpStorm](#)
- [Local, Committed and Incoming Changes](#)

Procedures:

- [Checking in Files](#)
- [Using Subversion Integration](#)
- [Integrating Files and Changelists from the Changes Tool Window](#)

Reference:

- [Version Control Reference](#)
- [Repository and Incoming Tabs](#)
- [Integrate to Branch](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Viewing and Fast Processing of Changelists

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

If you use Subversion 1.5 or higher on the server and in your local working copy, you can derive advantage from the extended **Merge Info** functionality implemented through the **Merge Info** pane of the **Changes** tool window, **Repository** tab.

With this functionality, instead of browsing [all changelists](#) within a certain period, you can define a set of changelists to display. This is done by specifying a pair of branches in the repository (source and target) whereupon PhpStorm shows only those changelists from the source branch that have been integrated into the target branch.

Additionally, you can specify various filtering options to minimize the number of extraneous changelists.

Finally, integration and managing integration status are also available directly from the **Merge Info** pane.

The extended browsing functionality includes:

- [Defining the Set of Changelists to Display](#)
- [Filtering Out Extraneous Changelists](#)
- [Viewing and Managing Integration Status](#)
- [Integrating Files and Changelists from the Changes Tool Window](#)

Tip

Before enabling extended Merge Info functionality, make sure you are using SVN Server 1.5 or higher.

To enable merge info functionality

1. To use SVN 1.5 format in the local working copies, choose **VCS | Subversion** in the main menu, then choose **Show Working Copies** in the submenu.
2. In the **Subversion Working Copies Information** dialog box that opens, select the relevant working copy and click the **Change Working Copy Format** button.
3. In the **Convert Working Copy Format** dialog box, choose the **1.5 format** option and click **OK**.
4. To have the **Merge Info** pane displayed, switch to the **Changes** tool window, **Repository** tab, and click the **Highlight Integrated** button  on the toolbar.
5. On the **Merge Info** pane, configure the [source and target](#) branches.

See Also

Concepts:

- [Version Control with PhpStorm](#)
- [Local, Committed and Incoming Changes](#)

Reference:

- [Version Control Reference](#)
- [Repository and Incoming Tabs](#)
- [Integrate to Branch](#)

External Links:

- <http://svnbook.red-bean.com/nightly/en/svn-book.html#svn.branchmerge>
- <http://www.collab.net/community/subversion/articles/merge-info.html>

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Defining the Set of Changelists to Display

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

With the extended [Merge Info](#) functionality, you can restrict the set of changelists displayed in the [Changelists](#) pane to the changelists that have been integrated from one specific branch to another. These branches are referred to as "source" and "target" respectively.

To specify the branches involved

1. Open the **Changes** tool window and switch to the **Repository** tab.
2. To have the [Merge Info](#) pane displayed, click the **Highlight Integrated** button  on the toolbar.
3. In the **From** text box, specify the URL address of the source branch.
4. In the **To** text box, specify the path to the target branch.
If necessary, use the **Browse** button  to open the [Select Branch](#) dialog box.
5. Specify the path to the local [working copy](#) to which you will apply patches created based on selected changelists.
If necessary, use the **Browse** button  to open the [Configure Working Copy Paths](#) dialog box.

See Also

Concepts:

- [Version Control with PhpStorm](#)
- [Local, Committed and Incoming Changes](#)

Procedures:

- [Viewing and Fast Processing of Changelists](#)

Reference:

- [Version Control Reference](#)
- [Repository and Incoming Tabs](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Filtering Out Extraneous Changelists

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

With the extended [Merge Info](#) functionality, you can make the set of changelists displayed in the [Changelists](#) pane even easier to browse by applying additional filtering according to integration status, committer, or version control system used.

Tip

Filtering out integrated/not integrated changelists is available only when integration status is highlighted.

Otherwise the **Filter out integrated** button  and **Filter out not integrated** button  are unavailable. To enable integration status highlighting, click the **Highlight Integrated** button  on the toolbar.

To apply additional filtering options, do one of the following in the merge info pane

- To display only changelists that have not been integrated into the working copy, click the **Filter out integrated** button  on the toolbar.
- To display only changelists that have been integrated into the working copy, click the **Filter out not integrated** button  on the toolbar.
- To hide changelists that are [managed in another VCS](#) or are located under another root, click the **Filter out others** button  on the toolbar.
- To display only the changelists that were committed by a specific user, select **User** in the **Filter by** drop-down list. Then select the name of the required user.
- To display only the changelists applied to a specific module or folder, select **Structure** in the **Filter by** drop-down list. Then select the required location.
- To have the changelists grouped by users who committed them or by commit dates, select the relevant option in the **Group by** drop-down list.

See Also

Concepts:

- [Version Control with PhpStorm](#)
- [Local, Committed and Incoming Changes](#)

Procedures:

- [Viewing and Fast Processing of Changelists](#)

Reference:

- [Version Control Reference](#)
- [Repository and Incoming Tabs](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Viewing and Managing Integration Status

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

With the extended [Merge Info](#) functionality, you can view and update integration status of changelists.

Subversion stores information on whether a changelist has been integrated into the local working copy or not. Based on this data, PhpStorm informs you on the integration status of a specific changelist by displaying one of the following icons next to the changelist:

-  integrated
-  not integrated
-  integration status unknown
-  common history

To have the integration status displayed, click the **Highlight Integrated** button  on the **Merge Info** pane toolbar.

You can change the integration status of a changelist without actually integrating it into the working copy or reverting the previous integration. This will affect only the administrative data.

To toggle the integration status of a changelist

Do one of the following:

- To mark a changelist as **Integrated**, select the changelist and click the **Mark As Merged** button  on the **Merge Info** pane toolbar.
- To mark a changelist as **Not integrated**, select the changelist and click the **Mark As Not Merged** button  on the **Merge Info** pane toolbar.

See Also

Concepts:

- [Version Control with PhpStorm](#)
- [Local, Committed and Incoming Changes](#)

Procedures:

- [Viewing and Fast Processing of Changelists](#)

Reference:

- [Version Control Reference](#)
- [Repository and Incoming Tabs](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Integrating Files and Changelists from the Changes Tool Window

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

You can integrate changelists or files into your local working copy directly from the **Changes** tool window.

To integrate a changelist or a file into your working copy, complete the following general steps

- In the **Changelists** pane, select the required changelist.

Note

If necessary, you can select and integrate several changelists at a time.

- To integrate the entire changelist, click the **Integrate to Branch** button  on the **Merge Info** pane toolbar.
- To integrate a particular file from the selected changelist, select the file in the **Changed Files** pane and click the **Integrate to Branch** button  on the **Merge Info** pane toolbar.
- To revert the last integration of the selected changelist into the working copy, click the **Undo Integrate to Branch** button  on the toolbar.

See Also

Concepts:

- [Version Control with PhpStorm](#)
- [Local, Committed and Incoming Changes](#)

Reference:

- [Version Control Reference](#)
- [Repository and Incoming Tabs](#)
- [Integrate to Branch](#)

External Links:

- <http://svnbook.red-bean.com/nightly/en/svn-book.html#svn.branchmerge>
- <http://www.collab.net/community/subversion/articles/merge-info.html>

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

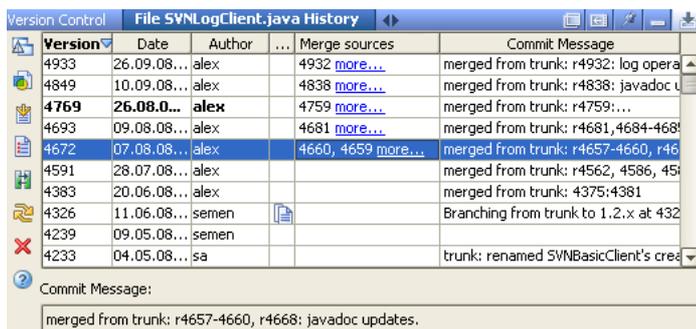
Viewing Merge Sources

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

For Subversion, the common [Viewing Change History](#) functionality is expanded with the possibility to view which revisions have been used in merges applied to a file. The merge sources can be presented as a tree of revisions.

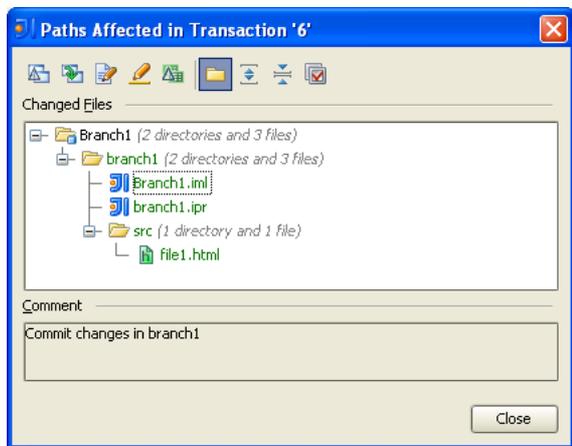
To view the merge source

1. Open the Changes tool window, [Repository](#) tab, and switch to the Changelists pane.
2. Select the required changelist, switch to the Changed Files pane, and select the required file.
3. Select **Show History** in the context menu or click the  button on the main editor toolbar. The **Version Control** tool window opens displaying the [change history](#) of the selected file. For each revision, the **Merge sources** read-only field shows the revisions that were used as sources for the merge.

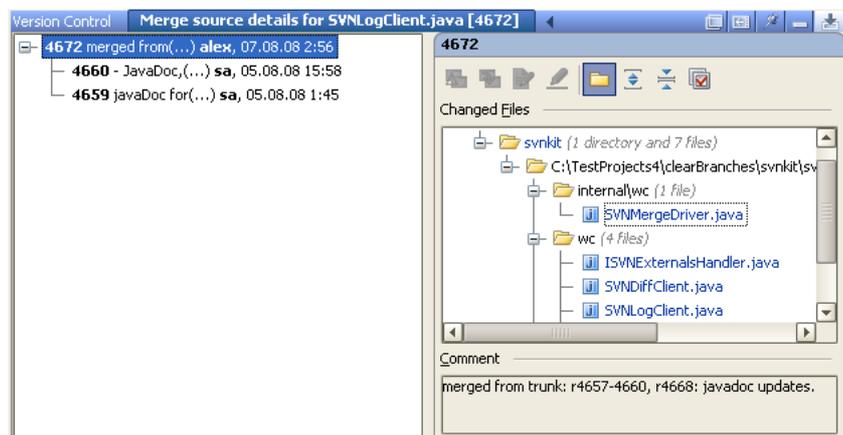


4. To view all the paths affected by a merge, select the relevant row and select **Show all paths affected in the selected transaction** in the context menu or click the  button on the tool window toolbar.

The following information dialog box appears:



5. To view the source details of a merge, select the relevant row and click **More**. The details are displayed in the following information dialog box:



See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Viewing Changes History for a File or Selection](#)

Reference:

- [Version Control Reference](#)
- [Version Control Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

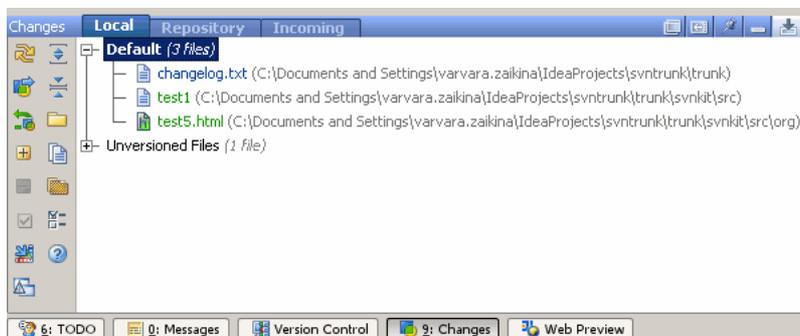
Checking SVN Project Status

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Besides indicating the [current file status](#) relative to the repository, PhpStorm integration with SVN provides you with the accumulated view of the project files' statuses.

To view differences between the current state of the project files and the repository

1. Open the required project.
2. On the main menu, choose **VCS | Refresh File Status**.
3. Switch to the **Changes** tool window, tab [Local](#).



The status of each file is indicated by the [color](#) in which the path to the file is displayed.

See Also

Procedures:

- [Viewing File Status](#)

Reference:

- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Using TFS Integration

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Using TFS from PhpStorm requires that the following prerequisites should be fulfilled:

1. You have access to a Team Foundation Server.

Note

No client is required, PhpStorm itself acts as the client side.

2. TFS integration is enabled at PhpStorm level through the [bundled TFS Integration plugin](#).
3. TFS is associated with the [entire project](#) or with a [specific directory](#).

When the TFS integration is [enabled](#), the **TFS** menu item is added to the main **VCS** menu, to the context menus of the editor and the **Project** tool window, and the project artifacts are highlighted according to their status. Results of modifications are shown in the [Changes](#) tool window.

This part considers only TFS-specific matters. For procedures that are identical for integration with all supported version control systems, refer to [Common Version Control Procedures](#).

In this part:

- [Creating and Managing TFS Workspaces](#)
- [Checking Out from TFS Repository](#)
- [TFS Check-in Policies](#)

See Also

Procedures:

- [Version Control with PhpStorm](#)

Reference:

- [TFS](#)
- [Checkout from TFS Wizard](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
 - <http://youtrack.jetbrains.net/issues/WI> 
-

Creating and Managing TFS Workspaces

Previous | [Next](#) | [See Also](#) | [Comments](#)

Interaction between your TFS server and the local projects is configured through [workspaces](#) . A workspace mainly maps the folders in the repository with their copies on your machine.

In PhpStorm, you can configure access to several TFS servers and have as many workspaces under them as you need. The list of available TFS server access configurations and workspaces is handled through the [Manage TFS Servers and Workspaces](#) dialog box.

In this part:

- [Opening the Manage TFS Servers and Workspaces dialog box](#)
- [Configuring access to a TFS server](#)
- [Creating a workspace](#)

To open the Manage TFS Servers and Workspaces dialog box

1. [Open the project settings](#), and then click TFS under the **Version Control** node.
2. On the [TFS](#) page, that opens, click the **Manage** button in the **Servers and Workspaces** area. The [Manage TFS Servers and Workspaces](#) dialog box, that opens, shows the list of all available servers and workspaces in them.

To configure access to a TFS server

1. [Open the Manage TFS Servers and Workspaces](#) dialog box with the list of all available servers and workspaces in them.
2. Click the **Add** button in the **Team Servers** area.
3. In the [Add Team Foundation Server](#) dialog box, that opens, specify the URL address of the target server and your credential to access it.
4. Click **OK**. PhpStorm returns to the [Manage TFS Servers and Workspaces](#) dialog box, where the new server is added to the list.

To discard a server access configuration, select the server in the list and click the **Remove** button in the **Team Servers** area.

To create a server workspace

A workspace is identified by its name and the name of its owner, contains the name of your machine, the URL address of the server the workspace belongs to, and a set of mappings between remote and local working folders that are accessible through the workspace.

1. [Open the Manage TFS Servers and Workspaces](#) dialog box, and select the server in question. To refresh the list of the available server workspaces, click the **Reload workspaces** button.
2. Click the **Create** button in the **Workspaces** area.
3. In the [Create Workspace](#) dialog box, that opens, specify the workspace name. Optionally, provide a brief description of the workspace in the **Comment** text box.
4. In the **Working folders** area, define the mappings.
 1. Click the **Add** button . A new line is added to the list of mappings.
 2. In the **Server path** text box, specify the folder on the server you need to work with.
 3. In the **Local path** text box, specify the local folder to store the downloaded data in.
 4. Specify the status of the mapping in the **Status** drop-down list.
 - To enable retrieving data from the server according to the mapping, choose **Active**.
 - To prevent downloading data from the server according to the mapping, choose **Cloaked**.
 5. To discard a mapping, select it in the list and click the **Remove** button .

See Also

Procedures:

- [TFS Check-in Policies](#)
- [Version Control with PhpStorm](#)

Reference:

- [TFS](#)
- [Checkout from TFS Wizard](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Checking Out from TFS Repository

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Checkout helps you obtain a local working copy of the repository folders, which you can edit as required. After making the necessary changes, you can publish results by [committing, or checking in](#) changes to the repository.

Interaction between your TFS server and the local projects is configured through [workspaces](#) that map the folders in the repository with their copies on your machine. You can have remote folders downloaded either according to the mappings from an [existing workspace](#) or define the required mappings during download and have the corresponding workspace generated automatically. In either case, the [Checkout from TFS](#) wizard is used. Upon checkout, a PhpStorm project is created around the downloaded sources.

In this part:

- [Downloading data to a new workspace generated automatically](#)
- [Downloading data to an existing workspace](#)

To have a new workspace generated

1. On the main menu, choose **VCS | Checkout from Version Control | TFS**. The [Checkout from TFS](#) wizard starts.
2. On the [Checkout Mode](#) page, choose the **Create workspace automatically** option, specify the name to identify the workspace, and click **Next**.
3. On the [Source Server](#) page, specify the server to download the data from. Do one of the following:
 - o Choose one of the existing server access configurations.
 - o Click **Add** and [define a new server access configuration](#) in the [Add Team Foundation Server](#) dialog box, that opens.

Click **Next**, when ready.

4. On the [Choose Source and Destination Paths](#) page, define the working folder mapping to generate a workspace around:
 1. In the **Source path** area, select the remote folder to download.
 2. In the **Destination path** text box, specify the location to store the downloaded data in. Type the path manually or click the **Browse** button  and select the folder in the dialog box that opens.

Click **Next**.

5. On the [Summary](#) page, check the details of the workspace to be generated and click **Finish** to launch the checkout procedure. When the checkout is completed, PhpStorm creates a project around the downloaded sources and suggests to open it.

To download the data to an existing workspace

1. On the main menu, choose **VCS | Checkout from Version Control | TFS**. The [Checkout from TFS](#) wizard starts.
2. On the [Checkout Mode](#) page, choose the **Choose workspace manually** option and click **Next**.
3. On the [Source Workspace](#) page, choose the server to download the data from and the workspace of this server to add the working folders to. If necessary, create new [server access configurations](#) and [workspaces](#).

Click **Next**, when ready.

4. On the [Choose Source Path](#) page, specify the remote folder to download.

Note

A folder is available only if the selected workspace contains a mapping either for the folder itself or for its parent. If such mapping is missing, return to the [Source Workspace](#) page and [add the required mapping](#).

Click **Next**, when ready.

5. On the [Summary](#) page, check the details of the workspace to be generated and click **Finish** to launch the checkout procedure. When the checkout is completed, PhpStorm creates a project around the downloaded sources and suggests to open it.

See Also

Procedures:

- [Getting Local Working Copy of the Repository](#)
- [Version Control with PhpStorm](#)
- [Creating and Managing TFS Workspaces](#)

Reference:

- [Checkout from TFS Wizard](#)
- [TFS](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

TFS Check-in Policies

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

A check-in policy is a rule that is executed before every check-in to ensure that the selected changeset is OK to commit. **Standard policies** are stored on the server and are executed on the client machines.

Custom policies are implemented as [custom plugins](#) to PhpStorm. The IDs of these plugins are stored on the server, while the policies themselves are applied locally. Therefore, to enable the

use of a policy in a team, all the team members should install the corresponding plugin.

In this section:

- [Defining the default policy settings](#)
- [Suppressing the default PhpStorm-wide check-in policy settings in a current project](#)
- [Managing the list of available policies](#)
- [Introducing a custom check-in policy](#)

To define the default policy settings to be applied at the PhpStorm level

1. [Open the project settings](#), and then click TFS under the **Version Control** node.
2. On the [TFS](#) page, that opens, select the applicable checkboxes in the **Checkin policies compatibility** area.
 - **Evaluate Team Explorer policies:** select this check box to have the `Microsoft Team Explorer` policy definitions installed and executed on the client machine.
 - **Evaluate Teamprise policies:** select this check box to have the `Teamprise` policy definitions installed and executed on the client machine.
 - **Warn about not installed policies:** select this check box to have warnings displayed in case the specified policy definition is not installed.

To suppress applying the default check-in policy settings to a project

1. [Open the project settings](#), and then click TFS under the **Version Control** node.
2. On the [TFS](#) page, that opens, click the **Manage** button in the **Servers and Workspaces** area.
3. In the [Manage TFS Servers and Workspaces](#) dialog box, that opens, select the project in question from the **Team project** drop-down list.
4. In the **Compatibility** area, select the **Override default settings for team project <project name>** check box.
5. Re-define the default settings by selecting or clearing the corresponding check boxes below.
 - **Evaluate Team Explorer policies:** select this check box to have the `Microsoft Team Explorer` policy definitions installed and executed on the client machine.
 - **Evaluate Teamprise policies:** select this check box to have the `Teamprise` policy definitions installed and executed on the client machine.
 - **Warn about not installed policies:** select this check box to have warnings displayed in case the specified policy definition is not installed.

To manage the list of available policies

The list of available policies consists of standard third-party policies and custom, user-defined policies.

1. [Open the project settings](#), and then click TFS under the **Version Control** node.
2. On the [TFS](#) page, that opens, click the **Manage** button in the **Servers and Workspaces** area.
3. In the [Manage TFS Servers and Workspaces](#) dialog box, that opens, select the required workspace and click the **Checkin Policies** button.
4. In the [Edit Checkin Policies](#) dialog box, that opens, configure the list of policies:
 - To activate a policy, select the **Enabled** check box next to it.
 - To suppress a policy, clear the **Enabled** check box next to it.
 - To discard a policy permanently, select it in the list and click the **Remove** button.

To introduce a custom check-in policy

1. Implement the required policy as a [custom plugin](#).
2. [Download, install](#), and [enable](#) the plugin.

See Also

Procedures:

- [Creating and Managing TFS Workspaces](#)

Reference:

- [Edit Check-in Policies Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Data Sources

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

PhpStorm does not enable you to create databases, but provides facilities to manage and query them. Once you are granted access to a certain database, you can [configure](#) one or more **data sources** within PhpStorm that reflect the structure of the database and store the database access credentials.

Based on this information, PhpStorm establishes connection to the database and provides the possibility to retrieve or change information contained therein.

In this section:

- [Global and Local Data Sources](#)
- [Types of Data Sources](#)
- [Connections](#)
- [Access to Data Sources](#)

Global and local data sources

PhpStorm distinguishes between `Global` and `Local` data sources.

- `Global` data sources are available in all projects in your workspace.
- `Local` data sources are available in one specific project only.

Types of data sources

In PhpStorm you can configure two types of data sources:

- DB data sources to access existing databases of the various types.
- DDL data sources to create data structures based on [DDL](#) files.

DB data sources can be both global and local; DDL data sources can be configured only at the project level.

Connection

PhpStorm establishes a `database connection` between a data source and the corresponding database automatically, the first time it needs to execute a query. A database connection is displayed in the [Data Sources tool window](#) as a node below the data source via which it is established, and is marked with the  icon.

PhpStorm closes a connection automatically upon the session termination; alternatively, you can close a database connection manually (**Close Database Connection** on the context menu of a connection, or **EditorDelete** on the toolbar of the Data Sources tool window).

Access to data sources

Access to data sources is supported through a variety of PhpStorm features and components:

- [Support for SQL](#), which includes [syntax and error highlighting](#), [code completion](#), selection of [words and statements](#), parameter lookup, and [refactoring](#) (**Introduce constant**, for instance).
- The possibility to use source files with [injected SQL code](#), with different [SQL dialects](#), configurable at the project, directory, or file level.
- Dedicated [Data Sources tool window](#), where you can configure data sources.
- [Database](#) console.

In this part:

- [Accessing the Data Sources Tool Window](#)
- [Creating and Importing Data Sources](#)
- [Configuring a DB Data Source](#)
- [Configuring a DDL Data Source](#)
- [Changing Properties of a Data Source](#)
- [Generating a Data Structure Definition \(DDL\) File](#)
- [Adding Data Structure Definition Files to a DDL Data Source](#)
- [Accessing Data Sources Via the Database Console](#)
- [Viewing Table Data from the Data Sources Tool Window](#)
- [Manipulating Table Data in the Table Editor](#)
- [Running Injected SQL Statements from the Editor](#)
- [Comparing Data Sources](#)

See Also

Reference:

- [Table Editor](#)
- [Data Sources Tool Window](#)
- [Database Console Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Accessing the Data Sources Tool Window

Previous | [Next](#) | [See Also](#) | [Comments](#)

To open the **Data Sources** tool window, do one of the following

- On the main menu, choose **Tools | Data Sources**.
- On the main menu, choose **View | Tool Windows | Data Sources**.

See Also

Reference:

- [Data Sources Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Creating and Importing Data Sources

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts:  

You can create a data source of the required `type` from scratch or by importing the configuration files that contain the necessary data definitions.

In this section:

- [Importing a data source](#)

- [Creating a data source](#)

To import a data source

1. [Open the Data Sources tool window.](#)
2. Do one of the following:
 - Click  on the toolbar.
 - Press `Alt+InsertCommand N`.
 - Select  **Add Data Source** in the context menu.

To create a data source

1. [Open the Data Sources tool window.](#)
2. To start creating a data source, do one of the following:
 - Click  on the toolbar.
 - Press `Alt+InsertCommand N`.
 - Select  **Add Data Source** in the context menu.
3. Select the [type](#) of the data source or the import option:
 - To create a new DB data source, select  **DB Data Source**. [Configure the new data source](#) in the [DB Data Source Properties dialog](#).
 - To create a new DDL data source, select  **DDL Data Source**. [Configure the new data source](#) in the [DDL Data Source Properties dialog](#).

Note

You will need a file that contains the corresponding [DDL](#)  statements.

See Also

Concepts:

- [Library](#)
- [Data Sources](#)

Procedures:

- [Manipulating the Tool Windows](#)
- [Configuring Third-Party Tools](#)

Reference:

- [Data Sources Tool Window](#)
- [DB Data Source Properties](#)
- [DDL Data Source Properties](#)
- [File Encodings](#)
- [External Tools](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

2.0+

Configuring a DB Data Source

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Configuring a involves the following general activities:

- [Specifying general DB data source settings](#)
- [Appointing a JDBC driver](#)
- [Defining access to the target database](#)
- [Downloading tables](#)
- [Gaining access to specific tables only](#)
- [Configuring behavior of the Database Console](#)
- [To view and edit advanced connection properties](#)

To specify general DB data source settings

1. Open the [Data Source Properties](#) dialog box by doing one of the following:
 - [Start creating](#) a new DB data source.
 - In the [Data Sources](#) tool window, select the desired data source and choose **Data Source Properties** in the context menu.
2. Specify the name of the data source.
3. Specify the scope in which the data source will be available in the **Data Source Level** drop-down list.
 - To enable access to the data source only from the current project, select **Project**.
 - To enable access to the data source from any project in your workspace, select **Global**.

To appoint a JDBC driver to use

1. In the **JDBC Driver Files** drop-down list on the **Database** tab of the [Data Source Properties](#) dialog box, specify the library or archive with the class that implements the [JDBC driver](#)  of the required type. Do one of the following:

- If you already have a relevant library or archive downloaded, click the **Browse** button  and select the library or archive in the **Choose JDBC Driver Files** dialog box that opens.
- To have PhpStorm download the JDBC driver sources, select the required JDBC driver from the drop-down list. The read-only list below displays the relevant archive detected among the specified sources. Click the link to have PhpStorm start downloading the detected archive.

Note

PhpStorm analyzes the archives and libraries and detects classes that implement drivers. Detected classes are available in the **JDBC Driver Class** drop-down list.

2. From the **JDBC Driver Class** drop-down list, select the relevant driver implementation class.

To specify the settings to access the target database

1. In the **Database URL** text box on the **Database** tab of the [Data Source Properties](#) dialog box, type the URL address of the database.
2. In the **Username** and **Password** text boxes, type your database access credentials.
3. To check that the specified settings ensure successful connection to the database, click the **Test Connection** button.

Warning

The database server should be running.

To download tables

- Click the **Refresh Tables** button .

Tip

If you do not need to access the entire database, specify the set of [accessible tables](#).

To get access to specific tables

1. Open the **Schemas & Tables** tab of the [Data Source Properties](#) dialog box.
2. Do one of the following:
 - If the target database supports [schemas](#) , select the **Scan for Tables** check box next to the relevant schemas.

Tip

To have PhpStorm resolve the names of the tables accessed through a schema so you do not need to write their fully qualified names, select the **Make Default** check box next to the relevant schema.

- Specify the pattern of table names to retrieve the matching tables only.

To configure the behaviour of the Database Console

1. Open the **Console** tab of the [Data Source Properties](#) dialog box.
2. Specify the **SQL dialect** to be used by default. Based on this setting, PhpStorm provides you with code assistance in the [Database Console](#) tool window.
3. Specify the [run configuration](#) to launch the **Database Console** tool window.

To view and edit advanced connection properties

- In the **Advanced** tab of the [Data Source Properties](#) dialog box, update the contents of the **Value** text box next to the names of the relevant connection properties.

See Also

Concepts:

- [Data Sources](#)

Procedures:

- [Data Sources](#)

Reference:

- [Data Sources Tool Window](#)
- [DB Data Source Properties](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Configuring a DDL Data Source

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac 

DDL data sources allow you to model structures of databases that do not yet exist or that you do not have access to. A DDL data source is based on a [data structure definition](#)  specified in a `.ddl` file. PhpStorm reads [DDL](#)  statements from this file and creates the corresponding structure in the new DDL data source.

DDL data sources can be used for various purposes. For example, if your target is to prepare a data source structure definition file to use in the future, you may need to check that it determines the correct data structure. You can also check whether the structure generated through your definition file is consistent with the target database. To do so, specify the JDBC data source that provides access to the target database as the [parent data source](#).

PhpStorm provides various coding assistance for handling data structure definition files. You can [navigate](#) from statements in a .ddl file to the corresponding tables or columns in the **Data Sources** tool window. Any changes made to the tables in the data source are reflected in the .ddl file.

You can create a data structure definition file manually, or copy it from another location, or [generate](#) it based on an entire configured data source or any table or column accessible through it.

Note

DDL data sources only exist on the project level.

To configure a DDL data source from a data structure definition

1. [Start creating](#) a new DDL data source.
2. In the [DDL Data Source Properties](#) dialog box, type the name of the new data source.
3. Click the **Add DDL File** button  and specify the desired data structure definition file.

Warning

PhpStorm parses the specified file according to the SQL dialect assigned to it or its parent. To have the data structure created successfully, this dialect should support DDL statements. SQL dialect settings are specified in the [SQL Dialects](#) page of the [Settings](#) dialog box.

4. Select the parent data source from the list of already configured data sources.

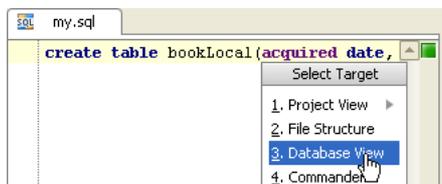
Note

You can leave these properties undefined, and [change](#) them later, adding SQL files, or specifying parent data sources.

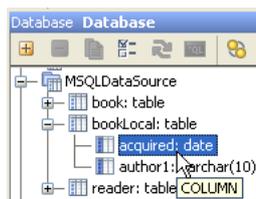
5. Click **OK**.

To navigate from a statements in a DDL file to the corresponding table or column

1. Open the desired .ddl file in the editor.
2. Position the cursor at the name of the desired table or column and press **Alt+F1Alt F1**.



The corresponding column is highlighted in the **Data Sources** tool window:



See Also

Concepts:

- [Data Sources](#)
- [Supported Languages](#)

Procedures:

- [Data Sources](#)
- [Adding Data Structure Definition Files to a DDL Data Source](#)

Reference:

- [Data Sources Tool Window](#)
- [DDL Data Source Properties](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Changing Properties of a Data Source

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

To change the properties of a data source

- In the [Data Sources](#) tool window, select the desired data source and do one of the following:
 - Choose **Local Data Source Properties** on the context menu.
 - Click the **Local data Source Properties** toolbar button .
- In the dialog box that opens, change the settings as necessary. Proceed as during configuration of a [JDBC](#) or [DDL](#) data source, depending on the current data source type.

See Also

Concepts:

- [Data Sources](#)

Procedures:

- [Data Sources](#)
- [Configuring a DB Data Source](#)
- [Configuring a DDL Data Source](#)

Reference:

- [Data Sources Tool Window](#)
- [DB Data Source Properties](#)
- [DDL Data Source Properties](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Generating a Data Structure Definition (DDL) File

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac 

To generate a data structure definition (DDL) file

- In the [Data Sources](#) tool window, select the desired data source, table, or column, and do one of the following:
 - Choose **Generate and Copy DDL** on the context menu.
 - Press the `Ctrl+Shift+Command Shift C` keyboard shortcut.
 - Click the **Generate and Copy DDL** toolbar button .

As a result, PhpStorm will place the generated statements onto the clipboard.

- In the [Project](#) tool window, [create a file](#) with the desired name and the extension `.ddl`.
- Open new file in the editor and paste the generated statements from the clipboard.

Warning

PhpStorm parses the specified file according to the SQL dialect assigned to it or its parent. To have the data structure created successfully, this dialect should support DDL statements. SQL dialect settings are specified in the [SQL Dialects](#) page of the [Settings](#) dialog box.

3.0+ |

Note that you can change the SQL dialect for any SQL or DDL file open in the editor by selecting the corresponding context menu command.

To change the SQL dialect for an SQL or DDL file

- [Open the file of interest in the editor](#).
- Right-click somewhere in the editor to open the context menu, select **Change <current option> SQL dialect to**, and then select **more**.
- In the [SQL Dialects dialog](#) that opens, click the corresponding **SQL Dialect** cell, select the desired dialect from the list, and click **OK**.

See Also

Concepts:

- [Data Sources](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Adding Data Structure Definition Files to a DDL Data Source

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

To add a DDL file to a DDL data source, do one of the following:

- In the [Data Sources](#) tool window, select the desired DDL data source and click **Add** tool bar button . In the [Data Source Properties](#) dialog box that opens, click the **Add** button  and specify the desired SQL file that contains DDL statements.

Warning

PhpStorm parses the specified file according to the SQL dialect assigned to it or its parent. To have the data structure created successfully, this dialect should support DDL statements. SQL dialect settings are specified in the [SQL Dialects](#) page of the [Settings](#) dialog box.

- In the [Project](#) tool window, select one or more SQL files and drag them to the target DDL data source.

Note

- The target data source must of the DDL type; for JDBC data sources, this operation is not allowed.
- If the selection includes some non-SQL files, these files are ignored.

See Also

Concepts:

- [Supported Languages](#)

Procedures:

- [Configuring a DDL Data Source](#)

Reference:

- [Data Sources Tool Window](#)
- [DDL Data Source Properties](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

2.0+

Accessing Data Sources Via the Database Console

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

With PhpStorm, you can query and modify configured data sources from the dedicated [Database Console](#) tool window using one of the supported languages:

- Derby
- H2
- HSQLDB
- MySQL
- Oracle
- PostgreSQL
- PostgresPLSQL
- SQL Server
- SQL92
- SQLite
- Sybase

You can either enter all the necessary commands in the **Input** pane of the console or prepare a file that contains commands in the desired language. In both cases, PhpStorm provides code specific completion, syntax and error highlighting, and completion of the data source, table and column names.

PhpStorm also supports operations with database [binary large objects \(BLOBs\)](#).

Tip

The **Input** pane opens as a new editor tab as soon as the **Database** console is invoked.

Retrieved records are shown in the **Result** pane, where you can view, analyze, and update them.

PhpStorm opens connections to databases when necessary without any steps from your side and displays them below corresponding data source nodes in the **Data Sources** tool window.

Tip

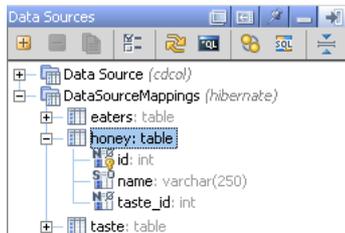
The SQL server should be running. You can create a special command using the external tools configuration facility, as described in the section [Using Third-Party Tools](#).

In this section:

- [Writing and Executing SQL Commands in the Database Console](#)
- [Viewing Query Results](#)

To launch the database console

1. Open the [Data Sources](#) tool window.
2. Select the desired data source.



3. Click the **Run Database Console** toolbar button  or press `Ctrl+Shift+F10` / `Command Shift F10`. The **Database console tool window** for accessing the selected data source opens.
 - o The **Result** and **Output** panes open as usual, in the bottom area of the screen.
 - o The **Input** pane opens as a separate editor tab.

See Also

Concepts:

- [Data Sources](#)

Procedures:

- [Data Sources](#)
- [Manipulating Table Data in the Table Editor](#)
- [Running Injected SQL Statements from the Editor](#)

Reference:

- [Database Console Tool Window](#)
- [Database Console Properties Dialog](#)
- [Data Sources Tool Window](#)
- [Table Editor](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Writing and Executing SQL Commands in the Database Console

Previous | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts:  

Prior to running SQL commands, you may want to check and, if necessary, [change the SQL dialect](#) to be used.

When typing the SQL commands in the **Input** pane of the [Database Console tool window](#) (this pane opens as a new editor tab), use various kinds of coding assistance, such as code-specific completion, syntax and error highlighting, completion of the data source, table and column names. You may also be interested in using the following features:

- [Viewing parameter information](#).
- [Introducing parameters](#).
- [Navigating to referenced tables](#).

[Run the command](#) when ready.

As you run the SQL statements, PhpStorm memorizes them. So, at a later time, you can view the statements you have already run and, if necessary, run them again.

There are two ways of working with the history of executed SQL statements. You can:

- [Browse the statements one by one](#), right in the **Input** pane of the Database Console, moving to the previous or next statement (`Ctrl+Up` / `Command Up` or `Ctrl+Down` / `Command Down`).
- [View the executed statements all at once](#), in a dedicated history dialog (`Ctrl+E` / `Command E`). Using the history dialog, you can remove unnecessary statements from the history. You can also copy selected statements to the Console, for example, to run them again.

Note

Note that:

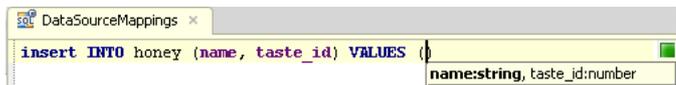
- There is a separate history of executed statements for each of your data sources.
- The number of executed SQL statements to be memorized may be changed by editing the value of **Console commands history size** in the [Editor](#) page of the IDE Settings.
- The history files that haven't been used for two weeks are automatically deleted. That is, if you haven't run SQL statements for a certain data source within this time, the corresponding history is discarded.

Changing the SQL dialect

1. [Open the Database Console](#) for the data source of interest.
2. Click the **Properties** button  on the toolbar.
3. In the [Database Console Properties dialog](#) that opens, select the [General](#) tab.
4. Select the desired dialect from the **SQL Dialect** list and click **OK**.

Viewing parameter information

When typing an SQL statement, you can view [parameter information](#) on-the-fly or by pressing `Ctrl+P` / `Command P`. PhpStorm informs you about the type of required parameters in a tooltip.



Using parameters in commands

You can use parameters in SQL statements and specify which values should be used in place of these parameters when running the statements:

1. When typing a command, enclose the parameters in #, \$, or ? characters.
2. In the Parameters pane, specify the values to substitute for the corresponding parameters at command execution.



Navigating from a reference in the query to the corresponding table

Do one of the following:

- Place the caret at the desired name and press **Ctrl+BCommand B**. The corresponding table and column are highlighted in the data source for which the console was launched.
- With the **CtrlCtrl** key pressed, hover the mouse pointer over the reference to a table or column. So doing, the corresponding table or column declaration is displayed in a tooltip, and the reference turns to a hyperlink. Click the link to jump to the declaration in the data source.



Running SQL commands

You can run the SQL statements one at a time, or execute a number of statements that follow one another in the console at once:

- To execute a command at the caret, press **Ctrl+EnterCommand Enter** or click on the toolbar.
- To run several consecutive commands, select them in the console. Then, press **Ctrl+EnterCommand Enter** or click on the toolbar.

Tip

You can prepare a script in an SQL file, copy and then paste its contents into the **Input** pane.

Browsing the executed SQL statements in the input pane

When working with the **Input** pane of the **Database Console** tool window, you can browse the executed SQL statements one by one by moving to the previous or next statement:

- To navigate to the previous statement, press **Ctrl+UpCommand Up**.
- To navigate to the next statement, press **Ctrl+DownCommand Down**.

Working with the executed SQL statements in the history dialog

When working with the **Input** pane of the **Database Console** tool window, you can open the history dialog to see the executed SQL statements all at once.

To open this dialog:

- Press **Ctrl+ECommand E**.

In this dialog, you can:

- Navigate through the list of the executed statements using the **UpUp** and **DownDown** arrow keys.
- Remove the statements from the history. To delete a statement, select (navigate to) the statement to be deleted and press **DeleteDelete**.
- Copy the statements to the **Input** pane. To copy a statement, do one of the following:
 - Double-click the statement to be copied.
 - Select (navigate to) the statement of interest and press **EnterEnter**.
 - Select the statement and click **OK**.

As a result, the history dialog closes and the selected statement is copied to the **Input** pane.

See Also

Concepts:

- [Data Sources](#)

Procedures:

- [Data Sources](#)
- [Manipulating Table Data in the Table Editor](#)
- [Running Injected SQL Statements from the Editor](#)

Reference:

- [Database Console Tool Window](#)
- [Data Sources Tool Window](#)
- [Table Editor](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Viewing Query Results

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

The results of executing an SQL command are reflected in the **Output** and **Result** panes. The **Output** pane displays information and error messages on the command execution. If the command supposes data retrieval, a new tab opens in the **Result** pane showing the retrieved records. For each command PhpStorm opens a separate tab.

Besides viewing and analyzing, you can [update the retrieved records](#) as necessary right in the corresponding **Results** tab, just like in the [Table Editor](#).

To analyze the results of running SQL commands, perform these general steps

- View the results of query execution in the **Result** tab. Results of each query are displayed in a separate tab.



- To navigate between result sets, use the **First Page**, **Last Page**, **Previous Page**, and **Next Page** toolbar buttons.
- To refresh the current page after editing or resorting the table data, click the **Reload Page** toolbar button.
- To copy the values from a result, click the **Copy** button on the toolbar of the corresponding tab or press the **Ctrl+C** Command C or **Ctrl+Insert** Command Insert shortcut.
- To copy the query that produced a particular result, click the **Copy Query** button on the tool bar of the corresponding tab or press the **Ctrl+Alt+Shift+C** Command Alt Shift C shortcut.
- View query execution details and error messages in the **Output** tab.

To update the queried table, perform these general steps

- To update a record, select the corresponding row and modify the values as necessary.
- To add a record, click the **Add New Row** toolbar button and fill in the fields.
- To remove one or several records, select the corresponding rows and click the **Delete Selected Rows** toolbar button.

See Also

Concepts:

- [Data Sources](#)

Procedures:

- [Data Sources](#)
- [Manipulating Table Data in the Table Editor](#)
- [Running Injected SQL Statements from the Editor](#)

Reference:

- [Database Console Tool Window](#)
- [Table Editor](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Viewing Table Data from the Data Sources Tool Window

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

You can access tables immediately from the **Data Sources** tool window, without executing queries explicitly.

To view the structure of a table

1. In the [Data Sources](#) tool window, expand the desired [data source](#) node.
2. Select the desired table and press **Ctrl+Q** Command J. PhpStorm opens a quick documentation look-up window, which displays the following information:
 - The `CREATE TABLE` query through which the table has been created.
 - Results of executing the `SELECT * FROM table_name` query (the first 10 rows of the table).

See Also

Concepts:

- [Data Sources](#)

Procedures:

- [Accessing Data Sources Via the Database Console](#)

Reference:

- [Data Sources Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

2.0+

Manipulating Table Data in the Table Editor

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: 

You can add, remove and modify table records by running SQL statements [in the Database console](#) or [in the editor](#). You can also edit the records retrieved from tables [in the Results pane](#).

Besides these facilities, PhpStorm also features the `Table Editor` which provides a graphical interface for manipulating table data.

Note

The Table Editor is not available for the data sources of the [DDL type](#).

- [Opening a table in the Table Editor](#)
- [Navigating through the ranges of table rows](#)
- [Showing and hiding table columns](#)
- [Sorting and filtering data](#)
- [Editing fields and uploading files](#)
- [Adding a record to a table](#)
- [Deleting records](#)
- [Setting the number of rows to display simultaneously](#)
- [Specifying data export format](#)
- [Copying table data to the clipboard](#)
- [Copying queries to the clipboard](#)
- [Saving large objects \(LOBs\) in files](#)
- [Refreshing a table view](#)

Opening a table in the table editor

1. In the [Data Sources tool window](#), select the table of interest.
2. Do one of the following:
 - Press `F4F4`.
 - Click  **Open Table Editor** on the toolbar of the `Data Sources` window.
 - Select **Open Table Editor** from the context menu.

As a result, the table opens in the [Table Editor](#) on a separate editor tab.

Navigating through the ranges of table rows

For tables containing many records, normally, not all but only a certain number of rows is shown at a time. This fixed number of rows displayed simultaneously is referred to as a page.

To navigate through the pages, use the following toolbar buttons, context menu commands, or keyboard shortcuts:

-  **First Page**. Go to the first of the pages.
-  **Previous Page** (`Ctrl+Alt+UpCommand Alt Up`). Go to the previous page.
-  **Next Page** (`Ctrl+Alt+DownCommand Alt Down`). Go to the next page.
-  **Last Page**. Go to the last of the pages.

Showing and hiding table columns

Do one of the following:

- Right-click the header row, and then click the name of the column which you want to hide or show.
- Click  **Filters and Ordering** on the toolbar of the Table Editor, or press `Ctrl+F12Command F12`.

In the [Filters and Ordering dialog](#), use the check boxes in the first (leftmost) column to control visibility of the table columns.

Sorting and filtering data

1. Do one of the following:
 - Press `Ctrl+F12Command F12`.
 - Click  **Filters and Ordering** on the toolbar of the Table Editor.
 - Select **Filters and Ordering** from the context menu.

The [Filters and Ordering dialog](#) that opens shows the list of table fields.

	Name	Order	Filter
<input type="checkbox"/>	id	-	
<input type="checkbox"/>	name	-	
<input type="checkbox"/>	taste_id	-	

- Use the check boxes in the first (leftmost) column to specify which fields should be shown or hidden. (The fields next to the check boxes that are not selected will be hidden.)
- You can have the records sorted by one or more visible fields. Use the lists in the **Order** column to define how the data should be ordered.

The hyphens (-) mean that no ordering rules are specified for the corresponding fields.

asc stands for ascending, desc - for descending.

The number in front of asc or desc means the ordering priority.

	Name	Order	Filter
<input type="checkbox"/>	id	1 desc	<30
<input checked="" type="checkbox"/>	name	-	
<input checked="" type="checkbox"/>	taste_id	1 asc	>2

- Use the **Filter** column to specify conditions for filtering the records. You can specify filtering conditions for several fields.

To define a filtering condition for a field, double-click the **Filter** cell next to the field name, type the condition, and then click a different cell or press **EnterEnter**.

	Name	Order	Filter
<input checked="" type="checkbox"/>	id	-	<10
<input type="checkbox"/>	name	-	
<input checked="" type="checkbox"/>	taste_id	-	>3

- Click **OK** in the **Filters and Ordering** dialog to apply the specified settings to the table.

Editing fields and uploading files

- To start editing a field, double-click the corresponding table cell.
- Edit the cell contents. To do that, you have the following options:
 - Modify the data right in the cell. To save the changes, press **EnterEnter**.
 - Click or press **Shift+EnterShift Enter** to open the **Enter text data or choose file** dialog. In this dialog you can:
 - Make the necessary edits in the upper area.
 - Upload a file into the field. To do that, click **Browse** and select the desired file in the **Choose Path** dialog.
 - Insert `null` into the field (click **Null**).
- To leave the editing mode, click a different cell or press **EscapeEscape**.

Adding a record to a table

- Do one of the following:
 - Press **Alt+InsertCommand N**.
 - Click **Add New Row** on the toolbar of the Table Editor.
 - Select **Add New Row** from the context menu.
- Fill in the fields. Use the **EnterEnter** or the **TabTab** key to indicate the end of your input in the current cell and move on to the next cell. The new record is saved when you press **EnterEnter** or **TabTab** in the last of the cells.

Deleting records

- Select the row or rows that you want to delete. To do that:
 - If you are going to delete one row, select any of the cells within the corresponding row.
 - If you are going to delete a range of rows, select one or more cells in each of the corresponding rows.

Note

You can select only the rows that follow one another.

- Do one of the following:
 - Press **Ctrl+YCommand Y**.
 - Click **Delete Selected Rows** on the toolbar of the Table Editor.
 - Select **Delete Selected Rows** from the context menu.
- Confirm your intention to delete the selected row or rows by clicking **OK** in the corresponding dialog.

Setting the number of rows to display simultaneously

- Do one of the following:
 - Click **Properties** on the toolbar of the Table Editor.
 - Select **Properties** from the context menu.
- In the **Table Editor Properties dialog**, select the **General** tab.
- In the **Page Size** field, specify the number of rows to display simultaneously. For all table rows to be shown at the same time, type zero.
- Click **OK**.

5. Optionally, if you want to apply the new page size setting to the current table view, do one of the following:
 - Press `Ctrl+R` Command `R`.
 - Click  **Reload Page** on the toolbar of the Table Editor.
 - Select **Reload Page** from the context menu.

Specifying data export format

The data export format is the format of data copied from a table to the clipboard. To define this format:

1. Do one of the following:
 - Click  **Properties** on the toolbar of the Table Editor.
 - Select **Properties** from the context menu.
2. In the [Table Editor Properties dialog](#), select the **Data Export** tab.
3. In the **String Quotation** fields, specify the characters that should be used to enclose the exported string values.

Note

The characters preceding and following the string values may be different. That's why two separate fields are provided.

4. In the **Values Separator** field, specify the character or characters to be used to separate exported values.
5. If you want to include the table header (column names) in exported data, select the **Include Table Header** check box.
6. If you want to include the row numbers in exported data, select the **Include Row Number** check box.
7. Click **OK**.

Copying table data to the clipboard

You can copy the data contained in selected table fragments to the clipboard. To do that:

1. Select the table fragment of interest. This may be a cell, a row, a range of cells or rows, or all rows currently shown.
2. Do one of the following:
 - Press `Ctrl+C` Command `C` or `Ctrl+Insert` Command `Insert`.
 - Click  **Copy** on the toolbar of the Table Editor.
 - Select **Copy** from the context menu.

As a result, the selected data are copied to the clipboard. The data format corresponds to the [data export format](#) currently specified.

Copying queries to the clipboard

You can copy the SQL statement that resulted in generating the current table view to the clipboard.

Do one of the following:

- Press `Ctrl+Alt+Shift+C` Command `Alt Shift C`.
- Click  **Copy Query** on the toolbar of the Table Editor.
- Select **Copy Query** from the context menu.

Saving large objects (lobs) in files

You can save large objects ([LOBs](#) ) stored in database tables in local files. To do that:

1. Select the cell containing the large object of interest.
2. Do one of the following:
 - Click  **Save LOB As** on the toolbar of the Table Editor.
 - Select **Save LOB As** from the context menu.
3. In the **Select File to Save** dialog:
 1. Specify the destination directory in the upper part.
 2. Type the name of the file **File name** field.
 3. Click **OK**.

Refreshing a table view

You may want to refresh the view of a table in the Table Editor in order to:

- Synchronize the data shown with the actual contents of the table in the database.
- Apply the changed [page size setting](#) to the current table view.

In all such cases, use the **Reload Page** command:

- Press `Ctrl+R` Command `R`.
- Click  **Reload Page** on the toolbar of the Table Editor.
- Select **Reload Page** from the context menu.

See Also

Concepts:

- [Data Sources](#)

Procedures:

- [Accessing Data Sources Via the Database Console](#)
- [Data Sources](#)

Reference:

- [Table Editor](#)
- [Filters and Ordering Dialog](#)
- [Table Editor Properties Dialog](#)
- [Data Sources Tool Window](#)
- [Database Console Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Running Injected SQL Statements from the Editor

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

You can [inject](#) SQL statements in your source code and invoke their execution by means of an [intention action](#). If an SQL [injection contains parameters](#), PhpStorm detects them when the statement execution is started and prompts you to specify the substitution values.

To run an injected SQL statement

1. Prepare the desired SQL statement and inject it in the source code.
2. Position the cursor at the desired SQL injection, press `Alt+Enter` `Alt Enter`, and choose **Run Query in Console** on the context menu.
3. Specify the data source to run the statement against. Do one of the following:
 - If no instances of the [Database Console](#) tool window are running, select the target data source in the **Choose Data Source** pop-up window.
 - If one or more instances of the **Database Console** are already running, the **Choose Console** pop-up window opens. In this pop-up window, specify whether you want to run the statement in one of them or open a new instance.
 - To have the statement executed in an already running console, choose the console with the name of the target data source from the **Running** list.
 - To have a new instance of console opened, choose **Open New**, then choose the target data source.
4. View and analyze the query execution result in the **Result** and **Output** panes.

To run an injected SQL statement with parameters

1. Prepare the desired SQL statement with parameters enclosed in #, \$, or ? characters. Inject the statement in the source code.
2. Position the cursor at the desired SQL injection and press `Alt+Enter` `Alt Enter` and choose **Run Query in Console** on the context menu.
3. [Choose the target data source](#).
4. In the **Please Provide Missing Values** dialog box that opens, specify the values to substitute for the parameters and click **OK**.
5. View and analyze the query execution result in the **Result** and **Output** panes.

See Also

Concepts:

- [Data Sources](#)

Procedures:

- [Data Sources](#)
- [Viewing Method Parameter Information](#)

Reference:

- [Database Console Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Comparing Data Sources

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

To compare data sources

1. In the [Data Sources](#) tool window, select two data sources, schemas, or tables to be compared.
2. On the toolbar of the **Data Sources** tool window, click the **Compare** toolbar button  or press `Ctrl+D` `Command D`.
3. Explore the differences in the [Differences viewer](#).

Tip

You can also open the difference viewer without running PhpStorm. This is done through the following command:

```
<path to PhpStorm executable file> diff <path_1> <path_2>
```

where path_1 and path_2 are paths to the database objects in question.

See Also

Procedures:

- [Comparing Folders](#)

Language and Framework-Specific Guidelines:

- [Data Sources](#)

Reference:

- [Data Sources Tool Window](#)
- [Differences Viewer for Folders and DB Objects](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Managing Plugins

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

In this section:

- [Configuring Plugins](#)
- [Managing Enterprise Repositories](#)
- [Updating, Installing and Uninstalling Plugins from a Repository](#)

See Also

Concepts:

- [Plugins](#)

Procedures:

- [Accessing the IDE Settings](#)

Reference:

- [Plugins](#)
- [Updates](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Configuring Plugins

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm distinguishes between the following three types of plugins:

- Plugins bundled with PhpStorm. These plugins are installed and enabled by default.
- Plugins available from the [JetBrains Plugin Repository](#). These plugins need installation and enabling.
- Custom plugins available from enterprise repositories. To use these plugins, you need to configure access to this repository from PhpStorm. After that its plugins are available for downloading, installing, and updating exactly same way as plugins from the JetBrains Plugins Repository.

The set of installed and enabled plugins determines the set of supported features, integrations, application servers, and technologies. When a plugin is no longer in use, you can either uninstall or disable it.

In this section:

- [Configuring plugins from JetBrains Plugins Repository](#)
- [Configuring plugins from enterprise repositories](#)
- [Enabling and disabling plugins](#)

To configure a plugin from the default jetbrains plugins repository

- Download, install, enable, update, and remove the plugin using the [Plugin Manager](#).

To configure a plugin from an enterprise repository

1. [Specify the URL address of the repository](#) to enable access to it from PhpStorm.
2. Download, install, enable, update, and remove the plugin using the [Plugin Manager](#).

To enable or disable a plugin

1. [Open the PhpStorm settings](#), and then click the **Plugins** node.
2. In the [Plugins](#) page, that opens, select the required plugin. Use the **Search** field and the **Show** drop-down list to make the search easier.
3. Do one of the following:
 - o To enable the plugin, select the **Enable** check box next to it.
 - o To disable the plugin, clear the **Enable** check box next to it.
4. Restart PhpStorm for the changes to take effect.

See Also

Concepts:

- [Plugins](#)

Procedures:

- [Managing Plugins](#)

Reference:

- [Plugin Configuration Wizard](#)
- [Plugins](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Managing Enterprise Repositories

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

By default, PhpStorm provides you with access to the [JetBrains Plugin Repository](#). Besides that, you can download and install plugins from other repositories of your choice. To use plugins from existing enterprise repositories from PhpStorm, you need to create a list of URL addresses to access such repositories through.

To manage the list of available enterprise repositories

1. [Open the PhpStorm settings](#), and then click the **Plugins** node.
2. In the [Plugins](#) page, that opens, click the **Browse repositories** button.

Tip

If you cannot see the **Browse repositories** and **Install plugin from disk**, use the slider that appears to the right of the **Plugin information pane**.

3. In the [Browse Repositories](#) dialog box, that opens, click the **Manage repositories** button.
4. In the [Custom Plugin Repositories](#) dialog box, that opens, configure a list of enterprise repositories, that you want to be available from PhpStorm.
 - o To add a new repository, click the **Add** button and specify the URL address of the required repository in the **Add New Plugin Host** dialog box, that opens. To make sure that the specified URL address guarantees successful connection to the repository, click the **Check Now** button. Upon clicking **OK** you return to the **Custom Plugin Repositories** dialog box, where the new URL address is added to the list.

Note

If the whole team uses customized PhpStorm, you can add the whole list of plugin hosts to `bin/idea.properties` under the PhpStorm installation folder. Just open this file and add the following line:

```
-D idea.plugin.hosts=[URL1],[URL2],...[URLn]
```

where `[URL1],[URL2],...[URLn]` is a comma-delimited list of URLs of the enterprise repositories.

- o To update the URL address of a repository, select it in the list and click the **Edit** button. In the **Edit Plugin Host** dialog box, update the URL address, as necessary.
- o To discard a repository, select its URL address in the list and click the **Remove** button.

See Also

Concepts:

- [Plugins](#)

Reference:

- [Custom Plugin Repositories](#)
- [Browse Repositories Dialog](#)
- [Plugins](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Updating, Installing and Uninstalling Plugins from a Repository

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Plugins from all repositories - either from the default JetBrains Repository or from your own enterprise ones - are downloaded, installed, updated, and removed through the PhpStorm Plugin Manager. This functionality is provided in the [Plugins](#) page of the [Settings](#) dialog box.

In this section:

- [Installing plugins from a repository](#)
- [Updating a plugin from a repository to a newer version](#)
- [Removing a plugin](#)

Tip

Refer to the [Legend of Plugins Status](#) for details.

To install a plugin from a repository

1. [Open the PhpStorm settings](#), and then click the **Plugins** node.
2. In the [Plugins](#) page, that opens, click the **Browse repositories** button.

Tip

If you cannot see the **Browse repositories** and **Install plugin from disk**, use the slider that appears to the right of the **Plugin information** pane.

3. In the [Browse Repositories](#) dialog box, that opens, select the required plugin. Use the controls of the dialog box to make the search easier:
 - From the **Repository** drop-down list, choose the repository the plugin reside in.
 - Use the [Category](#) drop-down list to limit the search area.
 - Use the [Sort by](#) drop-down list to have the available plugins displayed sorted by various criteria.
 - Use the [Search](#) field to search for the required plugin by name.
4. On the context menu, choose **Download and Install** command, or click the **Download and Install** button  on the toolbar of the dialog box.
5. Confirm your decision in the dialog box.
6. Restart PhpStorm for the changes to take effect.

To update a plugin from a repository to a newer version

1. [Open the PhpStorm settings](#), and then click the **Plugins** node.
2. Select the desired plugin from the list. Plugins that have newer versions are [highlighted blue](#).
3. On the context menu, choose **Update plugin** command, or click the **Update Plugin** button  on the toolbar of the dialog box.

Note

If a plugin with the same name is available both as a bundled plugin and as a user plugin under `config/plugins`, PhpStorm loads the plugin which has a later version.

4. Restart PhpStorm for the changes to take effect.

To remove a plugin

1. [Open the PhpStorm settings](#), and then click the **Plugins** node.
2. Select the plugin to be removed from the list.
3. On the context menu, choose **Uninstall**, or click the **Uninstall** button .

Warning

Plugins bundled with PhpStorm cannot be removed.

4. Confirm removal in the dialog box.
5. Restart PhpStorm for the changes to take effect.

See Also

Concepts:

- [Plugins](#)

Procedures:

- [Plugin Development Guidelines](#)

Reference:

- [Plugins](#)
- [Updates](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Managing Tasks and Context

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm provides facilities to set up your workflow according to the issue tracking procedure accepted in your team. You can bind your account in an issue tracker to your project, and work on it in the discourse of `tasks` and `contexts`.

Task

A `task` is an activity performed in PhpStorm and is identified by a task name. Normally, a task correlates with an issue in your issue tracking system. This correlation is set by using the desired issue ID as the task name. When you switch between tasks, PhpStorm cleans your workspace, create a changelist for the task, and load a stack trace, if any. Alternatively, you can define a task yourself so it reflects an activity that is not registered in your issue tracker.

Context

A `context` is a set of files opened in the editor while working on a task or independently from it. You can switch between contexts by switching between tasks associated with them. Alternatively, you can save and clear contexts independently from any tasks.

Task and context management involves:

- [Enabling Integration with an Issue Tracking System](#)
- [Creating and Deleting Tasks](#)
- [Viewing the Description of a Task](#)
- [Switching Between Tasks](#)
- [Saving and Clearing Contexts](#)
- [Switching Between Contexts](#)

See Also

Reference:

- [Open Task Dialog](#)
- [Tasks](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Enabling Integration with an Issue Tracking System

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Enabling integration between PhpStorm and an issue tracking system allows you to work on your project in the discourse of tasks and contexts and thus set up your workflow in accordance with the process established in your team.

To enable integration with an issue tracking system

1. Access the [Servers](#) dialog box. Do one of the following:
 - [Open the Project Settings](#). Below the `Tasks` node, click `Servers`.
 - In the [Open Task](#) dialog box, click the `Configure` link.
2. In the `Servers` dialog box, specify the following:
 - The URL address of your issue tracking server.
 - Your account credentials on the server in question. These credentials will be different for the different issue tracking systems.
 - Specify whether you want to access the server via proxy and specify the proxy settings.
 - To allow access to the specified server for other members of your team, select the `Share URL` check box.
 - To check whether the specified settings ensure successful connection to the server, click the `Test` button.

See [reference page](#) for the detailed description of controls.

3. Configure synchronization between PhpStorm and your issue tracking system. To do so, [open the Project Settings](#), and click `Tasks`. In the [Tasks](#) page, configure interaction between PhpStorm and your tracker. Do one of the following:
 - To have PhpStorm synchronize with the issue tracking system in the background on a regular basis, select the `Enable issue cache` check box and specify the synchronization frequency and the cache size.

No matter whether you actually request on information from your issue tracker or not, PhpStorm will connect to your issue tracking system according to the specified frequency and refresh the cached issues. The advantage of this approach is that when you need to switch to a task, the up-to-date information is already at your disposal so you do not need to wait till PhpStorm establishes connection with the tracker and retrieves the information.

Tip

This configuration is especially recommended when working with rather "slow" issue tracking systems.

- To have PhpStorm connect to the issue tracking system only when you actually need information on issues, clear the `Enable issue cache` check box.

See Also

Procedures:

- [Managing Tasks and Context](#)

Reference:

- [Servers](#)

- [Tasks](#)

External Links:

- [Bug tracker YouTrack](#)
- [Jira](#)
- [Lighthouse](#)
- [Pivotal tracker](#)
- [Redmine](#)
- [GitHub](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Creating and Deleting Tasks

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

In this section:

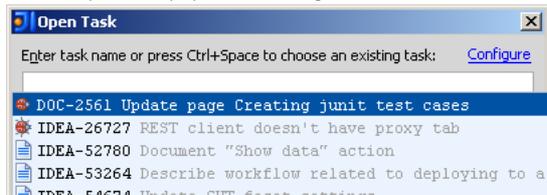
- [Creating tasks](#)
- [Deleting tasks](#)

To create a task

1. Access the [Open Task](#) dialog box. Do one of the following:
 - Choose **Tools | Tasks | Open** on the main menu or press **Alt+Shift+NAlt Shift N**.
 - Choose **Tools | Tasks | Switch** on the main menu or press **Alt+Shift+TAlt Shift T**. Then choose **Open New Task** in the **Switch to Task** pop-up window.
2. In the **Open Task** dialog box, specify the ID of the corresponding issue in your issue tracking system in the **Enter task name or choose existing task** text box.

Tip

- Alternatively, you can specify any name and thus configure and process tasks without integration with your issue tracking system.
- To have PhpStorm display a list of existing tasks to select the relevant task from, press **Ctrl+SpaceCommand Space**.



3. Specify whether you want PhpStorm to clear the current context and create a changelist for the new task by selecting the corresponding check boxes.

Tip

To configure access to your issue tracking system, click the **Configure** link and specify your account settings in the [Servers](#) dialog box that opens.

To delete a task

1. On the main menu, choose **Tools | Tasks | Switch**, or press **Alt+Shift+TAlt Shift T**.
2. Select the task you want to delete.
3. Click the right-arrow button, and choose **Remove** from the list.

See Also

Procedures:

- [Managing Tasks and Context](#)

Reference:

- [Open Task Dialog](#)
- [Servers](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Viewing the Description of a Task

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

When you are [choosing a task to switch to](#), the list in the **Switch to task** pop-up window shows only IDs of tasks and their summaries. This information is not always sufficient, because it reflects neither the steps that lead to the problem nor the related discussion.

You can learn more about your current task in two ways:

- [View the description of the task](#) in a pop-up window.
- [Open the task in the browser](#) and view both the issue description and all the comments on it.

To view the description of the current task in the pop-up window

- Choose **Tools | Tasks | Show '<task ID>' Description** on the main menu.

To open the current task in the browser, do one of the following:

- Choose **Tools | Tasks | Open '<task ID>' in Browser** on the main menu.
- Press **Alt+Shift+BAlt Shift B**.

See Also

Procedures:

- [Switching Between Tasks](#)
- [Managing Tasks and Context](#)
- [Creating and Deleting Tasks](#)

Reference:

- [Servers](#)
- [Tasks](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Switching Between Tasks

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: 

To switch to another task

1. Choose **Tools | Tasks | Switch** on the main menu or press **Alt+Shift+TAlt Shift T**.
2. In the **Switch to Task** pop-up window, select the desired task from the list.

Tip

Alternatively, choose **Open New Task** and proceed as during [task creation](#).

See Also

Procedures:

- [Managing Tasks and Context](#)
- [Viewing the Description of a Task](#)

Reference:

- [Open Task Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Saving and Clearing Contexts

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: 

With PhpStorm, you can save and clear [contexts](#) without associating them with specific [tasks](#).

To manage the current context independently from a task, do one of the following:

- To save the current context, choose **Tools | Context | Save** on the main menu or press **Alt+Shift+SAlt Shift S**.
- To clear the current context without loading another one, choose **Tools | Context | Clear** on the main menu or press **Alt+Shift+XAlt Shift X**.

See Also

Procedures:

- [Creating and Deleting Tasks](#)
- [Managing Tasks and Context](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Switching Between Contexts

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts:  

With PhpStorm, you can switch between [contexts](#) that are not associated with specific [tasks](#).

To switch to another context

1. Choose **Tools | Context | Load** on the main menu or press **Alt+Shift+LAlt Shift L**.
2. In the **Load Context** pop-up window, select the desired context from the list.

See Also

Procedures:

- [Switching Between Tasks](#)
- [Managing Tasks and Context](#)

Web Resources:

- <http://www.jetbrains.com/idea/faq/faq-contexts.html> 
- <http://youtrack.jetbrains.com/issues/WI-10000> 

2.0+

Using Command Line Tools

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

You can run command line commands right from PhpStorm. Just configure the commands that fit your preferences and needs as a **command line tool** and register it in PhpStorm. You can have as many **command line tools** as you need.

At the PhpStorm level, integration with command line tools is supported via the **Command Line Tool** bundled plugin. The plugin is by default enabled. If not, enable it in the [Plugin Configuration Wizard](#) or the [Plugins](#) page of the [Settings](#) dialog box.

The available tools are listed in the [Command Line Tool Support](#) page of the [Settings](#) dialog box.

To work with a command line tool, perform these general steps:

- [Create and configure a command line tool](#).
- [Invoke commands](#) and analyze the results of their execution in the dedicated [Command Line Tools Console](#) tool window.

See Also

Procedures:

- [PHP-Specific Command Line Tools](#)

Reference:

- [Command Line Tool Support](#)
- [Command Line Tools Console Tool Window](#)

Web Resources:

- <http://www.jetbrains.com/idea/faq/faq-contexts.html> 
- <http://youtrack.jetbrains.com/issues/WI-10000> 

Creating a Command Line Tool

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

The command line tools that are available from PhpStorm are listed in the [Command Line Tool Support](#) page of the [Settings](#) dialog box. Once enabled, integration with a tool can be [de-activated and re-activated](#) at any time.

You can [update the command definitions](#) of a tool and have the `.xml` definition file validated right in the PhpStorm editor.

To create and configure a command line tool

1. [Open the project settings](#) and click the **Command Line Tool Support**.
2. In the [Command Line Tool Support](#) page, click the **Add** button. In the **Choose Framework to Add** dialog box that opens, choose **Custom Framework**.
3. In the [Framework Settings](#) dialog box that opens, specify the following:
 1. Tool name and location.
 2. The alias to use in command calls instead of the full path to the tool.
 3. Brief explanation of the tool functionality.
4. When you click **OK**, PhpStorm brings you to the [Command Line Tool Support](#) page, where the new tool is added to the list of available frameworks.
5. Select the newly created tool and click the **Open definition in editor** button. In the tool definition `.xml` file that opens in the editor, define the tool commands.
6. In the **Show console in** area, specify where you want the **Input** pane for [typing commands](#) opened:
 - To have the **Input** pane opened in a pop-up window, choose the **Pop-up** option.

- o To have the **Input** pane opened as a text box at the bottom of the [Command Line Tools Console](#) tool window, choose the **Tool window** option.

To activate a configured command line tool

1. [Open the project settings](#), and click the **Command Line Tool Support**.
2. Select the **Enable** check box next to to the tool name.

See Also

Procedures:

- [Using Command Line Tools](#)
- [PHP-Specific Command Line Tools](#)
- [Running Command Line Tool Commands](#)
- [PHP-Specific Guidelines](#)

Reference:

- [Command Line Tool Support](#)
- [Command Line Tools Console Tool Window](#)
- [PHP](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Running Command Line Tool Commands

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: 

You can [run commands](#) and analyze their output right in PhpStorm using the dedicated [Command Line Tools Console](#) tool window. If necessary, you can [export the output](#).

To execute a command in the command line

1. To access the **Input** pane, choose **Tools | Run Command** on the main menu or press `Ctrl+Shift+X` `Ctrl` `Shift` `X`.

Note

Depending on the **Show console** in setting in the [Command Line Tool Support](#) page, the **Input** pane opens either as a pop-up window or as a text box at the bottom of the dedicated [Command Line Tools Console](#) tool window.

2. In the **Input** pane, type the call of the command in the format: `<tool alias> <command>`

Tip

1. Besides framework-specific commands, you can also run simple, operating system-wide, commands, such as `ls`, etc.
2. Most framework tools operate within the context of their own projects. A **framework tool project** is not the same as a **PhpStorm project**. Therefore, when using a tool from PhpStorm for the first time, you need to create its own "initial" framework tool project besides you current PhpStorm project. To do so, run the project generation command specific for the current tool in the **Command Line Tool Input** pane. The "initial project" is located in the PhpStorm project root.
3. View the result of command execution in the **output** tab named after the last invoked command.

To terminate a running command, do one of the following:

- During the command execution, click the **Stop** toolbar button .
- Kill the current thread from the progress bar, if the **output** tab is already closed.

To export the output

1. Click the **Export to Text** toolbar button  or press `Alt+O` `Command` `O`.
2. In the **Export Preview** dialog box that opens, do one of the following:
 - o To have the output saved in a text file, specify the target text file in the **Export to file** text box and click the **Save** button. If the file with this name and location already exists, choose to overwrite its contents with the new data or to append the new data to the existing file.
 - o To copy the output to the clipboard, click the **Copy** button.

See Also

Procedures:

- [Using Command Line Tools](#)
- [PHP-Specific Command Line Tools](#)
- [PHP-Specific Guidelines](#)

Reference:

- [Command Line Tools Console Tool Window](#)
- [Command Line Tool Support](#)
- [PHP](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Updating a Command Line Tool

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

You can update the command definitions of a command line tool right in the PhpStorm editor and have the .xml definition file [validated](#). As you type, the file is checked for well-formedness on the fly. [Full validation](#) is performed every time you invoke a command. If any inconsistencies are detected, the tool is marked with the **Invalid description** icon  in the [Command Line Tool Support](#) page.

To edit the command definitions

1. [Open the project settings](#) and click the **Command Line Tool Support**.
2. In the [Command Line Tool Support](#) page, select the desired framework and click the **Open definition in editor** button. The corresponding .xml definition file opens in the editor. Update the file as necessary.

To validate the definition file for structure consistency

1. [Start executing a command](#): choose **Tools | Run Command** on the main menu or press `Ctrl+Shift+X` `Shift X`. Then type the call of the command in the **Input** tab of the [Command Line Tools Console](#) tool window.
2. View the results in the **Framework definition file errors** tab. As soon as PhpStorm detects any structure inconsistency, it displays an error message that contains a brief description of the problem and indicates the line number where the problem is detected.

Note

By default, the tab is not displayed and opens only upon validation failure. In this case, PhpStorm displays a **Command Line Tool** pop-up window with a notification on validation failure. Upon clicking the **More** link, the **Framework definition file errors** tab opens showing messages on detected inconsistencies. Each message contains information on the file and the line number where the problem was found, as well as a brief description of the error.

You can close the tab by clicking the cross on its header. To re-open the tab, again click **More** in the **Command Line Tool** notification pop-up window, which remains on the screen until you close it manually.

See Also

Procedures:

- [Using Command Line Tools](#)
- [PHP-Specific Guidelines](#)

Reference:

- [Command Line Tools Console Tool Window](#)
- [Command Line Tool Support](#)
- [PHP](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Language and Framework-Specific Guidelines

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

This part provides descriptions of the actions required to fulfil certain tasks using the frameworks integrated into PhpStorm:

- [PHP-Specific Guidelines](#)
- [JavaScript-Specific Guidelines](#)
- [CoffeeScript Support](#)
- [NodeJS](#)

See Also

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

PHP-Specific Guidelines

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

This part describes some procedures that are specific for developing PHP applications and some preliminary steps that are required to configure PHP development environment.

To develop an application using php

Follow these general steps:

1. [Configure the PHP development environment](#).

2. [Enable PHP](#) support in PhpStorm.
3. [Create a project](#) from scratch or around existing sources.

Tip

To run and debug your application on a local Web server, create the project root below the Web server document root. Thus your application sources will be "visible" for the local Web server.

4. [Create](#) and [configure](#) the required data sources.
5. [Populate the application](#) using provided coding assistance.
6. [Deploy](#) the application.

Tip

With PhpStorm, you can flexibly configure deployment of PHP applications. For example, you can set up your PHP project on a local Web server from the very beginning, or develop and test an application locally and then upload it to a remote Web server, etc.

7. [Run](#) the application. You can do it in several ways:
 - o From PhpStorm using a [run configuration](#) of the type [PHP Web Application](#) to view application output in a browser.
 - o From PhpStorm using a [PHP Console](#) run configuration to view the application output in the Run tool window.
 - o Open the application starting page in your browser.

Tip

The following optional steps may be helpful:

- [Set up unit testing](#) in your project.
- [Install and configure a debugging engine](#) and specify the [debugger options](#).
- [Debug](#) the application.
- Enable and use integration with [PHP-specific command line tools](#).

See Also

Procedures:

- [Creating and Managing Projects](#)
- [Data Sources](#)
- [Creating and Editing Run/Debug Configurations](#)

Reference:

- [PHP](#)
- [Deployment](#)
- [Debugger](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Configuring PHP Development Environment

Previous | [Next](#) | [See Also](#) | [Comments](#)

Warning

This topic gives general guidelines in configuring environment for developing and testing PHP applications locally. These instructions by no means apply to configuring production environment, which is outside the scope of this Help.

PHP development requires the following software installed and configured:

- A Web server and a PHP engine are mandatory.
- A database server, if your application will use a database.
- A debugging tool, if you are going to debug your application.
- A command line tool, if you are going to run PHP commands from the command line.

To set up the php development environment

1. Download, install, and configure the Web server, the PHP engine, and the MySQL server. Do one of the following:
 - o Download, install, and configure the desired [AMP package](#) (Apache, MySQL, PHP).
 - o Install and configure [each component separately](#), then [check your installation](#).
2. Optionally, perform these steps:
 - o Install and configure a [debugging engine](#).
 - o Install or create and enable a [PHP-specific command line tool](#).

See Also

Procedures:

- [Enabling PHP Support](#)
- [Enabling a Command Line Tool](#)
- [Configuring a Debugging Engine](#)

Reference:

- [PHP](#)
- [Command Line Tool Support](#)

External Links:

- <http://httpd.apache.org/>
- <http://ru.php.net/downloads.php>
- <http://www.xdebug.org/>

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Installing an AMP Package

Previous | [Next](#) | [See Also](#) | [Comments](#)

AMP packages are operating system-specific. The most common ones are:

- [XAMPP](#) for Windows.

Tip

It is recommended that you use version 1.7.1 or later.

- The LAMP package compatible with the Linux distribution used.
- [MAMP](#) for MAC OS X.

The installation procedure may differ depending on the operating system used, follow the installation instructions provided.

Warning

If you are using Windows Vista, avoid installing the package in the `Program Files` folder. This folder is write-protected by default, which means that no files can be placed on the server and further processed by the PHP engine.

To install and configure an amp package

1. Download and install the desired AMP package.
2. Use the AMP control pane to start the components.

Tip

If the server does not start, most likely a port conflict takes place. By default, the Apache HTTP server listens to port 80. This port can be already used by other services, for example, Skype. To solve the issue, update the server configuration file as follows:

- Locate the line `Listen 80` and change it to, e.g., `Listen 8080`.
- Locate the line `ServerName localhost:80` and change it accordingly, in this example to `ServerName localhost:8080`.

Save the configuration file and restart the Web server.

3. To check your installation, open your browser and type the following URL address: `http://localhost:<port number>`. The AMP welcome page appears.

See Also

Procedures:

- [PHP-Specific Guidelines](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

```
ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js'; var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s); }());
```



PhpStorm 3.0.0 Web Help

Installing Components Separately

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

The installation procedure for PHP environment components may differ depending on the operating system used, follow the installation instructions provided.

IN this section:

- [Installing and configuring the components separately](#)
- [Checking the Web server installation](#)
- [Checking the PHP engine installation](#)

To install and configure each component separately

1. Download and install a [Web server](#). The most common choice is the [Apache HTTP server](#).
2. Download and install the [PHP engine](#). During the installation, specify the Web server you are going to use and its home directory.

Tip

To enable the use of the MySQL database server, select the **Complete installation** option or select the MySQL and MySQLi items in the **Extensions** list on the corresponding page of the installation wizard..

3. Download, install, and configure a [database server](#).

To check the web server installation

- To make sure that the Web server has been installed correctly, start the server, open your browser, and type the following URL address: `http://localhost`

The Test page for Apache installation should appear.

Tip

- If the server does not start, most likely a port conflict takes place. [Resolve the conflict](#) as during an AMP package installation.
- If you have changed the default port 80, specify the port number explicitly: `http://localhost:<port number>`

To check the php engine installation

Finally, make sure that the PHP engine has been installed successfully and interacts with the Web server correctly.

1. Using a text processor of your choice, create a file `test.php` and type the following text:

```
<?php
    echo phpinfo();
?>
```

2. Save the file on the Web server, in the folder the PHP engine looks at.
3. Run the browser and enter the following URL address: `http://localhost:<port number>/test.php`. The page that opens should show detailed information on the version of PHP engine used, the location of the configuration files, etc.



Click thumbnail to view larger image.

See Also

Procedures:

- [PHP-Specific Guidelines](#)

External Links:

- <http://httpd.apache.org/>
- <http://ru.php.net/downloads.php>

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Enabling PHP Support

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PHP development support at the PhpStorm level is provided via the `PHP` bundled plugin. [PHP deployment](#) is provided via the `Remote Hosts Access` bundled plugin. Both plugins are by default activated. If not, [enable](#) them in the [Plugin Configuration Wizard](#) or the [Plugins](#) page of the [Settings](#) dialog box.

You can have as many PHP engine versions installed and configured at the PhpStorm level and switch among them. However, within a project, you can have only one of these configurations activated

In this sections:

- [Configuring a PHP installation in PhpStorm](#)
- [Enabling PHP development support in a project](#)

To configure a PHP installation in PhpStorm

1. [Open the project settings](#) and click **PHP**.
2. Click the **Browse** button  next to the **Interpreter** drop-down list in the **Development environment** section.
3. In the [Interpreters](#) dialog box, that opens, click the **Add** toolbar button  on the left-hand pane.
4. On the right-hand pane of the dialog box, specify the PHP installation settings.
 - In the **Name** text box, type the identifier to distinguish the installation from others, for example, `php_installation_<version>`.
 - Specify the PHP engine installation directory in the **PHP Home** field. Type the path manually or click the **Browse** button  and choose the location in the **Choose PHP Home** dialog box, that opens.
 - In the **Debugger** drop-down list, specify the debugging engine to use in the project.

Tip

The chosen [debugging engine must be enabled](#) in the `php.ini` file in the specified PHP installation directory.

5. Optionally, customize the configuration settings of the installation. In the **Configuration options** field, compose a string of configuration directives to be passed through the [-d command line option](#) and thus add new entries to the `php.ini` file. To do that, click the **Browse** button  next to the **Configuration options** field, and then create a list of entries in the **Configuration Directives** dialog box, that opens.
 - To add a new entry, click the **Add** button . In the new line, that is added to the list, specify the name of the new entry and its value in the **Name** and **Value** text boxes respectively.

Note

You can add as many entries as you need, just keep in mind that they will be transformed into a command line with its length limited to 256 characters.

- To delete an entry, select it in the list and click the **Remove** button .
- To change the order of entries, use the **Up**  and **Down**  buttons.

Upon clicking **OK**, you return to the **Interpreters** dialog box, where the entries are transformed into a command line.

To enable PHP development support in a project

1. [Open the project settings](#) and click **PHP**.
2. In the [PHP](#) page that opens, choose the installation to use from the **Interpreter** drop-down list.
 - To make sure that the configuration you have chosen points at the relevant installation, click the **Reload** button  next to the drop-down list. If no PHP engine is actually installed at the specified directory, PhpStorm displays the corresponding error message.
 - To examine the installation details, click the **Show phpinfo** button  next to the drop-down list and view the full list of installation settings in a separate window. Actually, PhpStorm checks the installation and displays the result of executing the `phpinfo` command.

See Also

Procedures:

- [Configuring PHP Development Environment](#)
- [Creating New Project from Scratch](#)
- [Creating New Project from Existing Source Code](#)
- [Creating and Saving Temporary Run/Debug Configurations](#)

Reference:

- [PHP](#)
- [Interpreters](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

2.1+

Using Distributed Configuration Files (.htaccess)

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm provides syntax highlighting, formatting, [code completion](#) and [documentation lookup](#) while editing [distributed configuration files](#). Distributed configuration files are used to make directory based changes to the [HTTP Apache Server](#) configuration and usually have the name `.htaccess`.

If you are using a file with the name `.htaccess`, PhpStorm recognizes it as distributed configuration file and provides full coding assistance for it, so no additional steps are required from your side.

If you want to use a file with another name, you need to associate this full name or the corresponding pattern with the distributed configuration [file type](#). After that, PhpStorm will treat any file with the name matching the specified pattern as a distributed configuration file and process it accordingly.

To associate a name pattern with the distributed configuration file type

1. [Open the IDE settings](#) and click **File Types**.
2. In the [File Types](#) page that opens, select **Apache config files** from the **Recognized File Types** list.

3. In the **Registered Patterns** area, click the **Add** button.
4. In the **Add wildcard** dialog box that opens, specify the pattern that defines the extensions of your distributed configuration files.
5. Click **OK**. PhpStorm returns you to the **File Types** page where the specified pattern is added to the **Registered Patterns** list.

Note

- By default, the **Registered Patterns** list contains one item `htaccess`.
- To discard a pattern, select it in the list and click the **Remove** button.
- To change a pattern, select it in the list, click the **Edit** button, and update the pattern as necessary in the **Add wildcard** dialog box that opens.

See Also

Procedures:

- [Creating and Registering File Types](#)
- [Configuring PHP Development Environment](#)
- [Enabling PHP Support](#)
- [PHP-Specific Guidelines](#)

Reference:

- [File Types](#)

External Links:

- <http://httpd.apache.org/docs/trunk/howto/htaccess.html>
- <http://httpd.apache.org/docs/trunk/configuring.html#syntax>

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>
-

Configuring Include Paths

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

To use [PHP-related items](#) from outside your project content root, you need to have them stored under the PHP home directory and specify the relative paths to them. The specified include paths will be used:

- By the `require()`, `include()`, `fopen()`, `file()`, `readfile()`, and `file_get_contents()` functions when looking for files to use.
- By PhpStorm when resolving references to included files.

To configure include paths

- [Open the project settings](#) and click **PHP**.
- In the **PHP** page that opens, click the **Update include paths** button.
- In the **Include paths** area, that becomes available upon pressing the **Update include paths** button, configure the list of include paths.
 - To add a path to the list, click the **Add** button and select the required directory in the dialog box, that opens.
 - To remove a path from the list, select the required item and click the **Remove** button.

See Also

Procedures:

- [Configuring Project Structure](#)
- [PHP-Specific Guidelines](#)

Reference:

- [PHP](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>
-

Running PHP Applications

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: 

You can run your PHP application in several ways:

- From PhpStorm using a [run configuration](#) of the type [PHP Web Application](#) to view application output in a browser.
- From PhpStorm using a [PHP Console](#) run configuration to view the application output in the Run tool window.
- Open the application starting page in your browser.

PhpStorm enables running [entire PHP applications](#) as well as particular [classes or files](#).

To run an entire php application, perform the following general steps

1. Define a PHP run configuration:
 - o To launch the application via a console, create a [PHP Console](#) configuration.
 - o To run the application on a local or a remote Web server, create a [PHP Web Application](#) configuration.
2. [Deploy](#) the PHP application on a local or remote server, if applicable.
3. [Run](#) the application with the required run configuration.

To run a particular php class or file, do one of the following

- Open the desired class or file in the editor and choose **Run <file name>** on the context menu or press **Ctrl+Shift+F10** **Command Shift F10**.
- Select the class or file in the Project view and choose **Run <file name>** on the context menu of the selection.

See Also

Concepts:

- [Run/Debug Configuration](#)

Procedures:

- [Running Applications](#)

Reference:

- [Run Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

3.0.0.+

Creating PHP Documentation Comments

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

PhpStorm creates stubs of [PHPDoc blocks](#) on typing the opening tag `/**` and pressing **EnterEnter**.

If this feature is [applied to a method or a function](#), `@param`, `@access`, `@static`, `@throws`, and `@abstract` tags are created. In any other places PhpStorm adds an empty documentation stub.

If you need [additional PHP-specific tags](#), PhpStorm provides code completion that suggests tag names that are relevant in the current context. If a certain tag has multiple values, the same code completion provides a list of available values.

In PHPDoc comments, PhpStorm supports the following formatting options in compliance with ZEND, PEAR, and other standards:

- Align parameter names
- Align tag comments
- Align key-value pairs
- Align consecutive assignments

Note

PHPDoc comments in your source code are available for the [Quick Documentation Lookup](#) feature and open for review on pressing **Ctrl+Q** **Command J**.

To create a phpdoc block for a method or a function

1. Place the caret before the method or function declaration.
2. Type the opening block comment `/**` and press **EnterEnter**.
3. Describe the listed parameters and return values.

Tip

PhpStorm checks syntax in the comments and treats it according to the [PHP Inspections](#) settings.

To create tags in a phpdoc comment block

1. In a PHPDoc block, select the desired empty line and press **Ctrl+Space** **Command Space**.
2. Select the relevant tag from the suggestion list.
3. If the entered tag has several values, press **Ctrl+Space** **Command Space** and select the desired value from the suggestion list.

See Also

Procedures:

- [Viewing Inline Documentation](#)

Reference:

- [Inspections](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Plugins

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Quick Documentation Lookup helps you get quick information for any symbol, provided that this symbol has been supplied with Documentation comments in the applicable format. PhpStorm recognizes inline documentation created in accordance with the [PHP Documentation](#) format.

In this section:

- [Viewing quick documentation.](#)
- [Changing font size in the quick documentation window.](#)
- [Toolbar of the quick documentation window.](#)

To view documentation for a symbol at caret, do one of the following

- On the main menu, choose **View | Quick Documentation Lookup**.
- Press **Ctrl+Q/Command J**.

Note

When you explicitly invoke code completion, then quick documentation for an entry selected in the suggestion list can be displayed automatically. The behavior of quick documentation lookup is configured in the [Code Completion](#) page of the Settings dialog.

To change the font size of quick documentation, do one of the following

- Click the  button in the upper-right corner of the quick documentation window, and move the slider.
- Rotate the mouse wheel while keeping the **Ctrl/Ctrl** key pressed.

Quick documentation lookup window

The **Quick Documentation Lookup** window helps navigate to the related symbols via hyperlinks, and provides a toolbar for moving back and forth through the already navigated pages, change font size, and viewing documentation in an external browser.

Icon	Keyboard shortcut	Action
	Alt+Shift+LeftCommand Shift Left Alt+Shift+RightCommand Shift Right	Navigate to the previous or next screen in the definition pop-up window after using hyperlinks in the definition. Note On a Mac OS X computer, you can also use the three-finger right-to-left and left-to-right swipe gestures.
	Shift+F1/Shift F1	View documentation in an external browser.
		Click this button to show font size slider. Move the slider to increase or decrease the font size in the quick documentation window as required.

See Also

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Deploying PHP Applications

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

[Deployment](#) of PHP applications is provided via the **Remote Hosts Access** bundled plugin. The plugin is by default activated. If not, [enable](#) it in the [Plugin Configuration Wizard](#) or the [Plugins](#) page of the [Settings](#) dialog box.

With PhpStorm, you can flexibly configure deployment of PHP applications. For example, you can set up your PHP project on a local Web server from the very beginning, or develop and test an application locally and then upload it to a remote Web server, etc.

By default, the sources are deployed according to the chosen configuration when you run your application. However, you can have PhpStorm [upload files automatically](#) every time they are changed. In this case the default server configuration is used.

To configure deployment

1. Create a local or remote [Web server configuration](#).
2. [Define mappings](#) between local folders, remote folders, and URL addresses.
3. [Customize the upload](#) procedure by specifying additional options.

To have updated files uploaded automatically, do one of the following

- Choose **Tools | Deployment | Automatic Upload** on the main menu.
- [Open the project settings](#) and click **Options** below the [Deployment](#) node. In the Options dialog box, select the **Upload changed files automatically to the default server** check box.

See Also

Procedures:

- [Remote Hosts](#)
- [PHP-Specific Guidelines](#)

Reference:

- [Deployment](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>
-

Debugging PHP Applications

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

In this part:

- [Configuring a Debugging Engine](#)
- [PHP Debugging Session](#)
- [Creating a PHP Web Application Debug Configuration](#)
- [Multiuser Debugging Via XDebug Proxies](#)
- [Debugging a PHP HTTP Request](#)

See Also

Procedures:

- [Debugging](#)

Language and Framework-Specific Guidelines:

- [Profiling the Performance of a PHP Application](#)
- [PHP-Specific Guidelines](#)

Reference:

- [PHP](#)
- [Debug](#)
- [Deployment](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>
-

Configuring a Debugging Engine

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

This section provides description of configuring two most popular tools used in PHP debugging:

- [XDebug](#)
- [Zend Debugger](#)

Warning

These tools cannot be used simultaneously because they block each other. To avoid this problem, you need to update the corresponding sections in the `php.ini` file.

See Also

Procedures:

- [Configuring PHP Development Environment](#)
- [Creating and Editing Run/Debug Configurations](#)

Reference:

- [Debug](#)
- [PHP](#)

External Links:

- <http://www.xdebug.org/>
- <http://www.zend.com/en/>
- <http://static.zend.com/topics/Zend-Debugger-Installation-Guide.pdf>
- <http://www.zend.com/en/products/studio/downloads>

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Configuring XDebug

Previous | [Next](#) | [See Also](#) | [Comments](#)

Configuring the XDebug tool involves:

- [Downloading and installing the tool.](#)
- [Updating the php.ini configuration file](#) with XDebug-specific settings.
- Checking the installation.
- [Integrating XDebug](#) with PhpStorm.

To install the XDebug debugging engine

- Download the [XDebug](#) extension compatible with your version of PHP and save it in the `php/` folder.

Note

The location of the `php/` folder is defined during the [installation of the PHP engine](#).

Tip

If you are using an [AMP package](#), the XDebug extension may be already installed. Follow the instructions in the `xdebug.txt`.

To enable XDebug integration with the php engine

1. Locate and open the active `php.ini` file.

Tip

To find out which `php.ini` file is active, [create and run a test file](#) with `phpinfo()`, then search for the Loaded Configuration File.

2. To disable the Zend optimizer, which blocks XDebug, remove the `[Zend]` section or mark it as comment in the `php.ini` file:

```
[Zend]
zend_extension_ts = "<path to ZendExtensionManager.dll>"
zend_extension_manager.optimizer_ts = "<path to Zend Optimizer>"
zend_optimizer.enable_loader = 0
zend_optimizer.optimization_level=15
zend_optimizer.license_path =
Local Variables:
tab-width: 4
End:
```

3. To enable XDebug, locate the `[XDebug]` section in the `php.ini` file and update it as follows:

- o For a thread-safe PHP 5.2 engine:

```
[XDebug]
zend_extension_ts="<path to php_xdebug.dll>"
xdebug.remote_enable=true
xdebug.remote_port="<the port for XDebug to listen to>" (the default port is 9000)
xdebug.profiler_enable=1
xdebug.profiler_output_dir="<AMP home\>tmp"
```

- o For a non-thread-safe PHP 5.2 engine:

```
[XDebug]
zend_extension_nts="<path to php_xdebug.dll>"
xdebug.remote_enable=true
xdebug.remote_port="<the port for XDebug to listen to>" (the default port is 9000)
xdebug.profiler_enable=1
xdebug.profiler_output_dir="<AMP home\>tmp"
```

- o For any PHP 5.3 engine:

```
[XDebug]
zend_extension="<path to php_xdebug.dll>"
xdebug.remote_enable=1
xdebug.remote_port="<the port for XDebug to listen to>" (the default port is 9000)
xdebug.profiler_enable=1
xdebug.profiler_output_dir="<AMP home\>tmp"
```

4. To enable [multiuser debugging via Xdebug proxies](#), locate the `xdebug.idekey` setting and assign it a value of your choice. This value will be used to register your IDE on Xdebug proxy

servers.

5. Save and close the `php.ini` file.

To check the XDebug installation

1. Open the [XDebug checker](#).
2. Enter the output of the `phpinfo()`.

To integrate XDebug with PhpStorm

1. Open the [Project Settings](#) and click **PHP**.
2. In the [PHP](#) page that opens, choose **XDebug** from the **Debugger** drop-down list.
3. Define the XDebug behaviour. Click **Debug** under the **PHP** node. On the [Debug](#) page that opens, specify the following settings in the **XDebug** area:
 - o In the **Debug Port** text box, appoint the port through which the tool will communicate with PhpStorm. This must be exactly the same port number as specified in the `php.ini` file:

```
xdebug.remote_port = <port_number>
```

By default, Xdebug listens to port 9000.

- o To have PhpStorm accept any incoming connections from XDebug engines through the port specified in the **Debug port** text box, select the **Can accept external connections** check box.
- o To have addresses of variables shown during a debugging session, select the **Show variables addresses** check box.
- o To have the debugging engine stop at the first line automatically, select the **Stop at first line** check box.
- o To have PhpStorm apply a workaround to overcome a [known XDebug on FreeBSD crash issue](#), select the **Enable workaround for [XDebug on FreeBSD crash]** check box.

See Also

Procedures:

- [Configuring PHP Development Environment](#)
- [Creating and Editing Run/Debug Configurations](#)
- [Multiuser Debugging Via XDebug Proxies](#)
- [Configuring Zend Debugger](#)

Reference:

- [Debug](#)
- [PHP](#)

External Links:

- <http://www.xdebug.org/>

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

2.0+

Configuring Zend Debugger

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Configuring the Zend Debugger tool involves:

- [Downloading and installing the tool](#).
- [Updating the php.ini configuration file](#) with Zend debugger settings.
- [Integrating Zend Debugger](#) with PhpStorm.

To download and install zend debugger

1. Download the [Zend Debugger package](#) which corresponds to your operating system.
2. Locate the `zenddebugger.so` (Unix) or `zenddebugger.dll` (Windows) file in the directory which corresponds to your version of PHP (e.g. 4.3.x, 4.4.x, 5.0.x, 5.1.x, 5.2.x).
3. Copy the file to your Web server in a location that is accessible by the Web server.

To enable zend debugger integration with the php engine

1. Locate and open the active `php.ini` file.

Tip

To find out which `php.ini` file is active, [create and run a test file](#) with `phpinfo()`, then search for the Loaded Configuration File.

2. Locate or create the `[Zend]` section.
3. To load the Zend Debugger extension, add one of the following lines inside the `[Zend]` section depending on your operating system:
 - o **Linux and Mac OS X:**

```
zend_extension=<full_path_to_zendDebugger.so>
```

- o **Windows:**

```
zend_extension_ts=<full_path_to_ZendDebugger.dll>
```

- o Windows non-thread safe:

```
zend_extension=<full_path_to_ZendDebugger.dll>
```

Warning

The Windows non-thread safe binary file is only used with Zend Core 2.0.

4. To enable access to Zend Debugger from PhpStorm, add the following lines:

```
zend_extension=<full_path_to_zend_debugger_extension>
zend_debugger.allow_hosts=127.0.0.1
zend_debugger.expose_remotely=allowed_hosts
```

The value of the `zend_debugger.allow_hosts` parameter is the IPs of your machine to connect to the server debugger. It could be a comma-separated list of IPs in the format `x.x.x.x` (for example, `192.168.0.6`).

Tip

For a thread-safe Windows binary use the `zend_extension_ts` parameter instead of `zend_extension`.

5. Restart your Web server.
6. To check that the Zend Debugger has been installed and configured correctly, create a file with the following contents:

```
<?php
    phpinfo();
?>
```

Open the page that corresponds to the file in the browser. The output should contain a **Zend Debugger** section.

To integrate zend debugger with PhpStorm

1. Open the [Project Settings](#) and click **PHP**.
2. In the [PHP](#) page that opens, choose **Zend Debugger** from the **Debugger** drop-down list.
3. Define the Zend Debugger behaviour. Click **Debug** under the **PHP** node. On the [Debug](#) page that opens, specify the following settings in the **Zend Debugger** area:
 - o In the **Debug Port** text box, appoint the port for PhpStorm to communicate with the tool through. Type the port number specified in the `php.ini` file.
 - o To have PhpStorm accept any incoming connections from Zend Debugger engines through the port specified in the **Debug port** text box, select the **Can accept external connection** check box.
 - o To use a debugger toolbar in the browser, specify the port through which the debugger settings are passed to the browser in the **Settings broadcasting port** text box.

See Also

Procedures:

- [Configuring PHP Development Environment](#)
- [Creating and Editing Run/Debug Configurations](#)
- [Configuring XDebug](#)

Reference:

- [Debug](#)
- [PHP](#)

External Links:

- <http://www.zend.com/en/>
- <http://static.zend.com/topics/Zend-Debugger-Installation-Guide.pdf>
- <http://www.zend.com/en/products/studio/downloads>

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

2.0+

PHP Debugging Session

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm supports three main approaches to initiating a PHP debugging session:

- PhpStorm launches the application, opens the browser, and activates the debugging engine according to a [debug configuration](#).
- PhpStorm accesses a specific page through a request generated on the base of a [PHP HTTP Request debug configuration](#).

- [Zero-configuration debugging](#), when the user opens the starting page of the PHP application in the browser manually, and then activates the debugging engine from the browser, while PhpStorm listens to incoming debugger connections. This method requires that you previously enable control over the debugger session from the browser.

During the debugging session on the local or remote server, the server side of the debugger tells PhpStorm the name of the currently processed file and the number of the line to be processed. PhpStorm opens the local copy of this file and indicates the line with the provided number. This behaviour is enabled by specifying correspondence between files and folders on the server and their local copies. This correspondence is called `mapping`.

When using Xdebug, you can also [debug PHP applications in the multiuser mode](#) via Xdebug proxy servers.

To debug a php application according to a configuration

1. [Define a debug configuration](#) of the type **PHP Web Application**.
2. [Set the breakpoints](#), where necessary.
3. Click the **Debug** button  on the toolbar.
4. [Examine the application](#) as soon as the debugger suspends on reaching the first breakpoint.
5. To control the program execution manually, [step through the code](#) using menu commands or toolbar buttons.
6. To have the program run automatically up to the next breakpoint, [resume](#) the session.

To enable starting and stopping a debugging session through control over the debugger cookie, do one of the following

- Specify `GET/POST` or `COOKIE` parameters for [Zend Debugger](#)  or for [Xdebug](#)  manually.
- Generate bookmarklets through which you will start/stop a debugging session by controlling the debugger cookie.
 1. [Open the Project Settings](#) and click **PHP**. Then click **Debug** under the **PHP** node.
 2. In the **Debug** page, that opens, click the **Use debugger bookmarklets to initiate debugger from your favorite browser** link.
 3. On the [Zend Debugger & XDebug bookmarklets](#)  page that opens, check the debugging engine settings and click **Generate**. The bookmarks for listed debugging-related actions are generated.
 4. Add the generated links to the list of your bookmarks.

To debug a php application without a debug configuration

1. [Enable control over the debugger from the browser](#), if you have not done it yet.
2. To prepare the PhpStorm for a debugging session,
 1. [Set the breakpoints](#), where necessary.
 2. Toggle the **Start Listen PHP Debug Connections** button  so it changes its color to green . After that PhpStorm starts listening to the port of the [debugging engine used in the current project](#). Ports for debuggers are set at the PhpStorm level in the [Debug](#) dialog box (**File | Settings | PHP | Debug** for Windows and Linux, **PhpStorm | Preferences | PHP | Debug** for Mac OS).
3. Open the starting page of your application in the browser.
4. To activate the debugging engine from the browser, choose the `<debugging_tool> Start session` bookmark.
5. Re-load the current page (the starting page of the application).
6. Switch to PhpStorm.
7. [Examine the application](#) as soon as the debugger reaches the first breakpoint and suspends.
8. To control the program execution manually, [step through the code](#) using menu commands or toolbar buttons.
9. To have the program run automatically up to the next breakpoint, [resume](#) the session.

See Also

Procedures:

- [Debugging](#)

Reference:

- [Debug Tool Window](#)
- [Run/Debug Configuration: PHP Web Application](#)

External Links:

- <http://blogs.jetbrains.com/webide/2011/02/zero-configuration-debugging-with-xdebug-and-phpstorm-2-0/#more-1364> 

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Creating a PHP Web Application Debug Configuration

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac 

A **PHP Web Application** debug configuration tells PhpStorm where the PHP application to debug is deployed, the URL address to access the starting page of the application, the browser to open the starting page in, and the correspondence between files and folders deployed on the server and their local copies (`mappings`).

To create a php web application debug configuration

1. Open the [Run/Debug Configuration](#) dialog box by doing one of the following:
 - o On the main menu, choose **Run | Edit Configurations**.
 - o Press `Alt+Shift+F10` `Alt Shift F10`, then press `00` to display the **Edit Configuration** dialog box or select the configuration from the pop-up window and press `F4F4`.
2. Click  on the toolbar or press `InsertInsert`. From the drop-down list, select the **PHP Web Application** configuration type. The [PHP Web Application](#) dialog box opens.
3. Specify the configuration name.

- Choose the applicable debug server configuration from the **Server** drop-down list or click the **Browse** button  and [define a debug server configuration](#) in the **Servers** dialog box that opens.
- In the **Start URL** text box, type the server path to the file that implements the application starting page. Specify the path relative to the [server configuration root](#).

Note

The read-only field below shows the URL address of the application starting page. The URL address is composed dynamically as you type.

- Specify the browser to open the application in. Choose a configured browser from the **Browser** drop-down list or click the **Browse** button  and [specify another browser](#) in the **Web Browser** dialog box that opens.
- To have the debugging engine stop as soon as connection between it and PhpStorm is established (instead of running automatically until the first breakpoint is reached), select **Stop at first line** check box.

To define a debug server configuration

- Open the [Servers](#) dialog box by doing one of the following:
 - Choose **File | Settings** for Windows and Linux or **PhpStorm | Preferences** for Mac OS. Then click **Servers** under the **PHP** node.
 - In the [Run/Debug Configuration: PHP Web Application](#) dialog box, click the **Browse** button  next to the **Server** drop-down list.
- Specify the server configuration name.
- Specify the host where the application is run and the port to access it.
- From the **Debugger** drop-down list, choose the [debugging engine](#) to use.
- Specify how the PhpStorm will set up a correspondence between files on the server and their local copies. Based on these mappings, PhpStorm will open local copies of currently processed files.
 - To have PhpStorm suggest mappings itself, clear the **Use path mappings** check box. When you start a debugging session, PhpStorm will try to detect the local copies of the application files on the server. The suggestions are displayed in a dialog box where PhpStorm asks you to confirm or edit suggested mappings.

Tip

PhpStorm detects mappings 100% correctly if the application is deployed to the server root and the folder structure on the server is identical with the folder structure under the project root.

- To specify correspondence between files on the server and their local copies manually, select the **Use path mappings** check box and map files and folders on the server to their local copies.

See Also

Procedures:

- [PHP Debugging Session](#)
- [Creating and Editing Run/Debug Configurations](#)
- [Debugging PHP Applications](#)
- [Debugging](#)

Reference:

- [Run/Debug Configuration: PHP Web Application](#)
- [Servers](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi> 
- <http://youtrack.jetbrains.com/issues/WI> 

Multiuser Debugging Via XDebug Proxies

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

When using Xdebug, you can debug PHP applications in the multiuser mode via [Xdebug proxy](#)  servers.

To enable multiuser debugging via an XDebug proxy server

- Open the active `php.ini` file, locate the `xdebug.idekey` setting and assign it a value of your choice.
- Configure access to the required Xdebug proxy server from your IDE:
 - To register your IDE to the server choose **Tools | Xdebug Proxy | Register IDE** on the main menu. In the [Xdebug Proxy](#) dialog box, that opens, specify the `xdebug.idekey`, the host on which the Xdebug proxy server resides, and the port which PhpStorm will listen to during a proxy debugging session.

When you click **OK**, PhpStorm connects to the specified proxy server and sends the specified credentials. The server registers the credentials and sends a confirmation.

- To update the credentials, choose **Tools | Xdebug Proxy | Configuration** on the main menu. In the [Xdebug Proxy](#) dialog box, that opens, edit the IDE key, the host, and the port settings.
- To discard the current credentials, choose **Tools | Xdebug Proxy | Cancel IDE Registration** on the main menu.

See Also

Procedures:

- [Configuring XDebug](#)

Reference:

- [XDebug Proxy](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

3.0.0.+

Debugging a PHP HTTP Request

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Besides [debugging the entire application](#), you can debug separate HTTP request. This helpful when you are actually interested in a specific page that is accessed in a number of steps, but for this or that reason you cannot specify this page as the start page for debugging, for example, because you need to "come" to this page with certain data.

Debugging PHP HTTP requests in PhpStorm is supported through the [PHP HTTP Request](#) run configuration.

To debug a specific php http request

1. [Define a debug configuration](#) of the type **PHP HTTP Request**. Based on the configuration settings, PhpStorm composes the request to run.
2. In the file, that implements the page in question, [set the breakpoints](#), where necessary.
3. Click the **Debug** button  on the toolbar.
4. [Examine the application](#) as soon as the debugger suspends on reaching the first breakpoint.
5. To control the program execution manually, [step through the code](#) using menu commands or toolbar buttons.
6. To have the program run automatically up to the next breakpoint, [resume](#) the session.

To create a php http request debug configuration

PhpStorm agglutinates the settings specified in this configuration into a PHP HTTP request.

1. Open the [Run/Debug Configuration](#) dialog box by doing one of the following:
 - On the main menu, choose **Run | Edit Configurations**.
 - Press **Alt+Shift+F10** **Shift F10**, then press **00** to display the **Edit Configuration** dialog box or select the configuration from the pop-up window and press **F4F4**.
2. Click  on the toolbar or press **InsertInsert**. From the drop-down list, select the **PHP HTTP Request** configuration type. The [PHP HTTP Request](#) dialog box opens.
3. Specify the configuration name.
4. In the **Server** drop-down list, specify the debug server configuration to interact with the Web server where the application is executed. Select one of the existing configurations or click the **Browse** button  and [define a debug server configuration](#) in the [Servers](#) dialog box that opens.
5. In the **URL** text box, complete the `host` element of the request to debug. Type the path relative to the host specified in the debug server configuration. As you type, PhpStorm composes the URL address on-the-fly and displays it below the text box.
6. Specify whether you want to bring any data to the target page. From the **Request method** drop-down list, choose the relevant request type:
 - To access the page without bringing any data, choose **GET**.
 - To access the page with some data saved in variables, choose **POST** and type the relevant variables in the **Request body** text box.
7. In the **Query** text box, type the query string of the request. This string will be appended to the request after the `?` symbol.
8. Click **OK**, when ready.

See Also

Procedures:

- [Debugging](#)

Reference:

- [Run/Debug Configuration: PHP HTTP Request](#)
- [Debug Tool Window](#)

PHP Support:

- [PHP Debugging Session](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

3.0+

Profiling the Performance of a PHP Application

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Besides pure debugging, you can also profile the performance of your applications. PhpStorm provides visual representation of the profiling data collected by the debugging engine you are currently using.

In this section:

- [Profiling with XDebug](#)
 - [Enabling Profiling with XDebug](#)
 - [Analyzing XDebug Profiling Data](#)
- [Profiling with Zend Debugger](#)
 - [Enabling Profiling with Zend Debugger](#)
 - [Analyzing Zend Debugger Profiling Data](#)

See Also

Procedures:

- [Debugging PHP Applications](#)
- [Configuring a Debugging Engine](#)
- [PHP Debugging Session](#)

Reference:

- [Debug Tool Window](#)
- [Debug](#)
- [Run/Debug Configurations](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
 - <http://youtrack.jetbrains.net/issues/WI> 
-

Profiling with XDebug

Previous | [Next](#) | [See Also](#) | [Comments](#)

Besides [interactive debugging](#), PhpStorm integration with [XDebug](#)  also supports profiling. PhpStorm provides visual representation of profiling data generated by XDebug. You can select several snapshots at a time and collect the aggregated profiling information.

Note

Before profiling with XDebug, download, install and configure the components of the [PHP development environment](#). Normally, these are a PHP engine, a web server, and the XDebug tool.

In this part:

- [Enabling Profiling with XDebug](#)
- [Analyzing XDebug Profiling Data](#)

See Also

Procedures:

- [Debugging PHP Applications](#)
- [Profiling with Zend Debugger](#)
- [Configuring XDebug](#)
- [Configuring a Debugging Engine](#)
- [PHP Debugging Session](#)

Reference:

- [Debug Tool Window](#)
- [Debug](#)
- [Run/Debug Configurations](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
 - <http://youtrack.jetbrains.net/issues/WI> 
-

Enabling Profiling with XDebug

Previous | [Next](#) | [See Also](#) | [Comments](#)

XDebug profiler is incorporated in the XDebug tool. Therefore you first need to download, install, and enable XDebug itself and after that enable the profiling functionality within it.

To enable profiling with XDebug, perform these general steps:

- [Configure the XDebug tool](#)
- [Enable the XDebug profiler](#)
- [Enable toggling the profiler from the browser](#)
- [Specify the location for storing accumulated profiling data](#)

To configure XDebug

1. [Download and install](#) the XDebug tool.
2. [Integrate XDebug with the PHP engine](#).
3. [Integrate XDebug with PhpStorm](#).

To enable XDebug profiler

1. Open the [active php.ini file](#).
2. Set the [xdebug.profiler_enable](#)  directive to 1:

```
xdebug.profiler_enable = 1;
```

3. To enable toggling the profiler from the browser through control over debugger cookies, set the [xdebug.profiler_enable_trigger](#) directive to 1:

```
xdebug.profiler_enable_trigger = 1;
```

To enable toggling the profiler from the browser

- [Specify the GET/POST or COOKIE parameters](#). Do one of the following:
 - [Specify the values manually](#).
 - [Generate bookmarklets](#) through which you will start/stop a debugging session by controlling the debugger cookie.

To specify the location for storing accumulated profiling data

1. Open the [active php.ini file](#).
2. Define location for accumulating profiling snapshots by specifying the [xdebug.trace_output_dir](#) directive.

```
xdebug.trace_output_dir = "<output folder name>"
```

3. Specify the name of the file to store snapshots in through the value of the `xdebug.trace_output_name` directive. The default name is `cachegrind.out.%p`, where `%p` is the name format specifier. Accept the default name or define a custom one in compliance with the following standard:
 1. The name should always be `cachegrind.out`.
 2. Use the [supported format specifiers](#).

See Also

Procedures:

- [Enabling Profiling with Zend Debugger](#)
- [Analyzing XDebug Profiling Data](#)

External Links:

- <http://xdebug.org/docs/profiler#starting>

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Analyzing XDebug Profiling Data

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

When integration with [XDebug profiler is enabled](#), PhpStorm provides visual representation of profiler snapshots. PhpStorm opens a separate editor tab with four views where the data are presented based on different criteria.

To have the profiling data collected and analyze it, perform these general steps:

- [Initiate an XDebug debugging session](#)
- [Retrieve the data accumulated by the profiler](#)
- [Examine the profiling data](#)

To initiate a debugging session, do one of the following

- To start debugging an entire application [with a configuration](#), create [debug configuration](#) of the type **PHP Web Application**, and [launch debugging](#) by clicking the **Debug** toolbar button .
- To debug a specific PHP HTTP request, [define a debug configuration](#) of the type **PHP HTTP Request**, and [launch debugging](#) by clicking the **Debug** toolbar button .
- [Enable control over the debugger from the browser](#).
- Generate [bookmarklets](#) to toggle the debugger through.
- Toggle the **Start Listen PHP Debug Connections** button  so it changes its color to green . After that PhpStorm starts listening to the port of the [debugging engine used in the current project](#). Ports for debuggers are set at the PhpStorm level in the [Debug](#) dialog box (**File** | **Settings** | **PHP** | **Debug** for Windows and Linux, PhpStorm | **Preferences** | **PHP** | **Debug** for MacOS).
- Open the starting page of your application in the browser.
- To activate XDebug from the browser, choose the XDebug [Start Profiler](#) bookmark.
- Refresh the page.
- Return to PhpStorm and continue the session.

To retrieve the collected profiling data

1. On the main menu, choose **Tools** | **Analyze XDebug Profiler Snapshot**.
2. In the **Select XDebug profiler snapshot** dialog box, that opens, choose the [folder and the file where the profiling data is stored](#).

PhpStorm presents the collected profiling data in a separate editor tab with the name of the selected profiler output file.

To view and examine the profiling data, perform these general steps

When you request on the accumulated profiling data, PhpStorm opens its visualized presentation in a separate editor tab. The tab is named after the selected [profiler output file](#) and consists of several views. Switch between the views to analyze the profiling data based on various criteria of analysis.

- In the **Execution Statistics** view, examine the summary information about execution metrics of every called function.
- In the **Call Tree** view, explore the execution paths of all called functions.
- To explore the execution paths of a specific function, select the function in question in the **Call Tree** view and view its callees in the **Callees** view.
- To explore all the paths that can result in calling a specific function, select the function in question in the **Call Tree** view and examine its possible callers in the **Callers** view.

See Also

Procedures:

- [PHP Debugging Session](#)
- [Enabling Profiling with XDebug](#)
- [Configuring XDebug](#)
- [Configuring PHP Development Environment](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Profiling with Zend Debugger

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Besides [interactive debugging](#), PhpStorm integration with [XDebug](#)  also supports profiling. PhpStorm provides visual representation of profiling data generated by Zend Debugger.

Note

Before profiling with Zend Debugger, download, install and configure the components of the [PHP development environment](#). Normally, these are a PHP engine, a web server, and the Zend Debugger tool.

See Also

Procedures:

- [Debugging PHP Applications](#)
- [Profiling with XDebug](#)
- [Configuring a Debugging Engine](#)
- [PHP Debugging Session](#)

Reference:

- [Debug Tool Window](#)
- [Debug](#)
- [Run/Debug Configurations](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Enabling Profiling with Zend Debugger

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

XDebug profiler is incorporated in the Zend Debugger tool. Therefore you only need to download, install, and enable XDebug itself.

To enable profiling with Zend Debugger, perform these general steps:

- [Configure Zend Debugger](#)
- [Enable toggling the profiler from the browser](#)

To configure zend debugger

1. [Download and install](#) the Zend Debugger tool.
2. [Integrate Zend Debugger with the PHP engine](#).
3. [Integrate XDebug with PhpStorm](#).

To enable toggling the profiler from the browser

- [Specify the GET/POST or COOKIE parameters](#). Do one of the following:
 - [Generate bookmarklets](#) through which you will start/stop a debugging session by controlling the debugger cookie.
 - Specify the values manually.

See Also

Procedures:

- [Enabling Profiling with XDebug](#)
- [Analyzing Zend Debugger Profiling Data](#)

External Links:

- <http://confluence.jetbrains.net/display/WI/Zend+Debugger+installation+guide>

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Analyzing Zend Debugger Profiling Data

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

When integration with the [Zend Debugger profiler is enabled](#), PhpStorm provides visual representation of profiler snapshots. PhpStorm opens a separate editor tab with four views where the data are presented based on different criteria.

With Zend Debugger, profiling is supported within a [zero configuration](#) debugging session.

To have the profiling data collected and analyze it, perform these general steps:

- [Initiate a zero configuration Zend Debugger session](#)
- [Examine the profiling data](#)

To initiate a debugging session

1. [Enable control over the debugger from the browser](#).
2. Generate [bookmarklets](#) to toggle the debugger through.
3. Toggle the **Start Listen PHP Debug Connections** button  so it changes its color to green . After that PhpStorm starts listening to the port of the [debugging engine used in the current project](#). Ports for debuggers are set at the PhpStorm level in the [Debug](#) dialog box (**File | Settings | PHP | Debug** for Windows and Linux, **PhpStorm | Preferences | PHP | Debug** for MacOS).
4. Open the starting page of your application in the browser.
5. To activate Zend Debugger from the browser, choose the **Zend Start Profiler** [bookmark](#).
6. Refresh the page.
7. Return to PhpStorm.
8. In the dialog box, that opens, select the incoming connection to profile and click **Accept**.

Note

The **Incoming Connection from Zend Debugger** dialog box appears only once, when you accept connection from this host for the first time.

PhpStorm presents the collected profiling data in a separate editor tab with the name of the selected profiler output file.

To view and examine the profiling data, perform these general steps

When you request on the accumulated profiling data, PhpStorm opens its visualized presentation in a separate editor tab. The tab is named after the file that implements the page you are currently profiling and consists of several views. Switch between the views to analyze the profiling data based on various criteria of analysis.

- In the **Execution Statistics** view, examine the summary information about execution metrics of every called function.
- In the **Call Tree** view, explore the execution paths of all called functions.
- To explore the execution paths of a specific function, select the function in question in the **Call Tree** view and view its callees in the **Callees** view.
- To explore all the paths that can result in calling a specific function, select the function in question in the **Call Tree** view and examine its possible callers in the **Callers** view.

See Also

Procedures:

- [PHP Debugging Session](#)
- [Enabling Profiling with Zend Debugger](#)
- [Configuring Zend Debugger](#)
- [Configuring PHP Development Environment](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Testing PHP Applications

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

The topics in this part provide guidelines in PHP-specific unit testing procedures. For general information on testing in PhpStorm, see the section [Testing](#).

PhpStorm supports unit testing of PHP applications through integration with the [PHPUnit](#) tool. The installation procedure depends on the operating system you use and your system settings. Please, refer to the [PHPUnit installation instructions](#) for information on installing and configuring this tool.

Generally, PhpStorm runs and debugs PHPUnit tests in the same way as other applications, by running the run/debug configurations [you have created](#). When doing so, it passes the specified test class, file, or directory to the test runner. You can run unit testing locally and remotely depending on the chosen run configuration.

To create and run unit tests on php applications, perform the following general steps:

- [Enable PHPUnit support](#).
- [Generate unit tests](#).

- [Create a run configuration](#):
 - To run unit tests locally, create a [PHPUnit](#) configuration.
 - To run unit tests on a remote server, create a [PHPUnit on Server](#) configuration.
- [Launch unit tests](#) and [monitor test results](#) in the [Run](#) tool window.

To enable PHPUnit support

1. Open the [Project Settings](#).
2. On the [PHP](#) page, specify the location of the PHP folder in the [PHP Path](#) text box.
3. On the [Directories](#) page, add the PEAR folder as a [new content root](#).

See Also

Procedures:

- [Configuring Project Settings](#)
- [Configuring Content Roots](#)
- [PHP-Specific Guidelines](#)
- [Testing](#)

Reference:

- [PHP](#)
- [Directories](#)
- [Run/Debug Configuration: PHPUnit](#)
- [Run/Debug Configuration: PHPUnit on Server](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

3.0.0.+

Enabling PHPUnit Support

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm supports unit testing of PHP applications through integration with the [PHPUnit](#) tool. The installation procedure depends on the operating system you use and your system settings. Please, refer to the [PHPUnit installation instructions](#) for information on installing and configuring this tool.

To enable PHPUnit support

1. Open the [Project Settings](#), and click [PHP](#). The [PHP](#) page opens.
2. From the [Interpreter](#) drop-down list, choose the [PHP installation](#) to use.
3. Configure the [PEAR](#) folder as an [include path](#). Click the [Add](#) button in the [Include path](#) area, and select the [PEAR](#) folder in the dialog box, that opens.

Tip

Alternatively, open the the [Directories](#) page and add the [PEAR](#) folder as a [new content root](#)

See Also

Reference:

- [PHP](#)
- [Directories](#)
- [Run/Debug Configuration: PHPUnit](#)
- [Run/Debug Configuration: PHPUnit on Server](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

3.0.0.+

Generating PHPUnit Test Class

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Besides executing unit tests, PhpStorm can generate PHPUnit test classes based on the PHP classes or files that are subject for testing.

To generate a PHPUnit test class

1. In the [Project](#) tool window, select the PHP class to create unit tests for.
2. On the context menu of the selection, choose [Generate PHPUnit Test](#).
3. In the [Generate PHPUnit Test from <selected_class_name>](#) dialog box that opens, specify the name of the test class to be created and the directory to store it in.

See Also

Procedures:

- [Testing](#)

Reference:

- [Generate PHPUnit Test Dialog](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

PHP-Specific Command Line Tools

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

With PhpStorm, you can [use PHP from the command line](#) through integration with PHP-specific command line tools.

PhpStorm supports integration with third-party tools [Zend Framework](#) and [Symfony](#). Alternatively, you can create your own tool and populate it with commands that fit your preferences and needs. The available tools are listed in the [Command Line Tool Support](#) page of the [Settings](#) dialog box.

Warning

Integration with Symfony version 1.1 or higher is supported.

To work with a php command line tool, perform these general steps:

- [Download and enable the required third party tool](#) or [define a custom tool](#).
- [Customize the tool commands](#), if necessary, and [have the tool definition file validated](#).
- [Invoke commands](#) and analyze the results of their execution in the dedicated [Command Line Tools Console](#) tool window.

See Also

Procedures:

- [Using Command Line Tools](#)
- [PHP-Specific Guidelines](#)

Reference:

- [Command Line Tool Support](#)
- [PHP](#)
- [Command Line Tools Console Tool Window](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

2.1+

Enabling a Command Line Tool

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

The PHP command line tools that are available from PhpStorm are listed in the [Command Line Tool Support](#) page of the [Settings](#) dialog box. Once you have enabled integration with a tool, you can deactivate it at any time and activate it again when necessary.

You can download and [activate an existing third-party tool](#) or [create your own tool](#). You can [update the command definitions](#) of third-party or a custom tool right in the PhpStorm editor.

To enable integration with a third party php command line tool

At the PhpStorm level, integration with command line tools is supported via the **Command Line Tool** bundled plugin. The plugin is by default enabled. If not, enable it in the [Plugin Configuration Wizard](#) or the [Plugins](#) page of the [Settings](#) dialog box.

1. Download the desired tool.

Note

Currently PhpStorm supports integration with [Zend Framework](#) and [Symfony](#) version 1.1 or higher.

2. [Open the project settings](#) and click the **Command Line Tool Support**.
3. In the [Command Line Tool Support](#) page, click the **Add** button. In the **Choose Framework to Add** dialog box that opens, choose **Zend Framework** or **Symfony**.
4. In the dialog box that opens, specify the path to the definition file of the chosen tool in the **Path to <tool>** text box. Depending on the chosen tool, specify the location of the following file:
 - o `zf.bat` for **Zend Framework**.
 - o `symfony` for **Symfony 1.***.
 - o `app/console` for **Symfony 2**.

file. PhpStorm searches the contents of the specified file for definitions of commands. When the file analyzes is completed, PhpStorm displays the specified file in the list.
5. In the **Alias** text box, specify the alias to use in calls of framework commands. Accept the default alias or edit it, if necessary.
6. To activate the detected command set, select the **Enable** check box.

7. [Customize the command set](#), if necessary.
8. In the **Show console in** area, specify where you want the **Input** pane for [typing commands](#) opened:
 - o To have the **Input** pane opened in a pop-up window, choose the **Pop-up** option.
 - o To have the **Input** pane opened as a text box at the bottom of the [Command Line Tools Console](#) tool window, choose the **Tool window** option.

To create a custom command line tool

1. [Open the project settings](#) and click the **Command Line Tool Support**.
2. In the [Command Line Tool Support](#) page, click the **Add** button. In the **Choose Framework to Add** dialog box that opens, choose **Custom Framework**.
3. In the [Framework Settings](#) dialog box that opens, specify the following:
 1. Tool name and location.
 2. The alias to use in command calls instead of the full path to the tool.
 3. Brief explanation of the tool functionality.
4. When you click **OK**, PhpStorm brings you to the [Command Line Tool Support](#) page, where the new tool is added to the list of available frameworks.
5. Select the newly created tool and click the **Open definition in editor** button. In the tool definition `.xml` file that opens in the editor, define the tool commands.
6. In the **Show console in** area, specify where you want the **Input** pane for [typing commands](#) opened:
 - o To have the **Input** pane opened in a pop-up window, choose the **Pop-up** option.
 - o To have the **Input** pane opened as a text box at the bottom of the [Command Line Tools Console](#) tool window, choose the **Tool window** option.

See Also

Procedures:

- [Using Command Line Tools](#)
- [Running Command Line Tool Commands](#)
- [PHP-Specific Guidelines](#)

Reference:

- [Command Line Tool Support](#)
- [Command Line Tools Console Tool Window](#)
- [PHP](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

2.1+

Using Phing

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm supports integration with the [Phing](#) tool for flexible building of PHP projects. Phing functionality in PhpStorm is provided through a dedicated [Phing Build](#) tool window.

In this section:

- [Enabling Phing support](#)
- [Accessing the Phing Build tool window](#)
- [Managing lists of build files](#)
- [Examining build targets defined in build files](#)
- [Running builds](#)
- [Running separate build targets](#)
- [Appointing targets to be executed before running or debugging according to specific configurations](#)

Warning

Using the integration requires that you have [downloaded](#) and [set up](#) Phing on your computer. If you are using an [AMP](#), the package may already contain Phing.

To enable Phing integration support

At the PhpStorm level, Phing integration is supported via the **Phing** bundled plugin. The plugin is by default enabled. If not, enable it in the [Plugin Configuration Wizard](#) or the [Plugins](#) page of the [Settings](#) dialog box.

1. Open a Phing build file in the editor or select it in the [Project](#) tool window.

Tip

A correct Phing build file is an `.xml` file with the root element `<project>`.

2. On the context menu of the editor or the selection, choose **Add as Phing build file**.
3. In the [Phing Build](#) tool window, that opens, click the **Properties** button  on the toolbar.

Note

The button is only available if the list of build files is not empty.

4. In the **Phing Properties** dialog box, that opens, specify the location of the `phing.bat` file.

After that the **Phing Build** tool window is available from the **View | Tool Windows | Phing Build** menu within the scope of the current project.

To open the Phing build tool window, do one of the following

- If Phing support is not [enabled in the project](#), open a Phing build file in the editor or select it in the [Project](#) tool window, and then choose **Add as Phing build file** on the context menu of the selection..
- If Phing is supported at the project level, choose **View | Tool Windows | Phing Build** on the main menu.

To configure a list of build files, perform these general steps

- [Open the Phing Build](#) tool window.
- To add a build file to the list, click the **Add** button  on the toolbar and choose the required `.xml` build file in the **Select Phing Build File** dialog box, that opens.

Note

1. Build files are created outside the **Phing Build** tool window. Consider details in [Getting Started](#) with writing Phing build files.
 2. Normally, Phing build files are stored under your PHP project root folder.
- To remove a build file from the list, select the file and click the **Remove** button  on the toolbar.
 - To navigate to the source code of a build file, select the desired file and choose **Jump to Source** on the context menu of the selection.

To examine build targets

- [Open the Phing Build](#) tool window.
- To view the build targets defined in a specific build file, expand the corresponding build file node.
- To have build targets defined in all build files displayed or hidden, click the **Expand All**  or **Collapse All**  toolbar buttons respectively.
- To navigate to the definition of a target in the source code, select the desired target and choose **Jump to Source** on the context menu of the selection.

To run a build file

1. [Open the Phing Build](#) tool window.
2. Select the required build file in the list and click the **Run** button  on the toolbar or choose **Run Build** on the context menu of the selection.

To run a build target

1. [Open the Phing Build](#) tool window.
2. Select the required build target in the list and click the **Run** button  on the toolbar or choose **Run Target** on the context menu of the selection.

To have a target always executed before running or debugging the project according to a specific configuration

1. [Open the Phing Build](#) tool window.
2. Select the desired target and choose **Before Run/Debug** on the context menu of the selection.
3. In the **Execute Target Before Run/Debug** dialog box that opens, select the [configurations](#) before which you want the target executed.

See Also

Procedures:

- [PHP-Specific Guidelines](#)

Reference:

- [Phing Build Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

3.0.0.+

UML

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

UML model in PhpStorm is represented by a Class diagram in standard notation.

PhpStorm enables using UML class diagrams to analyze PHP applications, and the structure of the databases and tables. Besides that, you can explore changes committed to VCS.

In PhpStorm, Class diagram features:

- Possibility to view UML model as a diagram in a [separate editor tab, in a pop-up window](#), or as a [preview](#).
- Possibility to invoke UML class diagram from the **Project, Structure, Data Sources, Changes** tool windows, the **History** tab of the **Version Control** tool window, and **Navigation** bar. In the editor, one can view class diagram for the whole class, or for symbols at caret.
- Navigation from a diagram element to the [underlying source code](#).
- Highlighting [siblings and children classes](#).
- Refactorings ([Rename](#), [Move](#), [Safe Delete](#), [Extract Class](#), [Inline](#)).
- Navigation to [class, file or symbol by name](#) and to the [last edit location](#).
- Viewing class information at the tooltip, and [quick documentation lookup](#).
- [Viewing changed classes](#) as a UML Class diagram.

- Quick hierarchy view in a [UML Class diagram pop-up window](#).
- [Viewing subtypes and super classes](#).
- Possibility to [find usages](#) of a node element or member.

See Also

Procedures:

- [Refactoring Source Code](#)
- [Viewing Structure and Hierarchy of the Source Code](#)

Reference:

- [UML Class Diagram Toolbar and Context Menu](#)

External Links:

- <http://www.uml.org/#UML2.0>

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

```
ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js'; var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s); });
```



PhpStorm 3.0.0 Web Help

Opening UML Class Diagram

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

You can invoke UML class diagram from different places:

- From the various tool windows.
- From the Navigation bar.
- From the editor. In this case, PhpStorm shows relevant class diagram for the whole class, or for a symbol at caret.

PhpStorm displays UML diagrams in two modes:

- In a pop-up window.
- In a separate editor tab.

To open a UML class diagram

1. Select the desired item in a tool window, in the Navigation bar, or open it in the editor.
2. Do one of the following:
 - On the context menu of the selection, click **UML**, and on the submenu, select the way you want to view the model: **Show Class Diagram** or **Show UML Pop-up**.
 - Press **Ctrl+Shift+Alt+UCommand Shift Alt U** or **Ctrl+Alt+UCommand Alt U**.

Tip

You can open UML class diagram without using your pointing device. Consider such workflow: press **Alt+HomeAlt Home**, then press **Ctrl+Alt+UCommand Alt U**.

See Also

Procedures:

- [UML](#)

Reference:

- [Diagram Toolbar and Context Menu](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Adding Node Elements to Diagram

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

You can add node elements to the background of the UML Class diagram, using the context menu action, or drag-and-drop technique.

In this section:

- [Adding node elements](#)
- [Dragging node elements](#)
- [Adding notes](#)

To add a node element to a UML class diagram

1. In the **Enter class name to add** dialog box, start typing the desired class name. PhpStorm automatically displays suggestion list with matching names.

Note that you can include classes outside the scope of your project, by selecting the **Include non-project classes** check box.

2. Select the desired class from the suggestion list, and press **EnterEnter**.

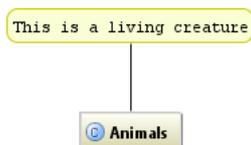
If there are relationships between the node elements in model, and the added element, these relationships will be displayed in diagram.

To add a node element using drag-and-drop technique

1. Select one or more desired elements in the Project tool window.
2. Drag selection to the diagram background.

To add a note to a node element

1. Right-click a node element, which you would like to comment.
2. On the context menu, choose **New | Note**.
3. In the text box, type the desired text. Note that **EnterEnter** starts a new line in the text box. To complete a note, click **OK**, or press **Ctrl+EnterCommand Enter**.



See Also

Language and Framework-Specific Guidelines:

- [UML](#)

Reference:

- [Diagram Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Creating Node Elements and Members

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

You can populate your model with [node elements](#) and delete them from diagram, or from your project. When a node element is created in a UML Class diagram, the corresponding stub file is created in the current package.

In a similar way, PhpStorm helps create [members](#) in the node elements.

It is also possible to clarify the contents of diagrams with notes, which are created same way.

To create a node element in a UML class diagram

1. Do one of the following:
 - Right-click the diagram background. Next, on the context menu, point to **New**, and choose element type from the submenu.
 - Press **Alt+InsertCommand N**, and choose element type from the pop-up frame.
2. Specify the element name, or type the contents of a note, and click **OK**.

To create a member in a node element

1. [Select](#) a node element in a diagram.
2. Do one of the following:
 - On the context menu of the selection, point to **New**, and then choose the member type on the submenu.
 - Press **Alt+InsertCommand N**, and choose member type from the pop-up frame.
3. Depending on the selected member type ([field.constant](#), or [method](#)), specify the member name and the other parameters. See preliminary results in the **Preview** pane of each dialog.

See Also

Procedures:

- [Creating Files from Templates](#)

Language and Framework-Specific Guidelines:

- [UML](#)

Reference:

- [UML Class Diagram Toolbar and Context Menu](#)

External Links:

- <http://www.uml.org/#UML2.0>

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Creating Relationship Links Between Elements

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

For creating relationship links between node elements, PhpStorm provides a special mode, in which drag-and-drop technique can be used for drawing links. As a link is created, the corresponding `extends` / `implements` clause is generated in the underlying source code. So doing, the `extends` link is represented with a solid line, and the `implements` link is represented with a dotted line.

To create a link between node elements

1. Make sure that the  button is pressed on the diagram toolbar.
2. Draw a link from the source to a target node.
3. Specify whether the target class should be declared abstract, or implement required abstract methods:

To delete a link

1. Select a link between two node.
2. Press `DeleteDelete`.
3. In the dialog box that opens, confirm deletion:

See Also

Reference:

- [Diagram Toolbar and Context Menu](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Deleting Node Elements from Diagram

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

To remove elements from view

1. In the diagram, select one or more elements to be deleted.
2. Press the `DeleteDelete`.

See Also

Procedures:

- [UML](#)
- [Safe Delete](#)

Reference:

- [Diagram Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Navigating to Source from a UML Class Diagram

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

In a UML Class diagram, you can use all the regular procedures that enable navigating to the source code of a class:

- Jump to source (`F4F4`). So doing, if the `Jump to Source` command has been invoked on a class node, the caret is placed at the class declaration. If the command has been invoked on a field or method, the caret is placed before the corresponding field or method declaration in the source code.
- View source (`Ctrl+EnterCommand Enter`)
- Navigate by name (`Ctrl+NCommand N`, `Ctrl+Shift+NCommand Shift N`, or `Ctrl+Shift+Alt+NCommand Shift Alt N`)

See Also

Procedures:

- [UML](#)
- [Navigating to Class, File or Symbol by Name](#)

Reference:

- [Diagram Toolbar and Context Menu](#)

External Links:

- <http://www.omg.org>

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Viewing Changes on UML Class Diagram

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

For the modules under version control, you can usually view the list of changed files in the Changes tool window. If you want to evaluate how your changes affect the model, use UML Class diagram. This view presents the complete picture of changes, including relationships between the modified classes.

For this purpose, PhpStorm provides the action [Show changed classes](#), which is available from the editor, Project tool window, and the Local tab of the Changes tool window.

Moreover, PhpStorm helps you analyze how the changes affect a model across revisions, and provides the action [Compare all classes from revision on UML](#).

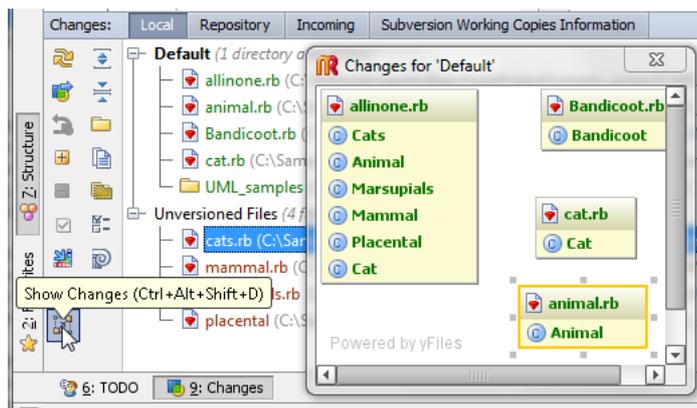
Changes in Class diagram are color-coded:

- Green for added elements.
- Gray for deleted elements.
- Blue for modified elements.

To view changes in UML class diagram

1. In the [Local](#) tab of the Changes tool window, select the desired changelist.
2. In the toolbar of the [Local](#) tab, click or press `Ctrl+Shift+Alt+D` / `Command Shift Alt D`.

Alternatively, on the context menu of the editor or the Project tool window, choose **UML | Show Changes**, or press `Ctrl+Shift+Alt+D` / `Command Shift Alt D`.



The diagram opens in a pop-up window. Double-click a node to view changes in a Differences viewer.

To view changes in revisions as UML class diagram

1. In the [History tab](#) of the Version Control tool window, select the desired revision.
2. Click or press `Ctrl+Shift+D` / `Command Shift D`. The diagram opens in a pop-up window.

See Also

Procedures:

- [Version Control with PhpStorm](#)
- [UML](#)

Reference:

- [Diagram Reference](#)
- [Local Tab](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

```
ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js'; var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s);})();
```



PhpStorm 3.0.0 Web Help

Viewing Class Hierarchy in a Pop-up Window

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: 

PhpStorm helps you view the complete hierarchy of the selected type. So doing, you can modify the contents of the UML Class diagram pop-up window, to show ancestor or descendant classes, types used in method signatures, and perform all actions that are available for UML Class diagrams.

To view class hierarchy as a UML class diagram pop-up window

1. Open the desired class in the editor, and place the caret at the type you want to see hierarchy for.
2. Do one of the following:
 - On the context menu, point to **Diagrams**, and then choose **Show Diagram**, or **Show Diagram Popup**
 - Press `Ctrl+Shift+Alt+U` or `Command Shift Alt U` or `Ctrl+Alt+U` or `Command Alt U`

Tip

You can view type hierarchy for any class selected in the Project tool window:

See Also

Procedures:

- [UML](#)

Reference:

- [Diagram Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Viewing Siblings and Children

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm can smartly distinguish nodes that belong to the same namespace from extraneous ones.

To highlight sibling nodes

1. [Select](#) a node element in diagram.
2. Start navigating through the diagram. So doing, extraneous classes are automatically grayed out.

See Also

Procedures:

- [UML](#)

Reference:

- [Diagram Toolbar and Context Menu](#)

External Links:

- <http://www.omg.org>

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Viewing Ancestors, Descendants, and Usages

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: 

- [To show ancestor and descendant types](#)
- [To find usages of a diagram element](#)

To show ancestor and descendant types

1. [Open UML Class diagram](#)
2. [Select](#) a node element in diagram.
3. On the context menu, choose one of the following commands, or press keyboard shortcuts:
 - **Go to | Implementation** `Ctrl+Alt+B` or `Command Alt B`
 - **Show Parents** `Ctrl+Alt+P` or `Command Alt P`
4. If there are several classes available, select the desired one from the pop-up list:

The selected class will be added to the UML Class diagram, with the corresponding relationship links:

To find usages of a diagram element

1. [Select](#) a diagram element.
2. On the context menu of the selection, select the desired [Find Usages](#) command.

See Also

Procedures:

- [Finding Usages](#)

Reference:

- [Diagram Reference](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

JavaScript-Specific Guidelines

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

PhpStorm enables creating rich Internet applications and Web applications by providing elaborate support of [JavaScript](#) and tight integration with [AJAX](#) and other adjacent frameworks and technologies.

JavaScript files are marked with  icon.

- [JavaScript support](#)
- [Developing an application that contains JavaScript](#)

JavaScript support

JavaScript support in PhpStorm includes:

- Full coding assistance:
 - Smart, DOM-based, browser-type aware JavaScript [code completion](#) for:
 - Keywords, labels, variables, parameters and functions.
 - User defined and built-in JavaScript functions.
 - JavaScript namespaces.
 - Error and syntax highlighting.
 - Code [formatting](#) and [folding](#).
 - Numerous code [inspections](#) and [quick-fixes](#).
 - Initial support for [ECMAScript](#) level 4 draft standard (mapped to js2/as extensions), including E4X syntax.
- [Code Generation](#)
 - Generating code stubs based on [file templates](#) during file creation.
 - Inserting, expanding, and generating JavaScript code blocks using [live templates](#).
 - Creating various applications elements via JavaScript and AJAX [intention actions](#).
 - Possibility to create [line and block comments](#) (Ctrl+Slash Command Slash or Ctrl+Divide Command Divide/Ctrl+Shift+Slash Command Shift Slash or Ctrl+Shift+Divide Command Shift Divide).
 - [Unwrapping and removing statements](#).
- Refactoring

PhpStorm provides both common refactoring types available for all the supported languages and JavaScript-specific refactoring.

- [Common refactorings](#):
 - [Rename](#) a file, function, variable, parameter, or label (both directly and via references).
 - [Move/Copy](#) a file.
 - [Safe Delete](#) a file.
 - [Extract](#) function.
- [JavaScript-Specific Introduce Parameter Refactoring](#)
- Numerous ways to [navigate](#) through the source code, among them:
 - [Navigating with Structure Views](#).
 - Enhanced [navigation with gutter icons](#).
 - Show/Goto Implementation (Ctrl+Alt+BCommand Alt B) from overridden method / subclassed class.
- Advanced facilities for [searching through the source code](#).
- Support of the [JSDoc](#) format and [generating documentation comments](#).
- [Viewing reference](#) information:
 - Definitions, inline documentation, parameter hints.
 - [JSDoc comments](#).
 - Lookup in [external JavaScript libraries](#).
- [Running and debugging](#).
 - Launching applications directly from PhpStorm by opening the starting application HTML page in the [default PhpStorm browser](#).
 - A dedicated [debug](#) configuration for launching debugging sessions directly from PhpStorm.
 - A JavaScript-aware [debugger](#) that lets you execute applications step by step, evaluate expressions, examine related information and find runtime bugs.
 - Support for [JavaScript breakpoints](#).
- Tight integration with related frameworks and technologies: [AJAX](#), [jQuery](#), [YUI](#), [Dojo](#), [Prototype](#), [MooTools](#), [Qooxdoo](#), and [Bindows](#):
 - Code completion for every framework.
 - [Dojo](#) style type annotations to provide more accurate completion and parameter type information.
 - [Quick Documentation lookup](#) for DoJo style commands.
- Support for the [JSON](#) (JavaScript Object Notation) format:
 - A JSON file [type template](#) mapped to json file extension.

- JSON code [formatting](#) and [folding](#).

To develop an application that contains JavaScript

Developing an application that contains JavaScript, generally, includes performing the following steps:

1. Make sure the JavaScript Support plugin is enabled. The plugin is bundled with PhpStorm and activated by default. If it is not, [enable the plugin](#).
2. [Create a project](#) to implement your application.
3. [Download, install, and configure JavaScript frameworks and libraries](#).
4. [Populate the project](#). Use the following PhpStorm facilities, where applicable:
 - [Coding assistance](#).
 - [Code Generation](#).
 - Various types of [navigation](#) and [search](#) through the source code.
 - [Viewing reference](#) and [generating documentation comments](#).
 - [Look-up in external libraries](#).
5. Improve the quality and maintainability of your code using various types of [refactoring](#), both common and JavaScript-specific.
6. Run your application by [opening its starting HTML page in the PhpStorm default browser](#).
7. [Debug](#) your application.

The JavaScript debugging functionality is incorporated in PhpStorm, so just [configure the debugger](#), whereupon you can start the [debugging session](#) and [proceed as usual](#): set the breakpoints step through them, stop and resume the program, and examine it when suspended.

Warning

Debugging for JavaScript applications is supported only in the [Firefox](#) and [Google Chrome](#) browsers.

See Also

Concepts:

- [Supported Languages](#)

Language and Framework-Specific Guidelines:

- [NodeJS](#)

Reference:

- [JavaScript. Libraries](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

2.0+

Configuring JavaScript Libraries

Previous | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

In PhpStorm, you can download and install a number of the most popular JavaScript libraries.

PhpStorm also provides the possibility to [set up custom JavaScript libraries](#) that contain previously downloaded external JavaScript libraries and frameworks. Configured libraries can be used in [code completion](#), highlighting, [navigation](#), and [Documentation Lookup](#).

To enable `Documentation Lookup` for symbols defined in an external library or framework, [provide a link](#) to its documentation. Upon pressing `Shift+F1Shift F1` with the cursor positioned at the symbol in question, PhpStorm invokes this link and opens the documentation page in the browser.

Note

For [jQuery](#), [jQuery UI](#), [Ext JS](#), [Prototype](#), and [Dojo](#) library files, PhpStorm automatically detects links to the library documentation.

Configured JavaScript libraries are available at the IDE level. To attach a library to your project or a part of it, [specify the library scope](#).

To download and install a JavaScript-related library from PhpStorm

1. [Open the Project Settings](#) dialog box and click [JavaScript Libraries](#).
2. In the Libraries area, click the Download button.
3. The `Download Library` dialog box, that opens, shows a list of most popular libraries with indications of their versions and URL addresses they are available through. Select the required library and click the `Download and Install` button. You return to the [JavaScript Libraries](#) page where the new library is added to the list. Click `OK` to save the settings.

To configure a custom JavaScript library

1. [Open the Project Settings](#) dialog box and click [JavaScript Libraries](#).
2. In the Libraries area, click the Add button.
3. In the `New Library` dialog box that opens, specify the library name.
4. [Create a list of files to be included in the library](#).

To add a library file to a JavaScript library

1. In the [New Library/Edit Library](#) dialog box, click the **Attach** button and select the required file or directory in the dialog box that opens. PhpStorm returns to the **New Library** dialog box where the **Name** read-only field shows the name of the selected library file or the names of relevant library files from the selected directory.
2. In the **Type** field, specify which version of library you have downloaded and are going to add.
 - o Choose **Debug** if you are adding a library file with uncompressed code. This version is helpful in the development environment, especially for debugging.
 - o Choose **Release** if you are adding a library file with [minified](#) code. This version is helpful in the production environment because the file size is significantly smaller.
3. Specify the URL addresses to access the documentation for library files.
 - o To add a link to the documentation for a library, select the corresponding library file, click the **Specify** button in the **Documentation URLs** area, and specify the documentation URL in the dialog box that opens.

Tip

For [jQuery](#), [jQuery UI](#), [Ext JS](#), [Prototype](#), and [Dojo](#) library files, PhpStorm automatically detects the link to the library documentation and suggests it in the **Enter documentation URL** text box.

- o To remove a link, select it in the **Documentation URLs** and click the **Remove** button.

To remove a library file

- In the [New Library/Edit Library](#) dialog box, select the required library file and click the **Remove** button.

To update the contents of a library

1. [Open the Project Settings](#) dialog box and click [JavaScript Libraries](#).
2. In the **Libraries** area, select the required library and click the **Edit** button.
3. In the [Edit Library](#) dialog box that opens, [add](#) library files, [remove](#) library files, and [change links to documentation](#) as necessary.

To delete a library

1. [Open the Project Settings](#) dialog box and click [JavaScript Libraries](#).
2. In the **Libraries** area, select the required library and click the **Remove** button.

To specify the scope to use a library in

1. [Open the Project Settings](#) dialog box, and click **Usage Scope** under the **JavaScript Libraries** node.
2. In the [JavaScript Usage Scope](#) dialog box that opens, specify the custom JavaScript libraries to use in files and folders within your project. To appoint a library for a file or folder, select the required item in the **File/Directory** field and choose the relevant library from the **Library** drop-down list.

See Also

Procedures:

- [Viewing Inline Documentation](#)
- [Enabling JavaScript Unit Testing Support](#)

Reference:

- [JavaScript Libraries](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Creating Documentation Comments

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

PhpStorm creates stubs of [JSDoc](#) comments on typing the opening tag `/**` and pressing `Enter`. If this feature is [applied to a method or a function](#), `@param` tags are created. In any other places PhpStorm adds an empty documentation stub.

[TODO patterns](#) inside documentation comments are also recognized.

Documentation comments in your source code are available for the [Quick Documentation Lookup](#) and open for review on pressing `Ctrl+Q` or `Command J`.

- [Example of JavaScript comment](#)
- [Enabling creation of documentation comments](#)
- [Creating a JSDoc comment block](#)

Example of JavaScript comment

Consider the following function:

```
function loadDocs(myParam1, myParam2){}
```

Type the opening documentation comment and press `Enter` to generate the documentation comment stub:

```
/**
```

```
*  
* @param myParam1  
* @param myParam2  
*/
```

To enable or disable automatic creation of documentation comments

1. [Open the Settings dialog box](#) and navigate to the [Editor, Smart Keys](#) page.
2. In the Enter section, select or clear **Insert documentation comment stub** check box.

To create a jsdoc comment block for a method or a function

1. Place the caret before the method or function declaration.
2. Type the opening block comment `/**` and press `EnterEnter`.
3. Describe the listed parameters and return values.

Tip

PhpStorm checks syntax in the comments and treats it according to the [Code Inspections](#) settings.

See Also

Procedures:

- [Viewing Inline Documentation](#)

Reference:

- [Inspections](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi> 
- <http://youtrack.jetbrains.com/issues/WI> 

Viewing JavaScript Reference

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Besides [common referential procedures](#) that are available in the context of any supported language, PhpStorm provides the following ways of retrieving JavaScript-specific reference:

- [Viewing JSDoc Comments](#)
- [Lookup in External JavaScript Libraries](#)

See Also

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi> 
- <http://youtrack.jetbrains.com/issues/WI> 

Viewing Inline Documentation

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Quick Documentation Lookup helps you get quick information for any symbol, provided that this symbol has been supplied with Documentation comments in the applicable format. PhpStorm recognizes inline documentation created in accordance with the [JavaScript Documentation Tool](#)  format.

In this section:

- [Viewing quick documentation.](#)
- [Changing font size in the quick documentation window.](#)
- [Toolbar of the quick documentation window.](#)

To view documentation for a symbol at caret, do one of the following

- On the main menu, choose **View | Quick Documentation Lookup**.
- Press `Ctrl+QCommand J`.

Note

When you explicitly invoke code completion, then quick documentation for an entry selected in the suggestion list can be displayed automatically. The behavior of quick documentation lookup is configured in the [Code Completion](#) page of the Settings dialog.

To change the font size of quick documentation, do one of the following

- Click the  button in the upper-right corner of the quick documentation window, and move the slider.

- Rotate the mouse wheel while keeping the `CtrlCtrl` key pressed.

Quick documentation lookup window

The **Quick Documentation Lookup** window helps navigate to the related symbols via hyperlinks, and provides a toolbar for moving back and forth through the already navigated pages, change font size, and viewing documentation in an external browser.

Icon	Keyboard shortcut	Action
	Alt+Shift+LeftCommand Shift Left Alt+Shift+RightCommand Shift Right	Navigate to the previous or next screen in the definition pop-up window after using hyperlinks in the definition. Note On a Mac OS X computer, you can also use the three-finger right-to-left and left-to-right swipe gestures.
	Shift+F1Shift F1	View documentation in an external browser.
		Click this button to show font size slider. Move the slider to increase or decrease the font size in the quick documentation window as required.

See Also

- Reference:
- [Viewing Reference Information](#)

- Web Resources:
- <http://www.jetbrains.com/devnet/community/wi>
 - <http://youtrack.jetbrains.com/issues/WI>

Documentation Look-up in External JavaScript Libraries

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Besides [common referential procedures](#) and [viewing JSDoc comments](#), you can also retrieve documentation for symbols defined in external JavaScript libraries and frameworks.

When this functionality is enabled, upon pressing `Shift+F1Shift F1` with the cursor positioned at the symbol in question, PhpStorm invokes the corresponding link and opens the documentation page in the [default PhpStorm browser](#).

- Note
- The default PhpStorm browser is configured on the [Web Browsers](#) page of the [Settings](#) dialog.
 - For more details on specifying the browser to use by default, see [Configuring Browsers](#).

To enable the external JavaScript library lookup functionality

1. Download the required libraries or frameworks.
2. [Configure the downloads as libraries](#) at the PhpStorm level.
3. [Specify links to external documentation](#).

To view documentation on a symbol defined in an external JavaScript library

- Position the cursor at the symbol in question and choose `View | External Documentation`, or press `Shift+F1Shift F1`.

See Also

- Procedures:
- [Viewing External Documentation](#)
 - [Viewing Reference Information](#)
 - [Viewing Inline Documentation](#)
 - [Configuring Browsers](#)

- Reference:
- [JavaScript Libraries](#)
 - [Web Browsers](#)

- Web Resources:
- <http://www.jetbrains.com/devnet/community/wi>
 - <http://youtrack.jetbrains.com/issues/WI>

Running and Debugging JavaScript

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

- In this part:
- [Running JavaScript](#)
 - [Debugging JavaScript](#)

To run a JavaScript from PhpStorm

No [run configuration](#) is required for launching applications with injected JavaScript code from PhpStorm. You just open the starting HTML page of your application in the browser and that's it.

1. Open the HTML file that implements the starting page of your application in the editor.
2. Do one of the following:
 - o To have the application opened in the [PhpStorm default browser](#), choose **View | Open in browser** on the main menu.
 - o To have the application opened in Firefox, choose **View | Reload in Firefox** on the main menu.
 - o To have the application opened in a specific browser of your choice, choose **View | Web preview** on the main menu or press **Alt+F2**. Then select the desired browser from the pop-up menu.

Tip

Alternatively, hover your mouse pointer over the code to show the browser icons bar, and click the icon that indicates the desired browser:



To debug JavaScript code, perform these general steps:

1. [Configure the JavaScript debugger.](#)
2. [Configure and set breakpoints](#) in the JavaScript code as required.
3. [Initiate a debugging session.](#)
4. [Step through the program, stop and resume](#) program execution, [examine it when suspended](#), etc.

Note

JavaScript debugging is supported for **Firefox** or **Google Chrome** only.

See Also

Concepts:

- [Running and Debugging](#)

Procedures:

- [Previewing Pages with Web Contents in a Browser](#)
- [Debugging](#)

Reference:

- [Web Browsers](#)
- [Debug Tool Window](#)

External Links:

- [http://wiki.jetbrains.net/intellij/Debugging JavaScript locally in Firefox with WebStorm and PhpStorm](http://wiki.jetbrains.net/intellij/Debugging_JavaScript_locally_in_Firefox_with_WebStorm_and_PhpStorm)
- [http://wiki.jetbrains.net/intellij/Remote JavaScript debugging with WebStorm and PhpStorm](http://wiki.jetbrains.net/intellij/Remote_JavaScript_debugging_with_WebStorm_and_PHPStorm)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Configuring JavaScript Debugger

Previous | [Next](#) | [See Also](#) | [Comments](#)

At the PhpStorm level, debugging of JavaScript injections is supported through the JavaScript Debugger [bundled plugin](#), which is by default enabled. If not, activate it in the [Plugins](#) page of the [Settings](#) dialog box.

Debugging of JavaScript code is supported in the [Firefox](#) and [Google Chrome](#) browsers by means of browser-specific **JetBrains extensions**. No special steps are required from your side to install or enable these extensions. PhpStorm does it for when you [initiate a debugging session](#) for the first time.

During a debugging session, PhpStorm considers any injected JavaScript code in a Web page opened in the browser as subject for debugging. This means that you cannot use the same instance of a browser for browsing and for debugging simultaneously. For Google Chrome you can solve this problem by configuring PhpStorm to use a separate Chrome user profile for debugging.

In this section:

To configure settings required for debugging JavaScript code

1. [Open the Project Settings](#) dialog box, and then click **Debugger**.
2. In the [Debugger JavaScript](#) page that opens specify the following options:
 - o In the **Value tooltip delay (ms)** text box, specify the time period between pointing at an object and displaying the object's value in a tooltip.
 - o Whether to display DOM properties in a tab or not.
 - o Whether to display function values in a tab or not. If you decide to have function values displayed, specify the range of functions to show vales of. The available options are:
 - **All**
 - **User-defined**
 - o To have the debugger skip specific scripts:
 1. Select the **Do not step into scripts** check box.

2. Configure a list of scripts not to be involved: use the **Add** and **Remove** buttons and specify the URL addresses of scripts in question.

See Also

Concepts:

- [Running and Debugging](#)

Procedures:

- [JavaScript-Specific Guidelines](#)
- [Debugging](#)

Reference:

- [Debugger. JavaScript](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi> 
- <http://youtrack.jetbrains.com/issues/WI> 

Debugging JavaScript

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac 

When an HTML file with injected JavaScript code is opened in the browser, the script execution starts and suspends on reaching the first breakpoint, whereupon you can analyze the suspended program, step through it, resume execution when ready, etc.

Note

This HTML file does not necessarily have to be the one that implements the starting page of the application.

Depending on the location of application files, PhpStorm distinguishes between the following **local** and **remote** debugging modes.

- **Local** debugging supposes that the application files are on your machine. In this mode, you can initiate a debugging session in two ways:
 - Manually, by opening a single HTML file with a JavaScript injection in the browser.

Note

The file will be opened in the [PhpStorm default browser](#). Make sure the default browser is either **Firefox** or **Google Chrome**, because debugging JavaScript is supported only in them.

- Through a debug configuration. In this case, you can specify the browser to use.
- In case of **Remote** debugging, the application files are deployed on a remote host and you have their copies on your computer.

When a remote HTML file with a JavaScript injection is opened, the debugger tells PhpStorm the name of the currently processed file and the number of the line to be processed. PhpStorm opens the local copy of this file and indicates the line with the provided number. This behaviour is enabled by specifying correspondence between files and folders on the server and their local copies. This correspondence is called **mapping**.

A **Remote** debugging session can be initiated only through a debug configuration.

- [Debugging a single local HTML file with injected JavaScript code](#)
- [Initiating a local debugging session through a debug configuration](#)
- [Remote debugging](#)

To debug JavaScript code in a single local HTML file

1. Set the [breakpoints](#) in the JavaScript injections, as required.
2. Open the required HTML file or select it in the [Project view](#).
3. On the context menu, choose **Debug <file name>**. The file opens in the default browser.
4. In the [Debug](#) tool window that opens, proceed as usual: [step through the program](#), [stop and resume](#) program execution, [examine it when suspended](#), etc.

To initiate a local debugging session through a debug configuration

1. Set the [breakpoints](#) in the JavaScript injections, as required.
2. [Create a debug configuration](#) of the type **JavaScript Debug - Local**.
 1. On the main menu, choose **Run | Edit Configurations**.
 2. Click the **Add New Configuration** toolbar button , and choose **JavaScript Debug - Local** on the context menu.
 3. In the [Run/Debug Configuration: JavaScript Debug](#) dialog box that opens, specify the full path to the HTML file that implements the page to start debugging from. Choose the browse to debug the application in (**Firefox** or **Google Chrome**). Click **OK** to save the configuration settings.
3. Choose **Run | Debug** on the main menu, or click the **Debug** button  on the toolbar, or press **Shift+F9**. The specified starting HTML file opens in the default browser.
4. In the [Debug](#) tool window that opens, proceed as usual: [step through the program](#), [stop and resume](#) program execution, [examine it when suspended](#), etc.

To debug a remote application with JavaScript injections

1. Synchronize with the server where the target application is deployed.

Note

Interaction with servers is supported through the `Remote Hosts Access` bundled plugin, which is by default enabled. If not, activate it in the [Plugins](#) page of the [Settings](#) dialog box.

1. [Configure access to the server](#) and to the [application files](#).
2. [Configure the synchronization](#).
3. [Download the application files](#).
2. Set the [breakpoints](#) in the JavaScript injections, as required.
3. [Create a debug configuration](#) of the type `JavaScript Debug - Remote`.
 1. On the main menu, choose `Run | Edit Configurations`.
 2. Click the `Add New Configuration` toolbar button , and choose `JavaScript Debug - Remote` on the context menu.
 3. In the `Run/Debug Configuration: JavaScript Debug` dialog box that opens, choose the browser to debug the application in (`Firefox` or `Google Chrome`).

In the `URL to open` text box, type the URL address of the page to start debugging from.

Set correspondence between files on the server and their local copies by mapping paths to local files to URL addresses of files on the server. You can specify URL addresses in the absolute form (`http://<host>:<port>/<path_to_server_root>/<path_to_file>`) or relative to the [server root URL](#) (`/<path_to_file>`).

Click `OK` to save the configuration settings.

4. Choose `Run | Debug` on the main menu, or click `Debug` button  on the toolbar, or press `Shift+F9` `Shift F9`. The specified starting HTML file opens in the default browser.
5. In the `Debug` tool window that opens, proceed as usual: [step through the program](#), [stop and resume](#) program execution, [examine it when suspended](#), etc.

See Also

Procedures:

- [Debugging](#)

External Links:

- [http://wiki.jetbrains.net/intellij/Debugging JavaScript locally in Firefox with WebStorm and PhpStorm](http://wiki.jetbrains.net/intellij/Debugging_JavaScript_locally_in_Firefox_with_WebStorm_and_PhpStorm) 
- [http://wiki.jetbrains.net/intellij/Remote JavaScript debugging with WebStorm and PhpStorm](http://wiki.jetbrains.net/intellij/Remote_JavaScript_debugging_with_WebStorm_and_PHPStorm) 
- [http://wiki.jetbrains.net/intellij/Debugging a third-party site with WebStorm and PhpStorm](http://wiki.jetbrains.net/intellij/Debugging_a_third-party_site_with_WebStorm_and_PHPStorm) 

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

2.0+

Setting Labels to Variables, Objects and Watches

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: 

You can add your own label to a variable, object, or watch and then reference it as if it were a local variable `<label-name>_DebugLabel` defined in the same context where the expression is evaluated. PhpStorm also displays these labels in suggestion pop-up lists for code completion in the [Evaluate Expression](#) dialog box.

To set a label

1. Select the desired watch in the list
2. Select `Mark Object` on the context menu or press `F11F11`. The `Select object` label dialog box opens.
3. Specify the label name.
4. Click the `Browse` button  to change the label color. Click `OK` when ready.

Note

To remove a label, right-click the item and select `Unmark Object` on the context menu, or select the item in the list and press `F11F11`.

See Also

Procedures:

- [Debugging](#)
- [Evaluating Expressions](#)

Reference:

- [Debug Tool Window](#)
- [Evaluate Expression](#)
- [Debug Tool Window. Variables](#)
- [Debug Tool Window. Watches](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

JavaScript-Specific Refactorings

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

In addition to the full range of [common refactorings](#), PhpStorm provides the following JavaScript-specific refactorings:

- [Introduce Parameter in JavaScript](#)
- [Change Signature in JavaScript](#)
- [Introduce Variable in JavaScript](#)

See Also

Procedures:

- [Refactoring Source Code](#)

Reference:

- [Refactoring Dialogs](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Introduce Parameter in JavaScript

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

The `Introduce Parameter` refactoring is used to add a new parameter to a method declaration and to update the method calls accordingly.

- [Examples](#)
- [Introducing a parameter in JavaScript](#)

Examples

The following table shows two different ways of introducing a function parameter.

In the first of the examples a new parameter is introduced as an optional parameter. So the corresponding function call doesn't change.

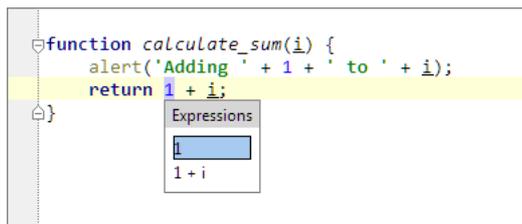
In the second of the examples the parameter is introduced as a required parameter. So the corresponding function call changes accordingly.

Before	After
<pre>// A new parameter will be added to this // function to replace the 1's: function calculate_sum(i) { alert('Adding ' + 1 + ' to ' + i); return 1 + i; } function show_sum() { // Here is the function call: alert('Result: ' + calculate_sum(5)); } // When adding a new parameter we'll specify // that it should be an optional one.</pre>	<pre>// The new parameter i2 has been added // as an optional parameter: function calculate_sum(i, i2) { i2 = i2 1; alert('Adding ' + i2 + ' to ' + i); return i2 + i; } function show_sum() { // The function call has not changed: alert('Result: ' + calculate_sum(5)); }</pre>
<pre>// A new parameter will be added to this // function to replace the 1's: function calculate_sum(i) { alert('Adding ' + 1 + ' to ' + i); return 1 + i; } function show_sum() { // Here is the function call: alert('Result: ' + calculate_sum(5)); } // When adding a new parameter we'll specify // that it should be a required one.</pre>	<pre>// The new parameter i2 has been added // as a required parameter: function calculate_sum(i, i2) { alert('Adding ' + i2 + ' to ' + i); return i2 + i; } function show_sum() { // The function call changed accordingly: alert('Result: ' + calculate_sum(5, 1)); }</pre>

Introducing a parameter in JavaScript

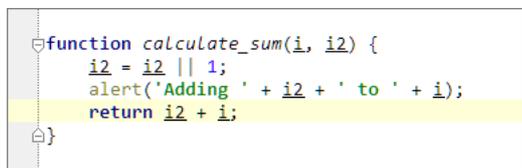
1. In the editor, place the cursor within the expression to be replaced by a parameter.
2. Do one of the following:
 - Press `Ctrl+Alt+P` / `Command Alt P`.
 - Choose `Refactor | Introduce Parameter` in the main menu.

- Select Refactor | Introduce Parameter from the context menu.
3. If more than one expression is detected for the current cursor position, the Expressions list appears. If this is the case, select the required expression. To do that, click the expression. Alternatively, use the UpUp and DownDown arrow keys to navigate to the expression of interest, and then press EnterEnter to select it.



4. In the [Introduce Parameter dialog](#):
1. Specify the parameter name in the Name field.
 2. The Value field, initially, contains the expression that you have selected. Normally, you don't need to change this. (Depending on whether the option **Optional parameter** is selected or not, this will be the value assigned to the new parameter in the function body or passed to the function in the function calls.)
 3. If, when introducing the new parameter, you don't want to change the function calls, select the **Optional parameter** check box. If you do so, the value specified in the Value field will be assigned to the new parameter in the function body. The calls to the function (if any) won't change.

If you want to pass the value (specified in the Value field) to the new parameter through the existing function calls, clear the **Optional parameter** check box. If you do so, all the function calls will change according to the new function signature; no explicit assignment of the parameter value will be added to the function body.
 4. If more than one occurrence of the expression is found within the function body, you can choose to replace only the selected occurrence or all the found occurrences with the references to the new parameter. Use the **Replace all occurrences** check box to specify your intention.
 5. Click OK.



See Also

Procedures:

- [Refactoring Source Code](#)
- [JavaScript-Specific Guidelines](#)

Reference:

- [Introduce Parameter Dialog for JavaScript](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Change Signature in JavaScript

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

In JavaScript, you can use the Change Signature refactoring to:

- Change the function name.
- Add new parameters and remove the existing ones. Note that you can also add a parameter using a dedicated [Introduce Parameter](#) refactoring.
- Reorder parameters.
- Change parameter names.
- Propagate new parameters through the method call hierarchy.

On this page:

- [Examples](#)
- [Changing a function signature](#)

Examples

The following table shows 4 different ways of performing the same Change Signature refactoring.

In all the cases, the function `result()` is renamed to `generate_result()` and a new parameter `input` is added to this function.

The examples show how the function call, the calling function (`show_result()`) and other code fragments may be affected depending on the refactoring settings.

Before

```
// This is the function whose signature will be changed:

function result() {
    // some code here
}
```

After

```
// The function has been renamed to generate_result.
// The new parameter input has been added.

function generate_result(input) {
    // some code here
}
```

```

function show_result() {

    // Here is the function call:

    alert('Result: ' + result());
}

// Now we'll rename the function and
// add one (required) parameter.

// This is the function whose signature will be changed:

function result() {

    // some code here
}

function show_result() {

    // Here is the function call:

    alert('Result: ' + result());
}

// Now we'll rename the function and add one parameter.
// This time, we'll specify that the parameter is optional.

// This is the function whose signature will be changed:

function result() {
    // some code here
}

// This function will also change its signature:

function show_result() {

    // Here is the function call:

    alert('Result: ' + result());
}

// Now we'll rename the function and add one required
// parameter. We'll also ask PhpStorm to propagate
// the new parameter through the calling function show_result()
// to the function call.

// This is the function whose signature will be changed:

function result() {

    // some code here
}

// This function will also change its signature:

function show_result() {

    // Here is the function call:

    alert('Result: ' + result());
}

// Now we'll rename the function and add one optional
// parameter. We'll also ask PhpStorm to propagate
// the new parameter through the calling function show_result()
// to the function call.

```

```

function show_result() {

    // The function call changed accordingly:

    alert('Result: ' + generate_result(100));
}

// When performing the refactoring, 100 was specified as
// the parameter value.

// The function has been renamed to generate_result.
// The new optional parameter input has been added.

function generate_result(input) {
    input = input || 100;
    // some code here
}

function show_result() {

    // The function call changed accordingly:

    alert('Result: ' + generate_result(100));
}

// When performing the refactoring, 100 was specified as
// the parameter value.

// The function has been renamed to generate_result.
// The new parameter input has been added.

function generate_result(input) {
    // some code here
}

// Note the new function parameter:

function show_result(input) {

    // The function call changed accordingly:

    alert('Result: ' + generate_result(input));
}

// The function has been renamed to generate_result.
// The new optional parameter input has been added.

function generate_result(input) {
    input = input || 100;
    // some code here
}

// Note the new function parameter:

function show_result(input) {
    input = input || 100;

    // The function call changed accordingly:

    alert('Result: ' + generate_result(input));
}

// When performing the refactoring, 100 was specified as
// the parameter value.

```

Changing a function signature

1. In the editor, place the cursor within the name of the function whose signature you want to change.
2. Do one of the following:
 - o Press **Ctrl+F6** Command **F6**.
 - o Choose **Refactor | Change Signature** in the main menu.

- o Select Refactor | Change Signature from the context menu.
3. In the [Change Signature dialog](#), make the necessary changes to the function signature and specify which other, related changes are required.

You can:

- o Change the function name. To do that, edit the text in the Name field.
- o Manage the function parameters using the table of parameters and the buttons to the right of it:
 - To add a new parameter, click  (Alt+InsertCommand N) and specify the properties of the new parameter in the corresponding fields.

When adding parameters, you may want to [propagate these parameters](#) to the functions that call the current function.
 - To remove a parameter, click any of the cells in the corresponding row and click  (Alt+DeleteMeta Delete).
 - To reorder the parameters, use  (Alt+UpCommand Up) and  (Alt+DownCommand Down). For example, if you wanted to make a certain parameter the first in the list, you would click any of the cells in the row corresponding to that parameter, and then click  the required number of times.
 - To change the name of a parameter, make the necessary edits in the corresponding table cell.
- o Propagate new function parameters (if any) along the hierarchy of the functions that call the current function.

(There may be the functions that call the function whose signature you are changing. These functions, in their turn, may be called by other functions, and so on. You can propagate the changes you are making to the parameters of the current function through the hierarchy of the calling functions and also specify which calling functions should be affected and which shouldn't.)

To propagate the new parameters:

1. Click  (Alt+GAlt G).
2. In the left-hand pane of the **Select Methods to Propagate New Parameters** dialog, expand the necessary nodes and select the check boxes next to the functions you want the new parameters to be propagated to.

Note

To help you select the necessary functions, the code for the calling function and the function being called is shown in the right-hand part of the dialog (in the **Caller Method** and **Callee Method** panes respectively).

As you switch between the functions in the left-hand pane, the code in the right-hand part changes accordingly.

3. Click OK.
4. To perform the refactoring right away, click Refactor.

To [see the expected changes](#) and make the necessary adjustments prior to actually performing the refactoring, click Preview.

See Also

Reference:

- [Change Signature Dialog for JavaScript](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Introduce Variable in JavaScript

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts:  

In JavaScript and ActionScript, you can replace an expression with a variable or a constant. For JavaScript 1.7 or a later version, there is also an option of introducing a local variable.

JavaScript examples

Before

```

Parenizor.method('toString', function () {
    return '(' + this.getValue() + ')';
})

var browserName = "N/A";
if (navigator.appName.indexOf("Netscape") != -1) {
    browserName = "NS";
} else if (navigator.appName.indexOf("Microsoft") != -1) {
    browserName = "MSIE";
} else if (navigator.appName.indexOf("Opera") != -1) {
    browserName = "O";
}
    
```

After

```

GLOBAL VARIABLE

Parenizor.method('toString', function () {
    var string = '(' + this.getValue() + ')';
    return string;
})

LOCAL VARIABLE

Parenizor.method('toString', function () {
    let string = '(' + this.getValue() + ')';
    return string;
})

var browserName = "N/A";
var appName = navigator.appName;
if (appName.indexOf("Netscape") != -1) {
    browserName = "NS";
} else if (appName.indexOf("Microsoft") != -1) {
    browserName = "MSIE";
} else if (appName.indexOf("Opera") != -1) {
    browserName = "O";
}
    
```

}

To introduce a variable

1. In the editor, select the expression to be replaced with a variable. You can do that yourself or use the **smart expression selection** feature to let PhpStorm help you. So, do one of the following:

- o Highlight the expression. Then choose Refactor | Introduce Variable from the main or the context menu. Alternatively, press Ctrl+Alt+VCommand Alt V.

```

var browserName = "N/A";
if (navigator.appName.indexOf("Netscape") != -1) {
    browserName = "NS";
} else if (navigator.appName.indexOf("Microsoft") != -1) {
    browserName = "MSIE";
} else if (navigator.appName.indexOf("Opera") != -1) {
    browserName = "O";
}

```

- o Place the cursor before or within the expression. Choose Refactor | Introduce Variable from the main or the context menu, or press Ctrl+Alt+VCommand Alt V.

```

var browserName = "N/A";
if (navigator.appName.indexOf("Netscape") != -1) {
    browserName = "NS";
} else if (navigator.appName.indexOf("Microsoft") != -1) {
    browserName = "MSIE";
} else if (navigator.appName.indexOf("Opera") != -1) {
    browserName = "O";
}

```

In the Expressions pop-up menu, select the expression. To do that, click the required expression. Alternatively, use the UpUp and DownDown arrow keys to navigate to the expression of interest, and then press EnterEnter to select it.

```

var browserName = "N/A";
if (navigator.appName.indexOf("Netscape") != -1) {
    browserName = "NS";
} else if (navigator.appName.indexOf("Microsoft") != -1) {
    browserName = "MSIE";
} else if (navigator.appName.indexOf("Opera") != -1) {
    browserName = "O";
}
var browserVers
var versionMatc
if (versionMatc

```

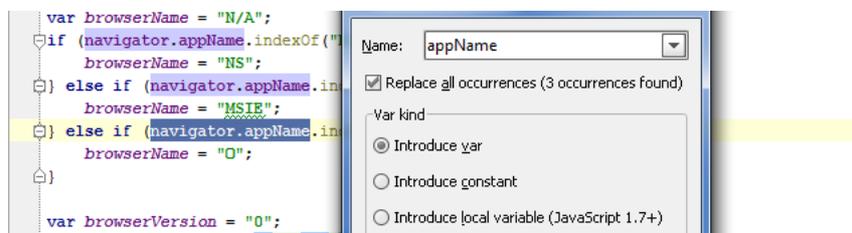
Note

The Expressions pop-up menu contains all the expressions appropriate for the current cursor position in the editor.

When you navigate through the suggested expressions in the pop-up, the code highlighting in the editor changes accordingly.

2. In the [Introduce Variable dialog](#):

1. Specify the variable name next to Name. You can select one of the suggested names from the list or type the name in the Name box.
2. If more than one occurrence of the selected expression is found, you can select to replace all the found occurrences by selecting the corresponding check box. If you want to replace only the current occurrence, clear the **Replace all occurrences** check box.
3. If you are working with JavaScript, select to introduce a (global) variable, a constant, or a local variable. To do that, click the required option in the **Var kind** area.
4. For ActionScript, you can choose to introduce a constant rather than a variable. To do that, select the **Make constant** check box.
5. Click OK.



See Also

Procedures:

- [Introduce Variable](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Unit Testing JavaScript

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm provides facilities for running JavaScript unit tests against a local or remote test server. You can run test cases, methods, or entire test files.

To create and run unit tests in JavaScript, perform these general steps

1. [Enable JavaScript unit testing support](#).
2. [Write the unit tests](#).
3. To have PhpStorm recognize the unit tests and detect the corresponding production source code, mark the folder where the unit tests are stored as [test folder](#).
4. If applicable, write configuration files that define which test and corresponding production files should be loaded into the browser and in which order.
5. Start the test server and capture a browser.
6. [Launch unit tests](#) and [monitor test results](#) in the [Run](#) tool window.

The topics in this part provide guidelines in JavaScript-specific unit testing procedures.

- [Enabling JavaScript Unit Testing Support](#)
- [Creating JavaScript Unit Tests](#)
- [Running JavaScript Unit Tests in Browser](#)

For general information on testing in PhpStorm, see the section [Testing](#).

See Also

Concepts:

- [Running and Debugging](#)

Procedures:

- [Testing](#)
- [Configuring JavaScript Libraries](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Enabling JavaScript Unit Testing Support

Previous | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

At the PhpStorm level, JavaScript browser-side unit testing is supported through the JSTestDriver [plugin](#). This plugin does the following:

- Runs the [JSTestDriver server](#) that captures an opened browser to execute tests in.
- During the test creation, detects the unit testing framework the test code complies with, whereupon suggests the **Add <test framework> support intention action**, provided that the framework is recognized as a [PhpStorm JavaScript library](#) and is thus available in the IDE.

At the project level, JavaScript unit testing is supported through integration with JavaScript unit testing frameworks:

- [JSTestDriver Assertion](#) framework. The framework libraries are bundled with the JSTestDriver plugin.
- [Jasmine](#).
- [QUnit](#).

In this section:

- [Enabling JavaScript unit testing in PhpStorm](#)
- [Enabling framework-specific JavaScript unit testing in a project](#)

To enable JavaScript unit testing in PhpStorm

1. [Download and install](#) the JSTestDriver repository plugin from the JetBrains default repository.
2. [Enable](#) the JSTestDriver plugin.
3. Restart PhpStorm for the changes to take effect.

To enable framework-specific JavaScript unit testing in a project

1. Download the framework of your choice.
2. Configure the downloaded framework as a [PhpStorm JavaScript library](#).
3. Do one of the following:
 - [Add the project folder to the library scope](#).
 - Enable the framework support on-the-fly [during test creation](#). Write the test as required, position the cursor at it, and press **Alt+Enter**. Then choose the **Add <test framework> support intention action** from the list.

See Also

Procedures:

- [Configuring JavaScript Libraries](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Testing Support:

- [Running and Debugging](#)
- [Testing](#)

Creating JavaScript Unit Tests

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm provides integration with JavaScript unit testing frameworks thus providing rich coding assistance for writing JavaScript unit tests. To get framework-aware coding assistance, download the desired framework, [integrate it with the PhpStorm](#), and [add the project root to the library scope](#).

To create JavaScript unit tests, perform these general steps

- [Create a folder](#) test at the same level as the `src` folder
- Populate the `test` folder. For each production file, create a separate test file and name it as follows: `<name of production file>.<Test>.js`
- Mark the folder where the tests are stored as [test folder](#).
- If necessary, have PhpStorm detect and [enable the required testing framework on-the-fly](#).

See Also

Procedures:

- [Testing](#)
- [Running JavaScript Unit Tests in Browser](#)
- [Configuring JavaScript Libraries](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Running JavaScript Unit Tests in Browser

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

JavaScript unit tests are run in a browser against a test server which actually handles the testing process.

The test server does not necessarily have to be on your machine. PhpStorm has a local test server that is launched right from the IDE. This server allows you to capture a browser to run tests in, loads the test targets to the captured browser, controls the testing process, and exchanges data between the browser and PhpStorm, so you can view test results without leaving the IDE.

To get access to using the test server, [download, install](#), and [enable the JSTestDriver plugin from the JetBrains default repository](#). Then restart PhpStorm for the changes to take effect.

The test server loads tests to the browser according to the [configuration files](#) that define which test and corresponding production files should be loaded and in which order. JSTestDriver server treats `*jstd` and `JSTestDriver.conf` files as test runner configuration files. There are two ways to obtain configuration files:

- Create one or several configuration files in advance manually and then specify them in the [run configuration](#).
- Have PhpStorm generate them automatically by choosing the **JSTestDriver configuration file: Generated** option in the [run configuration](#). You only need to specify the test file, method, or case, whereupon PhpStorm will detect the corresponding production file itself.

To run JavaScript unit tests, perform these general steps:

- [Create a test runner configuration file manually](#)
- [Start a PhpStorm default JSTestDriver test server](#)
- [Capture a browser to execute tests in](#)
- [Create a JSTestDriver run configuration](#)
- [Launch the unit tests](#)

To create a test runner configuration file manually

1. In the **Project** tree, select the parent folder of the `src` (production) and the `test` folders, and choose **New | File** on the context menu.
2. In the **New File** dialog box, that opens, type a name of the configuration file with the extension `jstd` or `conf`.
3. Open the new file in the editor and specify the full path to the current folder and paths to the files to load relative to the current folder. Use wildcards to specify file name patterns. The required format is [YAML](#), for more details, see [description of test runner configuration files](#).

You can also have a configuration file generated automatically

To start a PhpStorm default jstestdriver test server

If you are going to use a test server running on another machine or listening to another port, start it according to the server-specific instructions and specify its URL address in the **Server** area of the [Run/Debug Configuration: JSTestDriver](#) dialog box.

1. Make sure the JSTestDriver [plugin is enabled](#).
2. On the main menu, choose **View | Tool Windows | JSTestDriver Server**.
3. In the **JSTestDriver Server** tool window, that opens, click the **Run a local server** toolbar button .

To stop the server when you are through with unit testing, click the **Stop the local server** toolbar button .

To capture a browser for executing tests in

1. With the [JSTestDriver Server running](#), open the **JSTestDriver Server** tool window.
2. Copy the contents of the **Capture a browser using the URL** read-only field.
3. Open a browser of your choice and paste the copied URL address. The **Remote Console** of the **JSTestDriver** opens.
4. Switch to the **JSTestDriver Server** tool window and click the icon that indicates the browser you just opened. PhpStorm displays a message informing you that it is ready for executing tests.

To create a JSTestDriver run configuration

1. Open the [Run/Debug Configuration](#) dialog box by doing one of the following:
 - On the main menu, choose **Run | Edit Configurations**.
 - Press `RunConfigurationRunConfiguration` and choose **Edit Configuration** on the context menu.
2. Click the **Add** button  on the toolbar and select the **JSTestDriver** configuration type.
3. In the dialog box that opens, specify the test scope, configuration parameters, and activities to perform before test execution.
4. Apply the changes and close the dialog box.

To launch JavaScript unit tests

1. On the main toolbar, select the **JSTestDriver run/debug configuration** from the list.
2. Click the **Run** button  to the right of the list.

See Also

Procedures:

- [Running and Debugging](#)
- [Testing](#)

Reference:

- [Run/Debug Configuration: JSTestDriver](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Using JSLint Code Quality Tool

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm contains a built-in [JSLint](#)  verifier. This tool checks JavaScript code for most common mistakes and discrepancies without running the application.

To enable the built-in JSLint tool

1. [Open the project settings](#), and then click **Inspectios**.
2. In the central pane of the [Inspections](#) page, that opens, expand the **JavaScript** node, then expand the **General** node.
3. Select the check box next to **JSLint Validation**.
4. In the right pane of the page, define the set of common mistakes to check the code for. To enable a validation, select the check box next to it.
5. Configure the [inspection severity](#).

See Also

Procedures:

- [Code Inspection](#)
- [JavaScript-Specific Guidelines](#)

Reference:

- [Inspections](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

CoffeeScript Support

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: 

PhpStorm provides [CoffeeScript](#)  support. PhpStorm recognizes `*.coffee` files, and allows to edit them.

In this section:

- [Prerequisites](#)
- [CoffeeScript Support](#)

Prerequisites

Note

Before you start working with CoffeeScript files, make sure that **CoffeeScript** plugin is enabled. The plugin is bundled with PhpStorm and activated by default.

If it is not, [enable the plugin](#).

CoffeeScript support

CoffeeScript files are marked with  icon.

CoffeeScript support includes:

1. Coding assistance:
 - [Code completion](#) for keywords, labels, variables, parameters and functions.
 - Error and syntax highlighting.
 - Code [formatting](#) and [folding](#).
2. [Code generation](#)
 - Generating code stubs based on [file templates](#) during file creation.
 - Possibility to create [line and block comments](#) (Ctrl+Slash Command Slash Or Ctrl+Divide Command Divide/Ctrl+Shift+Slash Command Shift Slash Or Ctrl+Shift+Divide Command Shift Divide).
3. Numerous ways to [navigate](#) through the source code, among them:
 - [Navigating with Structure View](#).
 - [Navigate | Declaration](#) (Ctrl+BCommand B).
 - [Navigate | Implementation](#) (Ctrl+Alt+BCommand Alt B) from overridden method / subclassed class.
4. Advanced facilities to [search through the source code](#).
5. [Running](#).

Note

For executing CoffeeScript files, the following prerequisites should be observed:

- [Node.js](#)  utility is downloaded and installed on your machine
- NodeJS repository plugin is [installed](#) and [enabled](#).

See Also

Language and Framework-Specific Guidelines:

- [JavaScript-Specific Guidelines](#)
- [NodeJS](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

NodeJS

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm supports integration with the [NodeJS](#)  framework thus enabling running, debugging, and unit testing of **NodeJS** applications. This integration is provided by the **NodeJS** plugin, which is not bundled with PhpStorm.

PhpStorm recognizes NodeJS code and provides basic coding assistance and highlighting for it. To get guidance in Node development, see [HowToNode.org](#) .

In this section:

- [Prerequisites](#)
- [Changes to the UI](#)
- [Enabling NodeJS](#)

In this part:

- [Running and Debugging Node.JS](#)
- [Unit Testing Node.JS](#)

Prerequisites

Note

- **NodeJS** plugin is [installed](#) and [enabled](#).

This plugin is not bundled with PhpStorm, but it is available from the [JetBrains plugin repository](#) . Once enabled, NodeJS support is available at the IDE level, that is, you can use it in all your PhpStorm projects.

- [NodeJS](#)  framework is downloaded and installed on your computer.

Note

You can have several versions of the framework installed and switch between them depending on the requirements of a specific project. To appoint one of the installations as default, click the **Change NodeJS Version** toolbar button  and specify the folder with the NodeJS version. PhpStorm will find the executable file in the folder itself and always show the path to it in the [Run/Debug Configuration: NodeJS](#) dialog box.

Changes to the UI

Once enabled, NodeJS plugin introduces the following changes to the PhpStorm UI:

- Dedicated button  appears on the main toolbar.
- Run/debug configurations are added.

To enable NodeJS support in PhpStorm

1. Make sure that [NodeJS](#) framework is downloaded and installed on your computer.
2. [Install](#) and [enable](#) the **NodeJS** repository plugin.
3. Restart PhpStorm for the changes to take effect.

See Also

Procedures:

- [JavaScript-Specific Guidelines](#)
- [CoffeeScript Support](#)

Reference:

- [Plugins](#)
- [Run/Debug Configuration: Node JS](#)
- [Run/Debug Configuration: Node JS Remote Debug](#)
- [Run/Debug Configuration: Nodeunit](#)

External Links:

- <http://nodejs.org>
- <http://howtonode.org/>

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Running and Debugging Node.JS

Previous | [Next](#) | [See Also](#) | [Comments](#)

Running and debugging of NodeJS applications in PhpStorm is supported through the **NodeJS** plugin. The plugin is not bundled with PhpStorm, but it is available from the [JetBrains plugin repository](#). Make sure you have [installed and enabled](#) the plugin before running or debugging.

Running NodeJS application in PhpStorm is supported only in the **local** mode. This means that PhpStorm itself starts the NodeJS engine and the target application according to a [run configuration](#) and gets full control over the session.

Debugging can be performed in two modes:

- Locally, with the NodeJS engine started from PhpStorm.
- Remotely, when PhpStorm connects to an already running NodeJS application. This approach gives you the possibility to re-start a debugging session without re-starting the NodeJS server.

In either case, the debugging session is initiated through a [debug configuration](#)

In this section:

- [Running a NodeJS application.](#)
- [Debugging a NodeJS application locally](#)
- [Debugging a running NodeJS application](#)
- [Creating a NodeJS run/debug configuration](#)
- [Creating a NodeJS Remote Debug configuration.](#)

To run a NodeJS application

1. [Create a NodeJS run configuration](#).
2. To [launch](#) the application, select the run configuration from the list on the main tool bar and then choose **Run | Run <configuration name>** on the main menu or click the **Run** toolbar button .
3. In the **Run** tool window, that opens, copy the URL address at which the application is running. The URL address is specified in the following information message:

```
Server running at http://<host>:<port>/
```

4. Open the browser of your choice and open the page with the copied URL address. The page shows the result of executing your NodeJS application.

To debug a NodeJS application locally

1. Set the [breakpoints](#) in the NodeJS code, where necessary.
2. [Create a NodeJS run/debug configuration](#).
3. To [start a debugging session](#), select the debug configuration from the list on the main tool bar and then choose **Run | Debug <configuration name>** on the main menu or click the **Debug** toolbar button .
4. In the **Console** tab of the **Debug** tool window, that opens, copy the URL address at which the application is running. The URL address is specified in the following information message:

```
Server running at http://<host>:<port>/
```

- Open the browser of your choice and open the page with the copied URL address. Control over the debugging session returns to PhpStorm.
- Switch to PhpStorm, where the controls of the **Debug** tool window are now enabled. Proceed with the debugging session [step through the breakpoints](#), switch between frames, change values on-the-fly, [examine a suspended program](#), [evaluate expressions](#), and [set watches](#).

To debug a running NodeJS application

The procedure falls into two parts: first we start an application as usual and then connect to it with the debugger.

- Set the [breakpoints](#) in the NodeJS code, where necessary.
- [Create a NodeJS run configuration](#) with the following option in the **Node parameters** text box: `--debug=<port for connect to debugger remotely>`
- [Run the application](#) with the created configuration.
- [Create a NodeJS Remote Debug](#) configuration: in the **Debug port** text box, type the port number specified in the currently running NodeJS configuration.
- With the application still running, launch the **NodeJS Remote Debug** configuration (select the configuration in the list and click the **Debug** toolbar button .
- In the Run tool window, copy the URL address of the server and open the corresponding page in the browser. Control over the debugging session returns to PhpStorm.
- Switch to PhpStorm. In the **Debug** tool window, [step through the breakpoints](#), switch between frames, change values on-the-fly, [examine a suspended program](#), [evaluate expressions](#), and [set watches](#).

To create a NodeJS run/debug configuration

- On the main menu, choose **Run | Edit Configurations**.
- In the **Edit Configuration** dialog box, that opens, click the **Add New Configuration** toolbar button , and choose **NodeJS** on the context menu.
- In the **Run/Debug Configuration: NodeJS** dialog box, that opens, specify the following:
 - The name of the configuration.
 - The path to the NodeJS executable file.

Note

This field is already filled in if you have [appointed a default installation](#).

- To enable [remote debugging](#) of the application, specify the following option in the **Node parameters** text box: `--debug=<port for connect to debugger remotely>`
 - The location of the file to start running the NodeJS application.
 - If the file to run references any other files, specify their location in the **Working directory** field.
 - If applicable, in the **Application parameters** text box, specify the arguments to be passed to the application on start through the [process.argv](#)  array.
- Click **OK**, when ready.

To create a NodeJS Remote Debug configuration

- On the main menu, choose **Run | Edit Configurations**.
- In the **Edit Configuration** dialog box, that opens, click the **Add New Configuration** toolbar button , and choose **NodeJS Remote Debug** on the context menu.
- In the **Run/Debug Configuration: NodeJS Remote Debug** dialog box, that opens, specify the following:
 - The name of the configuration.
 - The host where the target application is running.
 - The port to connect to. Copy the port number from the information message in the [Run](#) tool window that controls the running application.
- Click **OK**, when ready.

See Also

Concepts:

- [Running and Debugging](#)

Procedures:

- [NodeJS](#)
- [Running](#)
- [Debugging](#)
- [Creating and Editing Run/Debug Configurations](#)

Reference:

- [Run/Debug Configuration: Node JS](#)
- [Run/Debug Configuration: Node JS Remote Debug](#)
- [Run Tool Window](#)
- [Debug Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Unit Testing Node.JS

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts:  

PhpStorm supports integration with the **nodeunit** framework thus enabling running unit test for NodeJS applications.

To create and run unit tests for a nodejs application, perform these general steps

1. [Enable nodeunit support](#).
2. [Write the unit tests](#).
3. To have PhpStorm recognize the unit tests and detect the corresponding production source code, mark the folder where the unit tests are stored as [test folder](#).
4. [Create a run configuration](#) of the type **Nodeunit**.
5. [Launch unit tests](#) and [monitor test results](#) in the [Run](#) tool window.

This topic provides guidelines in NodeJS-specific unit testing procedures. For general information on testing in PhpStorm, see the section [Testing](#).

To enable unit testing for NodeJS applications

1. [Install](#) and [enable](#) the **NodeJS** plugin from the [JetBrains plugin repository](#).
2. Download and install the [NodeJS](#) framework.
3. Download and install the [nodeunit](#) testing framework

To create nodeunit tests, perform these general steps

- [Create a folder](#) `test` at the same level as the `src` folder
- Populate the `test` folder. For each production file, create a separate test file.
- Mark the folder where the tests are stored as [test folder](#).

To create a nodeunit run configuration

1. Open the [Run/Debug Configuration](#) dialog box by doing one of the following:
 - On the main menu, choose **Run | Edit Configurations**.
 - Press `RunConfigurationRunConfiguration` and choose **Edit Configuration** on the context menu.
2. Click the **Add** button  on the toolbar and select the **Nodeunit** configuration type.
3. In the dialog box that opens, specify the following:
 1. The name to identify the configuration.
 2. The path to the NodeJS installation to use.

If you have [appointed one of the installations as default](#), the field displays the path to its executable file.
 3. The working directory. This can be the project root folder or the parent directory for the `test` folder.
 4. The scope of tests to run.
 - To have PhpStorm run all the test files in a folder, choose **All JavaScript test files in the directory** from the **Run** drop-down list. In the **Directory** text box, type the path to the test folder relative to the [working directory](#).
 - To have a specific test executed, choose **JavaScript test file** from the **Run** drop-down list. In the **JavaScript test file** text box, type the path to the file relative to the [working directory](#).
4. Apply the changes and close the dialog box.

See Also

Concepts:

- [Testing](#)
- [NodeJS](#)
- [Creating and Editing Run/Debug Configurations](#)

Procedures:

- [Unit Testing JavaScript](#)

Reference:

- [Run/Debug Configuration: Nodeunit](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Reference

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

This part contains miscellaneous information related to the UI of the dialogs, tool windows and views, keyboard shortcuts, syntax references etc. :

- [Dialogs](#)
- [Settings Dialog](#)
- [Keyboard Shortcuts and Mouse Reference](#)
- [Tool Windows Reference](#)
- [Version Control Reference](#)
- [Table Editor](#)
- [Regular Expression Syntax Reference](#)
- [Scope Language Syntax Reference](#)
- [Diagram Reference](#)
- [Icons Reference](#)

See Also

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Dialogs

Previous | [Next](#) | See Also | [Comments](#)

This part contains descriptions of PhpStorm's dialogs:

- [Analyze Stacktrace Dialog](#)
- [Bookmarks Dialog](#)
- [Breakpoints](#)
- [Choose Local Paths to Upload Dialog](#)
- [Checkout from TFS Wizard](#)
- [Create New Project](#)
- [Code Duplication Analysis Settings](#)
- [Differences Viewer](#)
- [Differences Viewer for Folders and DB Objects](#)
- [Evaluate Expression](#)
- [Export Threads](#)
- [Export to HTML](#)
- [File Cache Conflict](#)
- [Find and Replace in Path](#)
- [Find Usages. Class Options](#)
- [Find Usages. Method Options](#)
- [Find Usages](#)
- [Find Usages. Variable Options](#)
- [Generate PHPUnit Test Dialog](#)
- [Generate Instance Document from Schema Dialog](#)
- [Generate Schema from Instance Document Dialog](#)
- [Login to IntelliJ Configuration Server Dialog](#)
- [Open Task Dialog](#)
- [New Project from Existing Files Wizard](#)
- [Optimize Imports Dialog](#)
- [Override Server Path Mappings Dialog](#)
- [Print](#)
- [Productivity Guide](#)
- [PSI Viewer](#)
- [Recent Changes Dialog](#)
- [Refactoring Dialogs](#)
- [Reformat Code Dialog](#)
- [Register New File Type Association Dialog](#)
- [Run/Debug Configurations](#)
- [Select Path](#)
- [Scopes](#)
- [Specify Code Duplication Analysis Scope](#)
- [Specify Inspection Scope Dialog](#)

See Also

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Analyze Stacktrace Dialog

Previous | [Next](#) | [See Also](#) | [Comments](#)

Tools | Analyze Stacktrace

Use this dialog box to reach a navigable console stack trace for external applications.

Item	Description
Unscramble stacktrace	Select this check box to unscramble the external stack trace, if your source code is scrambled.
Unscrambler	Here you can select the unscrambler tool.
	Note
	PhpStorm ships with the Zelix Klass Master unscrambler plugin. You can develop your own plugin to unscramble stack trace of the code being processed with any other obfuscator.
Log file	Specify the location of the unscrambler log file.
Put a stack trace or a complete thread	Paste here the external stack trace or thread dump.

- dump here
- Automatically detect and analyze thread dumps copied to the clipboard outside of PhpStorm. If this check box is selected, PhpStorm will monitor and analyze the contents of the clipboard. As soon as something looking like a stack trace gets copied to the clipboard, PhpStorm will show this stack trace in the corresponding tool window.
- Normalize. If the stack trace text is corrupted (lines are cut or wrapped, or too long, etc.) after processing with some software (for example, bug tracker or mail client), click this button to restore the normal stack trace structure.

See Also

Procedures:

- [Analyzing External Stacktraces](#)
- [Navigating from Stacktrace to Source Code](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Bookmarks Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Edit | Show Bookmarks
Shift+F11Shift F11

Use this dialog to navigate between bookmarks, and manage the collection of anonymous bookmarks and bookmarks with mnemonics in a project.

Toolbar options

Item	Shortcut	Description
	Ctrl+EnterCommand Enter	Click to add/edit description for the selected bookmark.
	EditorDeleteEditorDelete	Click to delete the selected bookmark.
	Ctrl+UpMeta Up / Ctrl+DownMeta Down	Use this buttons to reorder bookmarks.

The left pane of the **Bookmarks** dialog displays the list of bookmarks created with the brief description, if any. The order of bookmarks in the collection defines the order in which **Navigate | Next/Previous Bookmark** command visits the bookmarks. The right pane displays the preview of the file where the selected bookmark is toggled.

See Also

Procedures:

- [Navigating with Bookmarks](#)
- [Managing Bookmarks](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Breakpoints

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Run | View Breakpoints

Ctrl+Shift+F8Command Shift F8



Run | View Breakpoints

Ctrl+Shift+F8Command Shift F8



Use this dialog to:

- Add [breakpoints](#) of [various types](#) to your code.
- Configure behavior of the breakpoints in your source code.
- Remove breakpoints.

Warning

If you haven't set any breakpoints in your code, behavior options are hidden. To configure them, you should first create a breakpoint. Options are displayed for the currently selected breakpoint only.

The dialog box consists of tabs corresponding to the [types of breakpoints](#) and the contexts in which breakpoints are set. The controls for all types of breakpoints are similar; differences are specially noted.

See Also

Concepts:

- [Breakpoints](#)

Procedures:

- [Using Breakpoints](#)

Reference:

- [Class Filters Dialog](#)
- [New Filter Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Choose Local Paths to Upload Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

The dialog box opens when you click the **Upload to Web Server** check box in the [Run/Debug Configuration](#) dialog box and click the **Browse** button  next to it.

Use the dialog box to configure a list of folders and files to be uploaded according to previously defined mappings and to edit these mappings, if necessary.

Item	Description
Add	Click this button to have a new row added to the list and specify the item to upload. Type the path manually or click the Browse button  to select the desired location in the Choose Path dialog box, that opens.
Remove	Click this button to remove the selected entry from the list.
Edit Mappings	Click this button to open the Deployment dialog box, where you can define mappings between local folders and the corresponding paths on the FTP/SFTP server.

See Also

Reference:

- [Deployment](#)
- [Run/Debug Configurations](#)
- [Remote Host Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Checkout from TFS Wizard

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

VCS | [Checkout from Version Control](#) | TFS

VCS | [Checkout from Version Control](#) | TFS

Use this wizard to download the files from a TFS server according to the settings from a new or an existing [workspace](#).

Note

The menu item and the wizard are available only when the **TFS Integration** bundled plugin is [enabled](#).

In this part:

- [Checkout from TFS Wizard: Checkout Mode](#)
- [Checkout from TFS Wizard: Source Server](#)
- [Checkout from TFS Wizard: Choose Source and Destination Paths](#)
- [Checkout from TFS Wizard: Source Workspace](#)
- [Checkout from TFS Wizard: Choose Source Path](#)
- [Checkout from TFS Wizard: Summary](#)

See Also

Procedures:

- [Using TFS Integration](#)
- [Version Control with PhpStorm](#)

Reference:

- [TFS](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Checkout from TFS Wizard: Checkout Mode

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

VCS | [Checkout from Version Control](#) | TFS

VCS | [Checkout from Version Control](#) | TFS

On this page, choose the way to map the files and folders on your server with their local copies. These mappings are referred to as [workspaces](#). You can either specify new mappings and have a new workspace generated or use mappings from an existing workspace.

Item	Description
Create workspace automatically	Choose this option to have PhpStorm generate a workspace for you, with the folder mapping based on your choice of the local and remote paths .
Workspace name	In this text box, type the name of the workspace to be generated.
Choose workspace manually	Choose this option to have the data downloaded according to the mappings from an existing workspace.

See Also

Reference:

- [TFS](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Checkout from TFS Wizard: Source Server

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

VCS | [Checkout from Version Control](#) | TFS - [Create workspace automatically](#)

VCS | [Checkout from Version Control](#) | TFS - [Create workspace automatically](#)

On this page, choose the server to download the data from.

Item	Description
Team servers	In this list box, manage the list of the servers you have access to and select the server to download the data from.
Add	Click this button to open the Add Team Foundation Server dialog box, and specify the address of your TFS server and the credentials to connect to it.
Remove	Click this button to delete the selected server from the list.
TFS Proxy	Click this button to open the Set TFS proxy for server... dialog box where you can specify the parameters for accessing the selected server via Proxy.

See Also

Reference:

- [TFS](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Checkout from TFS Wizard: Choose Source and Destination Paths

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

VCS | [Checkout from Version Control](#) | TFS - [Create workspace automatically](#)

VCS | [Checkout from Version Control](#) | TFS - [Create workspace automatically](#)

The page opens when you select the server to download the data from on the [Source Server](#) page and click [Next](#).

On this page, specify the remote folder to download the data from and map it to a local folder where the data will be downloaded to. The data is downloaded recursively, that is, the structure of subfolders under the selected source node is reproduced locally.

Item	Description
Source Path	In this area, specify the folder on the server to download the data from. Select the folder in the tree, and Source Path read-only field displays the path to it relative to the server root.
Destination path	In this text box, specify the local folder to download the sources to. Type the path to the folder manually or click the Browse button  and select the folder in the dialog box, that opens.

Note

The data is downloaded recursively, that is, the structure of subfolders under the source node is repeated locally.

See Also

Reference:

- [TFS](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Checkout from TFS Wizard: Source Workspace

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

VCS | [Checkout from Version Control](#) | [TFS - Choose workspace manually](#)

VCS | [Checkout from Version Control](#) | [TFS - Choose workspace manually](#)

On this page, specify the workspace to add the imported files and folders to. The data will be downloaded according to the mappings defined in the selected workspace. The data is downloaded recursively, that is, the structure of subfolders under the selected source node is reproduced locally.

You can use an existing workspace as is, or update it as necessary, or even create an entirely new workspace manually.

Item	Description
Server/Workspace	This read-only field shows the URL addresses of TFS servers you have access to and workspaces available on these TFS servers.
Workspace comment	This read-only field shows the descriptions of workspaces on the servers you have access to.
Team Servers	Use the buttons in this area to manage the list of available servers and workspaces and configure access to them. <ul style="list-style-type: none"> • Add: click this button to open the Add Team Foundation Server dialog box where you can specify the parameters for establishing connection to a TFS server. • Remove: click this button to remove the selected server from the list. • Reload workspaces: click this button to have the list of available workspaces refreshed. • TFS Proxy: click this button to open the Set TFS proxy for server... dialog box where you can specify the parameters for accessing the selected server via Proxy. • Check-in Policies: click this button to open the Edit Check-in Policies dialog box where you can manage the list of check-in policies to be applied.
Workspaces	Use the buttons in this area manage the list of available workspaces and update the workspaces, when applicable. <ul style="list-style-type: none"> • Create: click this button to open the Create Workspace dialog box for creating a new workspace. • Edit: click this button to open the Edit Workspace dialog box for editing the selected workspace. • Delete: click this button to remove the selected workspace from the list.

See Also

Reference:

- [TFS](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Checkout from TFS Wizard: Choose Source Path

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

VCS | [Checkout from Version Control](#) | [TFS - Choose workspace manually](#)

VCS | [Checkout from Version Control](#) | [TFS - Choose workspace manually](#)

The page opens when you choose an existing workspace on the [Source Workspace](#) page and click Next.

On this page, specify the folder on the server to download data from.

The page shows a tree of folders on the server that you have access to. When you select the required folder, \$product\$ tells you which local folder it is mapped to in the [current workspace](#).

See Also

Reference:

- [TFS](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Checkout from TFS Wizard: Summary

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

VCS | [Checkout from Version Control](#) | [TFS - Create workspace automatically](#)

VCS | Checkout from Version Control | TFS - Choose workspace manually

VCS | Checkout from Version Control | TFS - Create workspace automatically

VCS | Checkout from Version Control | TFS - Choose workspace manually

This last, read-only, page of the wizard shows the name of the workspace to be used or generated, the URL address of the server, the source folder to download data from, and the local folder to save the downloaded data in. Review the details and click Finish.

See Also

Reference:

- [TFS](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Create New Project

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

File | New Project

Welcome Screen | Create New Project

Use this dialog box to create a new project from scratch.

Item	Description
Project name	In this text box, type the name of the new project.
Location	In this text box, specify the location of the folder where the project-related files will be stored. Type the path manually or click the Browse button  and select the folder where the project will be created.
OK	Click this button to create a new project.
Cancel	Click this button to cancel operation and close the dialog box.

See Also

Getting Started:

- [Create and Run Your First PHP Project](#)
- [Create and Run Your First Web Project](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Code Duplication Analysis Settings

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Code | Locate Duplicates | OK

In this dialog can define the sensitivity of search, and set limitation that will help you avoid reporting about every similar code construct.

Use the controls of this area to specify your preferences in searching for duplicates in language-specific context.

Item	Description
CSS	<ul style="list-style-type: none"> • Do not show duplicates containing less than <number> properties: select this check box to set the minimum number of properties in a duplicate to have the duplicate shown.
CoffeeScript	<ul style="list-style-type: none"> • Anonymize Variables • Anonymize Functions • Anonymize Literals
ECMA Script level 4	<ul style="list-style-type: none"> • Anonymize Variables • Anonymize Functions • Anonymize Literals
HTML	<ul style="list-style-type: none"> • Do not show duplicates containing less than <number> tags • Anonymize values of tags and attributes
JSON	<ul style="list-style-type: none"> • Anonymize Variables • Anonymize Functions • Anonymize Literals

- JavaScript
 - Anonymize Variables
 - Anonymize Functions
 - Anonymize Literals

- PHP
 - Anonymize Variables
 - Anonymize Functions
 - Anonymize Literals

- XHTML
 - Anonymize values of tags and attributes

- XML
 - Anonymize values of tags and attributes

See Also

Procedures:

- [Analyzing Duplicates](#)

Reference:

- [Duplicates Tool Window](#)
- [Specify Code Duplication Analysis Scope](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Differences Viewer

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

This dialog is displayed every time you explore differences between files (local history for folders, recent changes, version control, local files and their versions on a remote host, or just comparing files). So doing, you can compare files of any types, including binaries.

Tip

You can also open the difference viewer without running PhpStorm. This is done through the following command:

```
<path to PhpStorm executable file> diff <path_1> <path_2>
```

where path_1 and path_2 are paths to the files in question, which can be of various types, including jar.

The differences viewer provides a powerful editor that enables code completion, live templates etc.

Item	Tooltip and Shortcut	Description
	Ctrl+C Command C or Ctrl+Insert Command Insert	Click this button to copy the line at caret in the right pane to the Clipboard.
	Ctrl+F Command F or Alt+F3 Alt F3	Click this button to initiate Finding text procedure and locate the specified string in the current file.
	F7F7 / Shift+F7Shift F7	Click this button to move to next / previous difference.
Ignore whitespace		Use this drop-down list to define how the differences viewer should treat white spaces in the text. <ul style="list-style-type: none"> • Do not ignore: white spaces are important, and the differences should be highlighted. • Leading and Trailing: Ignore the differences, if they appear in the end and in the beginning of a line. • All: white spaces are not important, regardless of their location in the source code.
	Compare Previous File Alt+LeftCommand Left	Click this button to compare the local copy of the previous file with its update from the server. Note The button is available only when the Differences Viewer is invoked from the Update Project Info tab of the Version Control tool window or from the Apply Patch dialog box.
	Compare Next File Alt+RightCommand Right	Click this button to compare the local copy of the next file with its update from the server. Note The button is available only when the Differences Viewer is invoked from the Update Project Info tab of the Version Control tool window or from the Apply Patch dialog box.

Legend Shows summary information about the encountered differences: number of differences found, and color map.

Tip

Color map for the Differences viewer is configurable in the [Colors and Fonts](#) dialog box.



Use these *chevron* buttons to apply differences between panes.

See Also

Concepts:

- [Local History](#)

Procedures:

- [Viewing Local History of a File or Folder](#)
- [Comparing Deployed Files and Folders with Their Local Versions](#)

Reference:

- [Show History for File / Selection Dialog](#)
- [Recent Changes Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Differences Viewer for Folders and DB Objects

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

[Project tool window](#) | [context menu of a folder](#) | [Compare Directory with](#)

[Project tool window](#) | [context menu of a folder](#) | [Sync with Deployed to](#)

[Remote Host tool window](#) | [context menu of a folder](#) | [Sync with local](#)

[Tools](#) | [Deployment](#) | [Sync with local](#)

[Data Sources tool window](#) | [context menu of two selected items](#) | [Compare](#)

[Project tool window](#) | [context menu of a folder](#) | [Compare Directory with](#)

[Project tool window](#) | [context menu of a folder](#) | [Sync with Deployed to](#)

[Remote Host tool window](#) | [context menu of a folder](#) | [Sync with local](#)

[Tools](#) | [Deployment](#) | [Sync with local](#)

[Data Sources tool window](#) | [context menu of two selected items](#) | [Compare](#)

This window is displayed every time you explore differences between:

- [Two local directories](#) (local history for folders, recent changes, or version control).
- [A remote folder and its local version.](#)
- [Two database objects](#) (data sources, schemas, tables or columns).

In this dialog box, explore the detected differences and synchronize the compared items.

Tip

You can also open the difference viewer without running PhpStorm. This is done through the following command: `<path to PhpStorm executable file> diff <path_1> <path_2>` where `path_1` and `path_2` are paths to the folders in question.

In this section:

- [Toolbar](#)
- [List of Items](#)
- [Differences Pane](#)

Toolbar

Item	Tooltip and shortcut	Description	Available for
	Refresh Ctrl+F5Command F5	Click this button to refresh the contents of the differences viewer.	All
	Show new on left side	Press this toggle button to have PhpStorm show the items, that are present in the first of the compared directories or database objects and are missing in the second one.	All
	Show new on right side	Press this toggle button to have PhpStorm show the items, that are present in the second of the compared directories or database objects and are missing in the first one.	All
	Show difference	Press this toggle button to have PhpStorm show the items that are present in both directories or objects and have the same contents, or timestamp, or size, depending on the parameter for comparison specified in the Compare by drop-down list.	All
	Show equals	Press this toggle button to have PhpStorm show items that are present in both folders or database objects but differ in contents, or timestamp, or size.	All

Compare by	<p>In this drop-down list, select the parameter to be used for comparison. The available options are:</p> <ul style="list-style-type: none"> • Contents • Size • Time stamp 	Local folders Local-remote folders
	<p>Synchronize Selected EnterEnter</p> <p>Click this button to have PhpStorm apply the specified action to the selected pair of items. Actions to be performed are shown in the * field.</p>	All
	<p>Synchronize All Ctrl+EnterCommand Enter</p> <p>Click this button to have PhpStorm apply the specified action to all the pairs of items in the list. Actions to be performed are shown in the * field.</p>	All
	<p>Hide excluded files</p> <p>Click this button to suppress showing files excluded from synchronization.</p>	Local-remote folders
Filter	<p>Type the filtering string (for example, file or table name). Use * wildcard to replace any number of arbitrary characters.</p> <p>Note that filter applies on pressing EnterEnter.</p>	All
Path	<p>These fields show the paths to the folders being compared. To change a directory, click the Browse button  and specify another directory in the Select Path dialog box, that opens.</p> <p>These read-only fields show the names of the data sources or tables being compared.</p>	Local folders Local-remote folders Data sources

List of items

The list shows the items from the compared objects that meet the comparison criterion specified in the [Compare by](#) drop-down list and the filtering criteria specified through the toolbar buttons.

Item	Description	Available for
Name	These read-only fields show the names of files, data source tables, or table fields under the object specified in the Path or  fields.	All
Size	These read-only fields show the sizes of files under the folders being compared.	Local folders Local-remote folders
Date	These read-only fields show the timestamps of files under the folders being compared.	Local folders Local-remote folders
*	<p>The icon in this field indicates the action that will be applied to the pair of items in the current line upon clicking the Synchronize Selected  or Synchronize All  toolbar button.</p> <p>To change the currently selected action, click the icon.</p> <p>Icon Action</p> <ul style="list-style-type: none">  Copy the item in the left side to the right side, possibly overwriting the contents of the corresponding target item, if it already exists.  Copy the item in the right side to the left side, possibly overwriting the contents of the corresponding target item, if it already exists.  The items are treated identical with regard to the selected criterion of comparison. No action will be performed by default.  The items differ with regard to the selected criterion of comparison. No action will be performed by default. Explore the differences in the Differences Pane and change the intended action by clicking the icon.  The item is present only in one of the folders and will be removed. 	All

Differences pane

Note

The differences pane only shows for files, data source tables, or table fields with the same names, which exist on both sides. For the files and DB objects that exist on one side only, the contents of the selected file/DB object is displayed.

If the files have read-only status, they are not editable in the differences pane.

Item	Description
	Click this button to place to the clipboard the fragment of code selected in the preview pane. If nothing is selected, the whole line at caret is copied.
	Click this button to initiate the find in text procedure in the differences viewer.
 F7F7 / Shift+F7Shift F7	Click this button to move to the next / previous difference.
Ignore whitespace	<p>Use this drop-down list to define how the differences viewer should treat white spaces in the text.</p> <ul style="list-style-type: none"> • Do not ignore: white spaces are important, and the differences should be highlighted. • Leading and trailing: Ignore the differences, if they appear in the end and in the beginning of a line. • All: white spaces are not important, regardless of their location in the source code.
	Use these <i>chevron</i> buttons to apply differences between files.

See Also

Procedures:

- [Comparing Folders](#)
- [Viewing Local History of a File or Folder](#)
- [Comparing Data Sources](#)

- [Comparing Deployed Files and Folders with Their Local Versions](#)

Reference:

- [Show History for Folder Dialog](#)
- [Recent Changes Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Evaluate Expression

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Run | Evaluate Expression

Alt+F8Alt F8

Use this dialog box to calculate values of expressions or code fragments during the debugging session.

Item	Description
Expression	Use this field to edit the expression to be evaluated. If an expression is selected in the editor, this field displays selection. Tip This field is available in the Expression Mode .
Statements to Evaluate	Type the group of statements to be evaluated. If a code fragment is selected in the editor, this field displays selection. Tip This field is available in the Code Fragment Mode .
Result	Here the results are displayed.
Evaluate	Click this button to evaluate the current expression or code fragment.
Close	Click this button to close the dialog box.
Code Fragment Mode	Click this button to toggle to the Code Fragment Mode .
Expression Mode	Click this button to toggle to the Expression Mode .

See Also

Procedures:

- [Evaluating Expressions](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Export Threads

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Run | Export Thread

Debug tool window | Threads | context menu | Export Threads

Alt+U+HALt U H

Use this dialog to export a thread to the specified text file.

Item	Description
Export to file	Type the target file to store the thread report.
	Use this button to navigate to the desired location in the Select Path dialog.
Save	Click to save the current threads in a text file indicated in the Export to file field.
Copy	Click to copy the thread status to the Clipboard.
Cancel	Use to discard all changes and close the dialog.

See Also

Procedures:

- [Examining Suspended Program](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Export to HTML

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

File | Export to HTML

Use this dialog to save selected files in HTML format.

Item	Description
File <name>	Click this radio button to print the file currently selected in the Project view, or open in the editor.
Selected text	Click this radio button to print the text selected in the editor.
All files in the directory	Click this radio button to print all files in the current directory.
Include subdirectories	Check this option to print the files in the subdirectories of the current directory.
Output directory	Specify the fully qualified path to the directory, where the resulting HTML file will be stored.
Show line numbers	Check this option to include line numbers in the resulting HTML file.
Generate hyperlinks to classes	Check this option to replace class names with the hyperlinks to the respective classes.
Open generated HTML in browser	Check this option to show HTML file in the default browser after export.

See Also

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

File Cache Conflict

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

An external process has changed a file, opened and unsaved in PhpStorm, which results in two conflicting versions of a file. Resolve this conflict using the following options.

Item	Description
Load FS Changes	Click this button to load the file version produced outside of PhpStorm, and overwrite your local changes.
Keep Memory Changes	Click this button to preserve the version produced in PhpStorm and stored in cache.
Show Difference	Click this button to invoke the differences viewer that shows the version in the file system to the left, and PhpStorm version to the right. You can merge differences as required and load the desired version to PhpStorm. By default, the cache version is saved.

See Also

Procedures:

- [Saving and Reverting Changes](#)

Reference:

- [General](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Find and Replace in Path

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts:  

Edit | Find | Find in Path, Replace in Path

Ctrl+Shift+FCommand Shift F

Edit | Find | Find in Path, Replace in Path

Ctrl+Shift+FCommand Shift F

Use these dialogs to find and replace a string in the specified scope. You can perform search taking into account case sensitivity and multiple file masks, and use regular expressions.

Item	Description
Text to find	In this field, specify the search pattern. Type the text manually or select one of the previously specified patterns from the drop-down list.
Tip	If you specify the search pattern through a regular expression, use the $\$n$ format in back references (to refer to a previously found and saved pattern).

Replace with	In this field, specify the replacement text. Type the text manually or select one of the previously specified substitutions from the drop-down list.
	<p>Note</p> <ul style="list-style-type: none"> If you specify the replacement text through a regular expression, use the $\\$n$ format in back references. This field is available only in the Replace in Path dialog.
Options	In this area, specify additional search parameters. The available options are: <ul style="list-style-type: none"> Case sensitive - select this check box to have PhpStorm distinguish between upper and lowercase letters while searching. Preserve case - if this check box is selected, PhpStorm retains the case of the first letter and the case of the initial string in general. For example, <i>MyTest</i> will be replaced with <i>YourTest</i> if you specify <i>yourTest</i> as the replacement. <p>Note</p> <p>This check box is disabled, if the Case sensitive or Regular expressions check box is selected.</p> <ul style="list-style-type: none"> Whole words only - select this check box to have PhpStorm search for whole words only, that is, for character strings separated with spaces, tabs, punctuation, or special characters. <p>Note</p> <p>This check box is disabled, if the Regular expressions check box is selected.</p> <ul style="list-style-type: none"> Regular expressions - select this check box to have PhpStorm consider the specified search pattern a regular expression. <p>Tip</p> <p>Click the [Help] link next to the check box to view reference on regular expressions syntax.</p>
Scope	In this area, specify the scope to apply the search to. The available options are: <ul style="list-style-type: none"> Whole project - select this option to search through the entire project. Directory - select this option to perform search within the specified directory. By default, the text area already contains the directory name where a file currently opened in the editor is located (if you call the dialog from the editor), or where a file selected in the tool window is located (if you call the dialog from the tool window), or the directory name selected in the tool window. Pressing the ellipsis button calls the Select Path dialog where you can select a directory to make a search. Recursively - this check box is only available for the directory search. If selected, sets the search to be performed in the chosen directory and its subdirectories. Custom - select this option to search in a scope. You can choose one of the scopes from the drop-down list, or click the ellipsis button, and configure the desired one in the Scopes dialog box.
File name filter	In this area, specify additional settings to narrow down the search scope. <ul style="list-style-type: none"> File mask - select this check box to narrow down the search scope through file masks. In the drop-down list, select the desired mask or specify a new one using wildcards. Wildcard can include * to substitute a set of any characters, and ? to substitute a single character. Note that you can specify multiple file masks, delimited with comma (for example, *.xml, *.sql, *.html).
Skip results tab with one usage	Select this check box to be navigated directly to the found string in the editor, when only one usage is found.
	<p>Note</p> <p>The check box is available only in the Find in Path dialog box.</p>
Open in new tab	If selected, sets the search results to be displayed in a separate tab.
	<p>Note</p> <p>The check box is available only in the Find in Path dialog and only if the Find tool window is already opened.</p>

See Also

Procedures:

- [Finding and Replacing Text in Project](#)

Reference:

- [Replace Text](#)
- [Regular Expression Syntax Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
 - <http://youtrack.jetbrains.net/issues/WI> 
-

Find Usages. Class Options

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: 

Edit | Find | Find Usages

Alt+F7Alt F7

Edit | Find | Find Usages

Alt+F7Alt F7

This section describes the controls for specifying Class Usage Search and Interface Usage Search options in the **Find Usages** dialog box.

Item	Description
Scope	In this area, specify the scope of search. Select a pre-defined scope from the drop-down list or click the Browse button  to open the Scopes dialog box, where you can define a custom scope.
Open in new tab	Select this check box to have the results of each search shown in a separate tab of the Find Results window. If the check box is cleared, the search results will overwrite the contents of the current tab.

See Also

Procedures:

- [Finding Usages in Project](#)
- [Working with Search Results](#)

Reference:

- [Find Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Find Usages. Method Options

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: 

Edit | Find | Find Usages

Alt+F7Alt F7

Edit | Find | Find Usages

Alt+F7Alt F7

This section describes the controls for specifying Method Usage Search options in the **Find Usages** dialog box.

Item	Description
Search for text occurrences	Select this check box to have text contents and comments involved in searching.
Skip results tab with one usage	Select this check box to be navigated directly to the found usage without the Find tool window displayed, when only one usage is found.
Scope	In this area, specify the scope of search. Select a pre-defined scope from the drop-down list or click the Browse button  to open the Scopes dialog box, where you can define a custom scope.
Open in new tab	Select this check box have the results of each search shown in a separate tab of the Find Results window. If the check box is cleared, the search results will overwrite the contents of the current tab.

See Also

Procedures:

- [Finding Usages in Project](#)
- [Working with Search Results](#)

Reference:

- [Find Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Find Usages

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: 

Edit | Find | Find Usages

Alt+F7Alt F7

Edit | Find | Find Usages

Alt+F7Alt F7

Use this dialog box to find usages of classes, tags, attributes, and references in the HTML, XML, and CSS files, . This dialog also allows to find usages of the data sources, tables, and columns.

Search results are displayed in the [Find tool window](#).

Item	Description
Skip results tab with one usage	Select this check box to be navigated directly to the found usage without the Find tool window displayed, when only one usage is found.
Scope	In this area, specify the scope of search. Select a pre-defined scope from the drop-down list or click the Browse button  to open the Scopes dialog box, where you can define a custom scope.

See Also

Procedures:

- [Finding Usages in Project](#)
- [Working with Search Results](#)

Reference:

- [Find Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Find Usages. Variable Options

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac 

Edit | Find | Find Usages

Alt+F7Alt F7

Edit | Find | Find Usages

Alt+F7Alt F7

This section describes Field, Variable, or Parameter usage search options in the Find Usages dialog.

Item	Description
Skip results tab with one usage	Select this check box to be navigated directly to the found usage without the Find tool window displayed, when only one usage is found.
Scope	In this area, specify the scope of search. Select a pre-defined scope from the drop-down list or click the Browse button  to open the Scopes dialog box, where you can define a custom scope.
Open in new tab	Select this check box have the results of each search shown in a separate tab of the Find Results window. If the check box is cleared, the search results will overwrite the contents of the current tab.

See Also

Procedures:

- [Finding Usages in Project](#)
- [Working with Search Results](#)

Reference:

- [Find Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Generate PHPUnit Test Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Project tool window | context menu of a PHP class | Generate PHPUnit Test

Use this dialog box to specify the name and location of a PHPUnit test generated on the basis of an existing PHP class.

Item	Description
Class	This read-only field shows the name of the selected PHP class to generate a unit test for.
Output Directory	In this text box, specify the path to the directory, where the generated unit test class will be stored. Type the path manually or click the Browse button  and select the desired folder in the Choose Test Directory dialog box that opens.
Output File Name	In this text box, specify the name of the file for the generated test class. By default, PhpStorm suggests a name of the following structure: <selected PHP file name>.php.
Create run configuration	Select this check box to have PhpStorm create a run configuration to execute the generated test.

See Also

Procedures:

- [Generating PHPUnit Test Class](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Generate Instance Document from Schema Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Tools | XML Actions | Generate XML Document from XSD Schema

In this dialog box, specify the options for generating an XML file based on the XSD (XML Schema Definition) [Schema](#) that describes its structure.

Note

The menu item and the dialog box are available when the file opened in the active editor tab contains an XML Schema.

Item	Description
Schema path	In this field, specify the location of the <code>.xsd</code> Schema file to be used as the basis for XML document generation. By default, the field shows the full path to the current file. To use another Schema, click the Browse button  and select the desired file in the Select Path dialog box, that opens.
Instance document name	In this text box, specify the name of the output XML file to be generated. By default, the generated XML file will inherit the name of the source Schema and will have the <code>.xml</code> extension. If you type another name for the document to be generated, make sure the extension is correct. The default location for the generated document is the directory where the source <code>.xsd</code> Schema file resides. To specify another location, click the Browse button  and select the desired path in the Select Path dialog box, that opens.
Element name	In this drop-down list, specify the local name of the global element to be used as the root of the generated document.
Enable restriction check	Select this check box to have PhpStorm take restriction particles into consideration (if the specified Schema uses any).
Enable unique check	Select this check box to have PhpStorm take uniqueness particles into consideration (if the specified Schema uses any).
Status	View the information in this read-only field to track and improve discrepancies when configuring the generation procedure.

See Also

Concepts:

- [Markup Languages and Style Sheets](#)

Procedures:

- [Generating Instance Document from XML Schema](#)
- [Generating XML Schema from Instance Document](#)
- [Markup Languages and Style Sheets](#)

Reference:

- [Generate Schema from Instance Document Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Generate Schema from Instance Document Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Tools | XML Actions | Generate XSD Schema from XML File

In this dialog box, specify the options for generating an XSD (XML Schema Definition) [Schema](#) that describes the structure of the desired XML file.

Note

The menu item and the dialog box are available when an XML document is opened in the active editor tab.

Item	Description
Instance document path	In this text box, specify the full path to the XML document to be used as the basis for Schema generation. By default, the field shows the full path to the current file. To use another XML document, click the Browse button  and select the desired file in the Select Path dialog box, that opens.
Result Schema file name	In this text box, specify the name of the output file for the Schema to be generated. By default, the generated XSD Schema file will inherit the name of the instance document used and will have the <code>.xsd</code> extension. If you type another name for the Schema to be generated, make sure the extension is correct. The default location for the generated Schema is the directory where the source XML instance document resides. To specify another location, click the Browse button  and select the desired path in the Select Path dialog box, that opens.
Design type	Use this drop-down to specify how to declare elements and complex types. The available options are: <ul style="list-style-type: none"> • Local elements / global complex types

- Local elements / types
- Global elements / local types

Detect simple content types From this drop-down list, choose the type to render leaf text, which should be distinguished from the text used. The available options are:

- String
- Smart

Detect enumeration limit In this text box, type the number of occurrences to cause the Schema enumeration appearance. To suppress Schema enumeration, specify 0.

See Also

Concepts:

- [Markup Languages and Style Sheets](#)

Procedures:

- [Generating XML Schema from Instance Document](#)
- [Validating Web Content Files](#)
- [Markup Languages and Style Sheets](#)

Reference:

- [Generate Schema from Instance Document Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Login to IntelliJ Configuration Server Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

IntelliJ Configuration Server is intended for storing common IDE settings to be used by a team. This ensures that all the team members have unified IDE look and feel.

The dialog box opens:

- When you start up the PhpStorm for the first time. In this case, the dialog box contains an additional area [On next startup](#).
- When you click the **IntelliJ Configuration Server Status** button  on the Status bar during a session and then click the **Login** button in the dialog box that opens.

Use this dialog box to log in to the IntelliJ Configuration Server and specify the Proxy settings to access the Server, if necessary.

Item	Description
Login	In this text box, type your JetBrains Account login.
Password	In this text box, type your JetBrains account password.
Create or manage your JetBrains account	Click this link to create a JetBrains Account or edit your account if you already have it.
Use HTTP Proxy	Select this check box if you want to access the IntelliJ Configuration Server through a Proxy server.
Host name	In this text box, type the Proxy host name.
Port Number	In this text box, type the Proxy port number.
Proxy Authentication	Select this check box if you want to use a login and password to access the HTTP Proxy server.
Login	In this text box, type your Proxy login.
Password	In this text box, type your Proxy password.
Remember password	If this check box is selected, PhpStorm remembers your Proxy password and in the future fills in the Password text box automatically when you log in.

On next startup area

The area is displayed in the dialog box only when PhpStorm starts up for the first time. Use this area to configure how you want to log in to IntelliJ Configuration Server in the future.

The following controls are available:

Item	Description
Show login dialog	Select this option to have the Login to IntelliJ Configuration Server dialog box displayed during the next PhpStorm startup.
Login silently	Select this option to log in to the Server without asking for login and password.
Do not login	Select this option if you do not plan to log in to the IntelliJ Configuration Server.

Note

You can reconfigure this behavior at any time [during the session](#).

Login	Click this button to log in to the IntelliJ Configuration Server.
Skip	Click this button to start an PhpStorm session without logging to the IntelliJ Configuration Server.

Note

You can log in to the Server at any time [during the session](#).

See Also

Concepts:

- [Project and IDE Settings](#)

Procedures:

- [Sharing Settings](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Open Task Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Tools | Task | Open

Tools | Task | Switch - Open New Task

Alt+Shift+NAlt Shift N

Use this dialog box to assign a name to a new task or specify the name of an existing task to open. Optionally, specify whether you want PhpStorm to create a change list for the created or selected and have the current context cleared.

Item	Description
Enter task name or choose existing task	In this text box, type the name of the new task or the name of an existing task to open.
Configure	Click this link to open the Servers page of the Settings dialog box, where
Clear current context	Select this check box to have the context associated with the current task cleared when the new/selected task is opened.
Create changelist	Select this check box to have PhpStorm create a new changelist for the specified task.

See Also

Procedures:

- [Managing Tasks and Context](#)

Reference:

- [Servers](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

New Project from Existing Files Wizard

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

File | New Project From Existing Files

Use this Wizard to set up new projects around existing files that reside locally or on remote hosts.

The wizard gives you a way to process specific files from a distributed project without downloading or copying it entirely. PhpStorm creates a project directory in the specified location with the `.idea` folder that contains the [project files](#).

Choose your scenario

On the first page of the Wizard, choose the way to access the files around which a new project will be set up.

Item	Description
Web server is installed locally, source files are located under its document root .	Select this option if you have a local Web server and you want to set up a project around existing files or folders that are located on this server below the folder appointed as the server document root, for example below <code>.\\htdocs</code> .
Web server is installed locally , source files are located elsewhere locally.	
Web server is on a remote host, files are accessible via network share or mounted drive .	Select this option to have PhpStorm copy files to a local drive via network and set up a project around them.
Web server is on a remote host, files are accessible via FTP/SFTP .	Select this option to have PhpStorm download files from a remote server via the FTP or SFTP protocol.
Source files are in a local directory, no Web server is yet configured .	Select this option if you want to work with files in a certain local directory without using any Web server.

See Also

Concepts:

- [Project](#)

Procedures:

- [Creating a Project from Downloaded Files](#)
- [Creating and Managing Projects](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Create New Project: Choose Project Directory

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

File | New Project From Existing Files - Web server is installed locally, source files are located under its document root
File | New Project From Existing Files - Source files are in a local directory, no Web server is yet configured

On this page of the wizard, specify the location of the files to set up the project around.

See Also

Concepts:

- [Project](#)

Procedures:

- [Creating and Managing Projects](#)
- [Creating a Project from Downloaded Files](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Create New Project: Specify Local Path

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

File | New Project From Existing Files - Web server is installed locally, source files are located elsewhere locally
File | New Project From Existing Files - Web server is on a remote host, files are accessible via network share or mounted drive
File | New Project From Existing Files - Web server is on a remote host, files are accessible via FTP/SFTP

On this page, specify the name of the project to be set up and the local folder to download the project sources to.

Item	Description
Project name	In this text box, type the name of the project to be set up based on the downloaded files.
Project local path	In this text box, specify the local folder where the new project will be set up. First specify the parent folder where the project folder will be created. Type the path manually or click the Browse button  and select the desired location in the Select Path dialog box that opens. As you type the project name, it is added to the project path.
Deployment options	In this area, specify the download configuration to use. The available options are: <ul style="list-style-type: none"> • Default - click this option to have PhpStorm download the sources according to the settings specified in the Options dialog box on the Deployment page. • Custom - click this option to customize the default settings in the Create New Project: Review Deployment Options dialog box that opens when you click Next.

See Also

Concepts:

- [Project](#)

Procedures:

- [Creating and Managing Projects](#)

Reference:

- [Deployment](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Create New Project: Review Deployment Options

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

- [File | New Project From Existing Files - Web server is installed locally, source files are located elsewhere locally](#)
- [File | New Project From Existing Files - Web server is on a remote host, files are accessible via network share or mounted drive](#)
- [File | New Project From Existing Files - Web server is on a remote host, files are accessible via FTP/SFTP](#)

The page opens upon clicking Next on the [Create New Project: Specify Local Path](#) page of the wizard if you have selected the Custom option in the Deployment options area. Use this page to customize the [default deployment procedure](#) to the current download.

Item	Description
Exclude items by name	In this text box, specify patterns for files to be excluded from download. Use semicolons as delimiters. Wildcards are welcome.
Stop transfer on the first error	Select this check box to have data transfer stopped as soon as an error occurs.
Overwrite up-to-date files	<ul style="list-style-type: none"> Select this check box to have all the files downloaded no matter whether local files with the same names have been changed since the previous download or not. Clear the check box to download only files that have not been changed since the previous download.
Preserve files timestamps	Select this check box to prevent resetting timestamps of files on download.
Delete target item if the source one is missing	If this check box is selected, a previously downloaded file will be removed if the file with this name is not involved in the current download.
Create empty directories	Select this check box to have an empty directory on the server created automatically if a new local directory has been created since the last download in the source folder.
Show detailed transfer logs	Select this check box to have more details on the download shown in the log, for example, full file paths.
Prompt when overwriting or deleting local items	Select this check box to have PhpStorm ask you for confirmation before overwriting or deleting local items for synchronization.

See Also

Concepts:

- [Project](#)

Procedures:

- [Creating and Managing Projects](#)

Reference:

- [Deployment](#)
- [Create New Project: Review Deployment Options](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Create New Project: Specify Local Server

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

[File | New Project From Existing Files - Web server is installed locally, source files are located under its document root](#)

The page opens when you click Next on the [Create New Project: Choose Project Directory](#) page. On this page of the wizard, choose the local [server access configuration](#) to use. You can select an existing configuration or define a new one.

Item	Description
Add new local server	Select this option to move to the Create New Project: Add Local Server page and specify a new Web server configuration.
Use existing server	Select this option to use one of the existing Web server configurations from the list below.
Name	This read-only field shows the name of the local server access configuration.
URL	This read-only field shows the URL address of the server configuration root .
Don't check HTTP connection to server	<ul style="list-style-type: none"> When this check box is cleared, PhpStorm checks whether the specified URL address ensures successful connection to the server. When this check box is selected, PhpStorm moves to the next page of the Wizard without any connection check.

See Also

Concepts:

- [Project](#)

Procedures:

- [Creating and Managing Projects](#)
- [Configuring PHP Development Environment](#)
- [Enabling PHP Support](#)
- [Creating a Project from Downloaded Files](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>

- <http://youtrack.jetbrains.net/issues/WI>

Create New Project: Add Local Server

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

File | New Project From Existing Files - Web server is installed locally, source files are located under its document root

The page opens if you have selected the Add new local server option on the [Create New Project: Specify Local Server](#) page. On this page, define a new local Web server access configuration.

Item	Description
Name	In this text box, type the name of the new configuration.
Web server root URL	In this text box, type the URL address of the server configuration root .
Open	Click this button to open the page at the specified URL address in the default browser and make sure that the URL actually points at the server configuration root .

See Also

Concepts:

- [Project](#)

Procedures:

- [Creating a Project from Downloaded Files](#)
- [Creating and Managing Projects](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Create New Project: Specify Remote Server

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

File | New Project From Existing Files - Web server is installed locally, source files are located elsewhere locally

File | New Project From Existing Files - My web server is on remote host, files are accessible via network share or mounted drive

File | New Project From Existing Files - My web server is on remote host, files are accessible via FTP/SFTP

/

The page opens when you click Next on the [Create New Project: Specify Local Path](#) page. On this page of the Wizard, choose the remote Web server to use. You can select a configured Web server or define a new configuration.

Item	Description
Add new remote server	Select this option to move to the Create New Project: Add Remote Server page and specify a new Web server configuration.
Use existing server	Select this option to use one of the existing Web server configurations from the list below.
Name	This read-only field shows the name of the remote server configuration.
URL address	This read-only field shows the URL address of the server document root according to the chosen configuration.
Mounted Folder	This read-only field shows the path to the mounted drive.
	<p>Note</p> <p>The field is available if you have selected the My web server is on remote host, files are accessible via network share or mounted folder option on the first page of the Wizard.</p>
Host	This read-only field shows the URL address of the FTP or SFTP host in the format <code>ftp://<host_name>:<port></code> or <code>ftps://<host_name>:<port></code>
	<p>Note</p> <p>The field is available if you have selected the My web server is on remote host, files are accessible via FTP/SFTP option on the first page of the Wizard.</p>
Don't check HTTP connection to server	<ul style="list-style-type: none"> • When this check box is cleared, PhpStorm checks whether the specified URL address ensures successful connection to the server. • When this check box is selected, PhpStorm moves to the next page of the Wizard without any connection check.

See Also

Concepts:

- [Project](#)

Procedures:

- [Creating and Managing Projects](#)
- [Configuring PHP Development Environment](#)
- [Enabling PHP Support](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Create New Project: Add Remote Server

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

File | New Project From Existing Files - Web server is installed locally, source files are located elsewhere locally

File | New Project From Existing Files - My web server is on remote host, files are accessible via network share or mounted drive

File | New Project From Existing Files - My web server is on remote host, files are accessible via FTP/SFTP

/

The page opens if you have selected the **Add new remote server** option on the [Create New Project: Specify Remote Server](#) page. On this page, specify a new configuration of settings to connect an authenticate to a remote server.

Item	Description
Name	In this text box, type the name of the new remote server configuration.
Access type	From this drop-down list, choose the way to access the server. The available options are: <ul style="list-style-type: none"> • FTP - choose this option to have PhpStorm access the server via the FTP file transfer protocol. • SFTP - choose this option to have PhpStorm access the server via the SFTP file transfer protocol. • Mounted folder - choose this option to have PhpStorm access the server via a mounted folder. • Local - choose this option to create a configuration with a Web server running on your local machine.
FTP/SFTP host	In this text box, specify the host name of the FTP/SFTP server to download the files from.
Port	In this text box, specify the port to use. The default values are: <ul style="list-style-type: none"> • 21 for FTP • 22 for SFTP
Advanced options	Click this button to specify the following additional uploading settings in the Advanced Options dialog box that opens: <ul style="list-style-type: none"> • Passive mode - select this check box to set the client to the passive mode. • Show hidden files - select this check box to have the hidden files and directories (those with names that start with a dot .) shown in the Remote Host Tool Window. • Limit concurrent connections - select this check box to have PhpStorm restrict the number of connections to be supported simultaneously and specify the maximum number of allowed connections in the text box.
User name	In this text box, type your user name for authentication to the server.
Log in as anonymous	Select this check box to enable anonymous access to the server with your email address as password.
Auth type	From this drop-down list, select the client authentication method. The available options are: <ul style="list-style-type: none"> • Password - select this option to use standard authentication through a password. • Key pair (OpenSSH) - select this option to use SSH authentication via a key pair.
	Warning PhpStorm supports private keys generated in accordance with the OpenSSH protocol .
	Note The field is available for SFTP server configuration only.
Password	In this text box, type your password for authentication to the server.
Private key file	In this text box, specify the location of your private key file.
	Note The field is available for SFTP server configuration only.
Passphrase	In this text box, specify your authentication passphrase.
	Note The field is available for SFTP server configuration only.
Save password	Select this check box to have PhpStorm remember the specified password.
Save passphrase	Select this check box to have PhpStorm remember the specified passphrase.

Note

The field is available for SFTP server configuration only.

- Test FTP/SFTP connection** Click this button to check that the specified settings ensure successful connection via FTP/SFTP.
- Root path** In this text box, specify the root folder for accessing files relative to the home folder of the FTP/SFTP server. Type the path manually, click the **Browse** button  and select the desired folder in the **Choose Root Path** dialog box that opens, or click the **Autodetect** button.
- Autodetect** Click this button to have PhpStorm detect the user home folder settings on the FTP/SFTP server and set up the **root path** according to them.
- Mounted folder** In this text box, type the path to the mounted folder which is associated with the files location on the server.
- Web server root URL** In this text box, specify the URL address of the [Web server root folder](#).
- Open** Click this button to make sure that the specified server root URL address is accessible and points at the correct Web page.
- Don't check HTTP connection to server**
 - When this check box is cleared, PhpStorm checks whether the specified URL address ensures successful connection to the server.
 - When this check box is selected, PhpStorm moves to the next page of the Wizard without any connection check.

See Also

Concepts:

- [Project](#)

Procedures:

- [Creating and Managing Projects](#)

Reference:

- [Deployment](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Create New Project: Choose Remote Path

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

- File | New Project From Existing Files - Web server is installed locally, source files are located elsewhere locally
- File | New Project From Existing Files - My web server is on remote host, files are accessible via network share or mounted drive
- File | New Project From Existing Files - My web server is on remote host, files are accessible via FTP/SFTP

The page opens when you click the Next button on the [Create New Project: Specify Remote Server](#) or [Create New Project: Add Remote Server](#) page. On this page, specify the folder on the selected remote FTP/SFTP server or mounted drive below which the sources to set up the project around are located.

See Also

Concepts:

- [Project](#)

Procedures:

- [Creating and Managing Projects](#)

Reference:

- [Deployment](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Create New Project: Specify Web Path (Remote)

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

- File | New Project From Existing Files - Web server is installed locally, source files are located elsewhere locally
- File | New Project From Existing Files - My web server is on remote host, files are accessible via network share or mounted drive
- File | New Project From Existing Files - My web server is on remote host, files are accessible via FTP/SFTP

The page opens when you click Next on the [Create New Project: Choose Remote Path](#) page. On this page, compose the Web path to access the project root directory via HTTP.

Item	Description
Web path for project root	In this text box, specify the path to the selected project directory relative to the Web server document root URL .
Project URL	This read-only field shows the full URL address via which the project root directory will be available. By default, the field is filled in with the name of the selected project root directory. PhpStorm dynamically constructs the full URL address as you type the relative path in the Web path for project root text

box.

See Also

Concepts:

- [Project](#)

Procedures:

- [Creating and Managing Projects](#)

Reference:

- [Deployment](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Create New Project: Specify Web Path (Local)

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

File | New Project From Existing Files - Web server is installed locally, source files are located under its document root

The page opens when you click Next on the [Create New Project: Specify Local Server](#) or on the [Create New Project: Add Local Server](#) page. On this page, compose the Web path to access the project root directory via HTTP.

Item	Description
Web path for project root	In this text box, specify the path to the selected project directory relative to the server configuration root URL .
Project URL	This read-only field shows the full URL address at which the project root directory will be available. By default, the field is filled in with the name of the selected project root directory. PhpStorm dynamically constructs the full URL address as you type the relative path in the Web path for project root text box.
Review PHP settings	Select this check box to move to the Create New Project: PHP Settings page and check the PHP installation.

See Also

Concepts:

- [Project](#)

Procedures:

- [Creating and Managing Projects](#)
- [Creating a Project from Downloaded Files](#)

Reference:

- [Deployment](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Create New Project: PHP Settings

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

File | New Project From Existing Files - Web server is installed locally, source files are located under its document root

The page opens if you have selected the **Review PHP settings** check box on the [Create New Project: Specify Web Path \(Local\)](#) page of the Wizard. Use this page to ensure that a PHP engine is installed on the specified local Web server and to configure a list of include paths. The page shows the PHP engine version installed on the selected local Web server. If no PHP engine is detected, PhpStorm displays a corresponding warning.

Item	Description
Include paths on server	Use the controls in this area to manage a list of include paths on the server: <ul style="list-style-type: none"> • Add - click this button to have a new entry added to the list. • Remove - click this button to remove the selected entry from the list. • Browse () - click this button to open the Select Path dialog box and choose the desired folder.

See Also

Concepts:

- [Project](#)

Procedures:

- [Creating and Managing Projects](#)
- [Configuring PHP Development Environment](#)

- [Enabling PHP Support](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Optimize Imports Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Code | **Optimize Imports**

Ctrl+Alt+OCommand Alt O

In this dialog box, specify where you want PhpStorm to remove unused import statements from, in order to optimize the import procedure.

Item	Description
File	Click this option to have unused imports removed from the current file. This option is selected by default, if a file in the editor has the focus, or if it is selected in the Project tool window .
All files in directory <directory name>	Click this option to have unused imports removed from all files in the current directory at once. This option is selected by default, if a directory is selected in the Project tool window .
Include subdirectories	Select this check box to have unused imports removed from all files in all subfolders of the current directory.
	Note The check box is shown, if the directory in question has subdirectories. The check box is enabled only when the All files in directory option is selected.
Do not show this dialog in the future	Select this check box to perform import optimization in the future silently.

See Also

Procedures:

- [Creating and Optimizing Imports](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Override Server Path Mappings Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Some features described here are available in PHP Developer edition only.

The dialog box opens when you click the **Server paths mappings** button in the **Debug** area of the [Run/Debug Configuration: PHP Web Application](#) or [Run/Debug Configuration: PHPUnit on Server](#) dialog box.

Use this dialog box to map folders on your local machine with folders on the server. These mappings are used during remote debugging and testing as the basis for switching between a test or a stack trace of an exception or assertion and the corresponding source code.

Tip

PhpStorm detects these mappings automatically but still provides you with this possibility to specify them manually.

Item	Description
Local Path on Client	In this text box, specify the absolute path to the desired local folder. Type the path manually or click the Browse button  and select the desired folder in the Select Path dialog box.
Local Path on Server	In this text box, specify the path to the corresponding folder on the server according to the file system used on the server.
Add	Click this button to have a new line added to the list of mappings.
Remove	Click this button to remove the selected mapping from the list.

See Also

Procedures:

- [PHP-Specific Guidelines](#)

Reference:

- [Run/Debug Configuration: PHPUnit on Server](#)
- [Run/Debug Configuration: PHP Web Application](#)
- [Deployment: Mappings Tab](#)
- [Test Runner Tab](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Print

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

File | Print

Use this dialog box to specify the text to print and configure the print layout.

In this section:

- [Settings Tab](#)
- [Advanced Tab](#)
- [Header and Footer Tab](#)

Item	Description
File <name>	Select this option to print the file, which is currently selected in the Project view or opened in the editor.
Selected text	Select this option to print the text selected in the editor.
All files in the directory	Select this option to print all files in the current directory.
Include subdirectories	Select this check box to have the files in the subdirectories of the current directory printed as well.

See Also

Reference:

- [Settings Tab](#)
- [Header and Footer Tab](#)
- [Advanced Tab](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Settings Tab

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

In this tab, specify the basic print layout settings.

Item	Description
Paper size	From this drop-down list, select the desired paper size.
Font	From the drop-down lists in this area, select the desired font style and size.
Show line numbers	Select this check box to have line numbers printed.
Draw border	Select this check box to have a border printed.
Orientation	In this area, specify the paper orientation. The available options are: <ul style="list-style-type: none"> • Landscape • Portrait
Style	In this area, specify the style of the printout by selecting the relevant check boxes. The available options are: <ul style="list-style-type: none"> • Color printing • Syntax printing • Print as graphics

See Also

Reference:

- [Print](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Advanced Tab

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Item	Description
Wrapping	In this area, configure text wrapping. The available options are:

- No wrap
- Wrap at word breaks

Margins Use the text boxes in this area to specify the margins in inches.

See Also

Reference:

- [Print](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Header and Footer Tab

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

In this tab, specify the contents and placement of the header and footer.

Item	Description
Text line	In this text box, specify the contents of the header or footer. If necessary, combine plain text with variables. By default, PhpStorm suggests to print the name of the file in the header and the current page number in the footer.
Placement	Use this drop-down list to specify whether the above line will be printed in the header or in the footer.
Alignment	From this drop-down list, select the desired alignment.
Font	In this area, specify the desired font style and size to print the header and footer text.

See Also

Reference:

- [Print](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Productivity Guide

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Help | Productivity Guide

The Productivity Guide shows statistics of PhpStorm features usage. This dialog regulates which tips display, when the Progress Bar is visible.

The table in the upper section of the dialog shows the list of features. Click on an entry opens its preview. Click on a column header enables sorting in the ascending or descending order.

Item	Description
Feature	List of analyzed PhpStorm features.
Group	Shows the group which the selected feature belongs to.
Used	Shows how many times the selected feature has been used since the first launch of PhpStorm.
Last used	Shows the lapse of time since the last invocation of the selected feature.
Average usage frequency	Shows how often the selected feature is used. The average value is estimated over the period since the first launch of PhpStorm.
Show Productivity Guide while compiling	If this option is checked, the personalized Tips of the Day window appears during compilation.
Show Productivity Guide during startup and other lengthy operations	If this option is checked, the personalized Tips of the Day window appears in course of version control operations, loading projects, and other activities that involve showing a progress bar.

See Also

Reference:

- [Usage Statistics](#)

Getting Started:

- [Getting Help](#)

Web Resources:

- <http://www.jetbrains.com/devnet/wi>
- <http://youtrack.jetbrains.com/issues/WI>

PSI Viewer

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: 

[Tools](#) | [View PSI Structure](#)

Use this viewer to explore internal structure of the various files or fragments of source code, as they are interpreted by PhpStorm. The dialog box is non-modal and enables you to keep on working with PhpStorm while being opened.

Item **Description**

Show PSI structure for	Use this drop-down list to specify file type, or language constructs to be explored. The set of recognized file types depends on the supported languages and installed plugins.
Show PsiWhiteSpace	If this check box is selected, the generated tree view will contain <code>PsiWhiteSpace</code> nodes, corresponding to the spaces in the source code. When you select of clear this check box, the tree view of the PSI structure changes accordingly.
Show Tree Nodes	
Dialect	This drop down list becomes available for the languages that support dialects, for example, SQL, JavaScript etc.
Text	Use this pane to enter source code to be explored. PhpStorm suggests the following ways to supply code: <ul style="list-style-type: none"> • Type immediately within the text area. • Paste from clipboard. If you have copied some text from the editor, and then open the PSI viewer, the previous contents of the Text pane is selected, which enables you to overwrite it from the clipboard using <code>Ctrl+V</code> Command <code>V</code> or <code>Shift+Insert</code> Shift <code>Insert</code>, or <code>Ctrl+Shift+V</code> Command <code>Shift V</code> or <code>Ctrl+Shift+Insert</code> Command <code>Shift Insert</code>. <p>Note that some editing features are also available: removing line at caret <code>Ctrl+Y</code>Command <code>Y</code>, duplicating text <code>Ctrl+D</code>Command <code>D</code>, and adding line with <code>Shift+Enter</code>Shift <code>Enter</code>.</p>
PSI Structure	This read-only pane displays the PSI structure tree view, generated on clicking the Build PSI Tree button, according to the file type selected in the Show PSI structure for drop-down list. <p>Navigating though the tree view highlights the corresponding fragments of source code in the Text pane. If currently selected tree node has references, they are also displayed in the References pane.</p>
References	This read-only field shows references to the nodes of the PSI Structure tree view (if any). <p>Unresolved references are shown red; the corresponding fragments of source code are also highlighted with a red frame.</p>
Build PSI Tree	Click this button to generate PSI structure tree view of the code in Text pane, according to the file type selected in the Show PSI structure for drop-down list. <p>If source code in the Text pane has been modified, use this button to refresh the tree view.</p>

See Also

- Concepts:
- [Plugins](#)
- Procedures:
- [Viewing PSI Structure](#)
- External Links:
- <http://blogs.jetbrains.com/idea/2009/11/psi-viewer/> 
- Web Resources:
- <http://www.jetbrains.net/devnet/community/wi> 
 - <http://youtrack.jetbrains.net/issues/WI> 

Recent Changes Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: 

[View](#) | [Recent Changes](#)
`Alt+Shift+CA`lt Shift `C`

In the **Recent Changes** pop-up window, use the arrow keys to navigate through the list, and the `Enter` key to view the selected changes. The [Differences Viewer](#) is displayed.

See Also

- Concepts:
- [Local History](#)
- Procedures:
- [Viewing Local History of a File or Folder](#)
- Reference:
- [Differences Viewer](#)
 - [Show History for Folder Dialog](#)
 - [Show History for File / Selection Dialog](#)
- Web Resources:
- <http://www.jetbrains.net/devnet/community/wi> 

- <http://youtrack.jetbrains.net/issues/WI>

Refactoring Dialogs

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

All refactoring dialogs provide similar controls to preview results and perform refactoring. The specific options and controls are described in the following topics:

- [Copy Dialogs](#)
- [Change Signature Dialog for JavaScript](#)
- [Extract Method Dialog](#)
- [Move File Dialog](#)
- [Rename Dialogs](#)
- [Safe Delete Dialog](#)
- [Extract Interface Dialog](#)
- [Extract Superclass Dialog](#)
- [Inline Dialogs](#)
- [Introduce Constant Dialog](#)
- [Introduce Variable Dialog](#)
- [Introduce Field Dialog](#)
- [Introduce Parameter Dialog for ActionScript](#)
- [Introduce Parameter Dialog for JavaScript](#)

See Also

Procedures:

- [Refactoring Source Code](#)

Reference:

- [Find Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Copy Dialogs

Previous | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Refactor | Copy

F5F5
Shift+F5Shift F5

Use this dialog box to create copies (F5F5) of files or directories in the specified target location or their clones (Shift+F5Shift F5) in the same directory.

Item	Description
To directory	In this drop-down list, specify the target directory where the copy of the selected file or directory will be created. If necessary, click the Browse button  and choose the desired location in the Select Target Directory dialog box that opens. This field is only available when the Copy command is invoked.
New name	In this text box, specify the name of the new file or directory.

See Also

Procedures:

- [Copy/Clone](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Change Signature Dialog for JavaScript

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Refactor | Change Signature

Ctrl+F6Command F6

Use this dialog to [change the function signature](#) and to perform other, related tasks.

Item	Description
Name	Use this field to modify the function name.
Use the table and the controls to the right of it to manage the function parameters and their properties .	
Name	Use this field to specify the name of a parameter.

Value	<p>A parameter value.</p> <p>If the parameter is a required one, the specified value (or expression) will be passed to the function in the function calls. If the parameter is an optional one, this value will be used in the function body to initialize the parameter.</p>
Optional	<p>Select this check box if the parameter is optional. Your selection will define how the parameter value is used.</p>
 or Alt+InsertCommand N	<p>Use this icon or shortcut to start adding a new parameter.</p> <p>Specify the parameter name and value. Also, specify whether the parameter is optional.</p> <p>Note that you can propagate the parameters you have added to the calling methods.</p>
 or Alt+DeleteMeta Delete	<p>Use this icon or shortcut to delete the selected parameter.</p>
 or Alt+UpCommand Up	<p>Use this icon or shortcut to move the selected parameter one line up in the list of parameters.</p>
 or Alt+DownCommand Down	<p>Use this icon or shortcut to move the selected parameter one line down in the list of parameters.</p>
 or Alt+GAlt G	<p>Use this icon or shortcut to propagate the added parameters to the calling methods.</p> <p>You can propagate new function parameters to any function that directly or indirectly calls the function whose signature you are changing.</p> <p>(There may be the functions that call the current function. These functions, in their turn, may be called by other functions. You can propagate new parameters to any of the functions in such sequences.)</p> <p>In the left-hand pane of the Select Methods to Propagate New Parameters dialog, expand the necessary nodes and select the check boxes next to the functions you want the new parameters to be propagated to.</p>
Signature Preview	<p>In this area, the current function signature is shown. (The information in this area is synchronized with the changes you are making to the function signature.)</p>
Refactor	<p>Click this button to perform the refactoring right away.</p>
Preview	<p>Click this button to see the expected changes prior to actually performing the refactoring.</p>

See Also

Code Examples:

- [Examples for JavaScript](#)

Procedures:

- [Changing a function signature in JavaScript](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Extract Method Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac 

Refactor | Extract Method
Ctrl+Alt+MCommand Alt M

Note

The title of the dialog box may change to **Extract Function**:

- In the JavaScript context.
- In the PHP context, when the code selection is made inside a function or a script.

Item	Description
Name	In this text box, specify the name of the function or method to be generated on the basis of the selected source code.
Visibility	<p>In this area, specify the visibility scope of the method to be generated. The available options are:</p> <ul style="list-style-type: none"> • Public • Protected • Private

Note

This area is available only in the **Extract Method** dialog box when refactoring is invoked from a method of a PHP class.

Declare static Select this check box to have a static method created.

Note

1. If the new method cannot be declared static, or, vice-versa, can be created only as a static method, the **Declare Static** check box is disabled.
2. This check box is available only in the **Extract Method** dialog box when refactoring is invoked from a method of a PHP class.

Declare functional expression	Select this check box to have the new function defined through a function expression, for example, <code>new_method = function()</code> .
	<p>Note</p> <p>The check box is only available when refactoring is invoked in the JavaScript context.</p>
Output variable(s)	This read-only text box displays the name of the variable through which the output of the new method/function will be passed to the calling method/function. Depending on your choice in the Return output variable(s) through area, this variable either will be used in a return statement or will be declared as the passed by reference parameter of the new method/function.
Return output variable(s) through	<p>In this area, specify the way in which the new method or function will return the output variables to the callee..</p> <ul style="list-style-type: none"> • Return statement - select this option to have the output variables returned by value. If the Output variable(s) read-only field shows exactly one output variable, it will be used as the return value. If the selection outputs several variables, these variables will be returned as an array. • Parameter(s) passed by reference - select this option to have the output variables returned by reference. PhpStorm will generate a method/function without a return statement. Instead, the output variables will be added to the set of input parameters in the method/function declaration. The names of these variables will be prepended with an ampersand <code>&</code>. <p>Note</p> <p>The area is available only when refactoring is invoked in the PHP context.</p>
Parameters	<p>In this area, specify the parameters to be passed to the new method/function.</p> <p>Note</p> <ol style="list-style-type: none"> 1. If a parameter that is critical for the functionality of the new method is not selected, PhpStorm will be unable to proceed with the refactoring. 2. In the PHP context, the list may also contain output variables that will be passed by reference from the calling function. The names of these variables are prepended with an ampersand <code>&</code>.
Move Up/Down	Use these buttons to change the order of the parameters.
Replace tail "break/continue" statements with return statements	Select this check box to have PhpStorm transform tail break or continue statements if the selection contains any.
Signature preview	In this read-only field, view the declaration of the new method/function.

See Also

Procedures:

- [Extract Method](#)

External Links:

- [Replace conditional logic with strategy pattern](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Move File Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Refactor | Move
⌘F6

The **Refactor | Move** menu command invokes one of the following dialog boxes depending on the selected item to be moved:

- The **Move File** dialog box is invoked when a file is selected in the **Project** tool window or opened in the editor.
- The **Move Directory** dialog box is invoked when a directory is selected in the **Project** tool window.

Item	Description
To directory	In this text box, specify the destination directory. Type the path manually or click the Browse button  and choose the target directory in the Select Path dialog box that opens.
Search for references	If this check box is not selected, only the selected file or directory will be moved to the specified location.

See Also

Procedures:

- [Move Refactorings](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

3.0+

Rename Dialogs

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: 

Refactor | Rename
Shift+F6 Shift+F6

Item	Description
Rename <symbol name> and its usages to	In this field, specify a new name for the symbol.
Search for references	Note This check box is only available for <code>Rename File</code> and <code>Rename Directory</code> refactoring. <ul style="list-style-type: none"> Clear the check box to have PhpStorm rename only the file or folder itself without attempting to find its usages and update them accordingly. Select this check box to have PhpStorm search for usages of the file/folder name and apply the refactoring to them.
Search in comments and strings	Select this check box to have the changes applied to comments and strings.
Search for text occurrences	Select this check box to have the changes applied to the documentation, HTML, and other files included in the project.

Note
PhpStorm memorizes the latest settings in this dialog box.

See Also

- Procedures:
- [Rename Refactorings](#)

- Web Resources:
- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>

Safe Delete Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: 

Refactor | Safe Delete
Alt+DeleteMeta Delete

Use this dialog box to configure the scope to apply the [Self Delete](#) refactoring to.

Item	Description
Search in comments and strings	Select this check box to apply the changes to comments and strings.
Search for text occurrences	Select this check box to apply the changes text files within the project (such as documentation, HTML, JSP, etc.)

See Also

- Procedures:
- [Safe Delete](#)

- Web Resources:
- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>

Extract Interface Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Refactor | Extract Interface

Item	Description
Extract interface from	This read-only field shows the name of the source package that contains the class to extract an interface from.
Extract interface	When this option is selected, PhpStorm extracts a new interface but does not use it immediately and the source code is not changed.
Interface name	In this text box, type the name for the new interface. The text box is available if the <code>Extract interface</code> option is selected.
Extract interface and use it where possible	Select this option to have an interface extracted and immediately applied to the source code, with the suggested changes displayed in the dedicated tab of the Find tool window.

Rename original class and use interface where possible	Use this option to rename the original class and make it an implementation of the newly created interface.
Rename implementation class to	In this text box, type the new name for the original class. The text box is available if the Rename original class and use interface where possible option is selected.
Package for new interface	In this drop-down list, specify the package for the new interface. If necessary, click the Browse button  and choose the target package in the Select Path dialog box that opens.
Members to Form Interface	In this area, specify the methods of the class, as well as final static fields (constants) to be included in the new interface. To have an element included in the interface, select the check box next to it.
JavaDoc/ASDoc	In the this area, specify the action to be applied to the inline documentation. The available options are: <ul style="list-style-type: none"> • As is - select this option to have the inline documentation left where it is. • Copy - select this option to have the inline documentation copied to the extracted interface without removing it from its current location. • Move - select this option to have the inline documentation moved to the extracted interface and delete it from its current location.

See Also

Procedures:

- [Extract Interface](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Extract Superclass Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Refactor | Extract Superclass

Item	Description
Extract superclass from	This read-only field shows the name of the source package that contains the class to extract a superclass from.
Extract superclass	When this option is selected, PhpStorm extracts a new superclass but does not use it immediately and the source code is not changed.
Superclass name	In this text box, type the name for the new superclass. The text box is available if the Extract superclass option is selected.
Extract superclass and use it where possible	Select this option to have a superclass extracted and immediately applied to the source code, with the suggested changes displayed in the dedicated tab of the Find tool window.
Rename original class and use superclass where possible	Use this option to rename the original class and make it an implementation of the newly created superclass.
Rename original class to	In this text box, type the new name for the original class. The text box is available if the Rename original class and use superclass where possible option is selected.
Package for new superclass	In this drop-down list, specify the package for the new superclass. If necessary, click the Browse button  and choose the target package in the Select Path dialog box that opens.
Members to Form Superclass	In this area, specify the members of the class to be moved or delegated to the new superclass. To have an element included in the interface, select the check box next to it.
Make Abstract	Select this option to leave the method implementation within the current class, and declare it abstract in the extracted superclass.
JavaDoc/ASDoc for abstracts	In the JavaDoc/ASDoc area, specify the action to be applied to the inline documentation. The available options are: <ul style="list-style-type: none"> • As is - select this option to have the inline documentation left where it is. • Copy - select this option to have the inline documentation copied to the extracted superclass without removing it from its current location. • Move - select this option to have the inline documentation moved to the extracted superclass and delete it from its current location.

See Also

Procedures:

- [Extract Superclass](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Inline Dialogs

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts:  

Refactor | Inline

<context menu of a selection> | Refactor | Inline

Ctrl+Alt+NCommand Alt N

Inline variable dialog

The **Inline Variable** refactoring allows to replace a redundant variable with its value. See [examples](#).

To access the **Inline Variable** dialog box through a menu item or the keyboard shortcut, position the cursor at the variable to be inlined. The dialog box does not contain any controls but just display the following message:

```
Inline variable <variable name>? (<the number of variable occurrences>)
```

See Also

Procedures:

- [Inline](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Introduce Constant Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Refactor | Introduce Constant

Ctrl+Alt+CCommand Alt C

Item	Description
Name	In this text box, specify the name of the new constant.
Replace all occurrences	Select this check box to have PhpStorm automatically replace all the occurrences of the selected expression (if the selected expression is found more than once in the method).

See Also

Procedures:

- [Introduce Variable](#)
- [Introduce Constant](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Introduce Variable Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Refactor | Introduce Variable

Ctrl+Alt+VCommand Alt V

Refactor | Introduce Variable

Ctrl+Alt+VCommand Alt V

Use this dialog box to introduce:

- Variables in PHP or JavaScript contexts.
- Local variables for JavaScript 1.7 and higher.
- JavaScript constants.

Item	Description	Language
Name	Specify the name for the new variable.	All

See Also

Procedures:

- [Introduce Variable](#)
- [Introduce Variable in JavaScript](#)

External Links:

- [ActionScript refactoring improvements in IntelliJ IDEA 10.5](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Introduce Field Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Refactor | Introduce Field
Ctrl+Alt+FCommand Alt F

Item	Description
Name	In this text box, specify the name of the new field.
Initialize in	In this area select where the new field will be initialized in.
Visibility	In this area, specify the visibility scope for the new field. The available options are: <ul style="list-style-type: none"> • Public - if you select this option, the new field will be accessible from anywhere. • Private - if you select this option, the new field will be accessible only from the current class. • Protected - if you select this option, the new field will be accessible from the current class as well as from its inherited and parent classes.
Replace all occurrences	Select this check box to have PhpStorm automatically replace all the occurrences of the selected expression.
	Note The check box is enabled only if the selected expression is used more than once in the class. All the found occurrences of the expression are highlighted.

See Also

Procedures:

- [Introduce Field](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Introduce Parameter Dialog for ActionScript

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Refactor | Introduce Parameter
Ctrl+Alt+PCommand Alt P

Use this dialog to specify the options and settings related to the [Introduce Parameter refactoring in ActionScript](#).

Item	Description
Type	Specify the type of the new parameter. Usually, you don't need to change the type suggested by PhpStorm.
Name	Specify the name for the new parameter.
Value	The expression to be replaced with the new parameter. Initially, this field contains the expression that you have selected. Normally, this initial value does not need to be changed.
Optional parameter	If you want the new parameter to be an optional parameter, select this check box. For information about required and optional parameters, see the discussion of function parameters in Flex/ActionScript documentation .
Replace all occurrences	Select this option to replace all the occurrences of the selected expression within the function.

See Also

Code Examples:

- [Example for ActionScript](#)

Procedures:

- [Introducing a parameter in ActionScript](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Introduce Parameter Dialog for JavaScript

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Refactor | Introduce Parameter
Ctrl+Alt+PCommand Alt P

Use this dialog to specify the options and settings related to the [Introduce Parameter refactoring in JavaScript](#).

Item	Description
Type	Specify the type of the new parameter. Usually, you don't need to change the type suggested by PhpStorm.
Name	Specify the name for the new parameter.

Value	The expression to be replaced with the new parameter. Initially, this field contains the expression that you have selected. Normally, this initial value does not need to be changed.
Optional parameter	<p>If this option is selected, the new parameter is assigned a value in the function body. The value corresponds to that currently set in the Value field. The calls to the function do not change.</p> <p>If this option is not selected, all the function calls change according to the new function signature. The value specified in the Value field is added to the function calls and thus is passed to the function. No explicit value assignment for the new parameter is added to the function body.</p>
Replace all occurrences	Select this option to replace all the occurrences of the selected expression within the function.

See Also

Code Examples:

- [Example for JavaScript](#)

Procedures:

- [Introducing a parameter in JavaScript](#)
-

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Reformat Code Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Code | Reformat Code
 Ctrl+Alt+LCommand Alt L

In this dialog box, specify the scope to apply reformatting to.

Item	Description
File	Select this option to have all the source code in the current file reformatted.
Selected text	Select this option to have the currently selected fragment of source code reformatted.
All files in directory...	Select this option to have the source code in all the files in the current directory reformatted at once.
Include subdirectories	Select this check box to have source code from files in subfolders reformatted.
Optimize imports	Select this check box to remove unused import statements from the code within the selected scope.
Run	Click this button to have PhpStorm start reformatting the source code within the specified scope.

See Also

Procedures:

- [Reformatting Source Code](#)

Reference:

- [Code Style](#)
- [Code Style](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Register New File Type Association Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

The dialog box opens when you attempt to open a file in the editor, but PhpStorm does not recognize its type by the file extension. Use the dialog box to associate unknown file extensions with existing file types.

Item	Description
Open matching files in PhpStorm	When this option is selected, PhpStorm treats the type of the file to be opened as one of the recognized file types. Choose the relevant type from the list box below, that displays all the file types recognized by PhpStorm.
File Pattern	In this text box, specify the file pattern to be associated with the selected file type. By default, the text box shows the following pattern: <code>*.<current file full extension></code> .
Open matching files in associated application	When this option is selected, PhpStorm attempts to open the selected file using its native application, if this application is available.

Tip

You can [change the association](#) later in the [File Types](#) dialog box.

See Also

Procedures:

- [Creating and Registering File Types](#)

Reference:

- [New File Type](#)
- [File Types](#)
- [Editor, Colors and Fonts](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Run/Debug Configurations

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Toolbar

Item	Keyboard shortcut	Description
	InsertInsert	Click this button to add new configuration to the list.
	DeleteDelete	Click this button to remove the selected configuration from the list.
	Ctrl+DCommand D	Click this button to create a copy of the selected configuration.
	Edit Defaults	Click this button to edit the default configuration templates. The defaults are used for the newly created configurations.
	UpUp DownDown	Use these buttons to move the selected configuration up and down in the list. The order of configurations in the list defines the order, in which configurations appear in the Run/Debug drop-down list on the main toolbar.

Common options

Item	Description
Name	In this text box, specify the name of the current run/debug configuration.
Defaults	This node in the left-hand pane of the dialog box contains the default run/debug configuration settings. Select the desired configuration to change its default settings in the right-hand pane. The defaults are applied to all newly created run/debug configurations.
Temporary configurations limit	Specify here the maximum number of temporary configurations to be stored and shown in the Select Run/Debug Configuration drop-down list.
Before Launch	In this area, specify the activities to be performed before starting the current configuration. The available options are: <ul style="list-style-type: none"> • Show Settings - select this check box to have the Run/Debug Configurations dialog box shown every time you launch this run/debug configuration. If this check box is cleared, the run/debug configuration starts silently.
Share configuration	If this check box is selected, the run/debug configurations become available to the other team members. The shared run/debug configurations are kept in separate xml files under <code>.idea\runConfigurations</code> folder, while the local run/debug configurations are kept in the <code>.idea\workspace.xml</code> .

See Also

Procedures:

- [Creating and Editing Run/Debug Configurations](#)

Reference:

- [Debugger](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Run/Debug Configuration: Query Language Console

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Use this dialog box to configure execution of queries to databases from the **Database** console. You can also set up additional arguments here, if needed.

When you run the **Database** console from the [Data Sources](#) tool window, a temporary Run/Debug configuration is created. You can save it and use later without reopening the corresponding tool window.

This section provides descriptions of the [configuration-specific items](#), and the [toolbar](#) and [options](#) that are common for all run/debug configurations.

Item	Description
Name	In this text box, specify the name of the current run/debug configuration.
VM Options	Specify the command-line options to be passed to the VM (for example, <code>user.language</code>).

When specifying the options, follow these rules:

- Use spaces to separate individual options, for example, `-client -ea -Xmx1024m`.
- If an option includes spaces, enclose the spaces or the argument that contains the spaces in double quotes, for example, `some "arg" or "some arg"`.
- If an option includes double quotes (e.g. as part of the argument), escape the double quotes by means of the backslashes, for example, `-Dmy.prop=\"quoted_value\"`.

Use Default Context

In this field, specify whether you want PhpStorm to execute queries in any context by default. To specify the context provider properties, click the **Browse** button  and select the desired context provider in the **Choose Data Source** dialog box that opens.

Toolbar

Item	Keyboard shortcut	Description
	InsertInsert	Click this button to add new configuration to the list.
	DeleteDelete	Click this button to remove the selected configuration from the list.
	Ctrl+DCommand D	Click this button to create a copy of the selected configuration.
	Edit Defaults	Click this button to edit the default configuration templates. The defaults are used for the newly created configurations.
	UpUp DownDown	Use these buttons to move the selected configuration up and down in the list. The order of configurations in the list defines the order, in which configurations appear in the Run/Debug drop-down list on the main toolbar.

Common options

Item	Description
Defaults	This node in the left-hand pane of the dialog box contains the default run/debug configuration settings. Select the desired configuration to change its default settings in the right-hand pane. The defaults are applied to all newly created run/debug configurations.
Temporary configurations limit	Specify here the maximum number of temporary configurations to be stored and shown in the Select Run/Debug Configuration drop-down list.
Before Launch	In this area, specify the activities to be performed before starting the current configuration. The available options are: <ul style="list-style-type: none"> • Run Phing target - if this check box is selected, the specified Phing target will be executed prior to running or debugging. To appoint a Phing target, click the Browse button  and select the desired target in the Choose Phing Target to Execute dialog box, that opens. • Show Settings - select this check box to have the Run/Debug Configurations dialog box shown every time you launch this run/debug configuration. If this check box is cleared, the run/debug configuration starts silently.
Share configuration	If this check box is selected, the run/debug configurations become available to the other team members. The shared run/debug configurations are kept in separate xml files under <code>.idea\runConfigurations</code> folder, while the local run/debug configurations are kept in the <code>.idea\workspace.xml</code> .

See Also

Concepts:

- [Data Sources](#)

Procedures:

- [Data Sources](#)

Reference:

- [Data Sources Tool Window](#)
- [Database Console Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Run/Debug Configuration: PHP Script

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Use this dialog box to configure running and debugging of single PHP files locally using a PHP console.

Click [here](#) for the description of the options that are common for all run/debug configurations.

Item	Description
Name	In this text box, specify the name of the current run/debug configuration.
File	In this text box, specify the location of the file to run or debug. Type the path to the file manually or click the Browse button  and select the desired location in the Choose PHP File dialog box that opens.
Custom working directory	In this text box, specify the PHP script execution directory of your choice. Type the path manually or click the Browse button  and select the desired folder in the Select Path dialog box that opens.
Note	

By default, the working directory for PHP script execution is the one that contains the script.

Script parameters In this text box, type the list of arguments to be passed to the PHP script, same way as if you were entering these parameters in the command line.

Debug Use the controls in this area to configure behaviour of the debugging tool.

- **Break at the first line** - select this check box to have the debugging tool stop at the first line of the source code.

Toolbar

Item	Keyboard shortcut	Description
	InsertInsert	Click this button to add new configuration to the list.
	DeleteDelete	Click this button to remove the selected configuration from the list.
	Ctrl+DCommand D	Click this button to create a copy of the selected configuration.
	Edit Defaults	Click this button to edit the default configuration templates. The defaults are used for the newly created configurations.
	UpUp DownDown	Use these buttons to move the selected configuration up and down in the list. The order of configurations in the list defines the order, in which configurations appear in the Run/Debug drop-down list on the main toolbar.

Common options

Item	Description
Defaults	This node in the left-hand pane of the dialog box contains the default run/debug configuration settings. Select the desired configuration to change its default settings in the right-hand pane. The defaults are applied to all newly created run/debug configurations.
Temporary configurations limit	Specify here the maximum number of temporary configurations to be stored and shown in the Select Run/Debug Configuration drop-down list.
Before Launch	In this area, specify the activities to be performed before starting the current configuration. The available options are: <ul style="list-style-type: none"> • Run Phing target - if this check box is selected, the specified Phing target will be executed prior to running or debugging. To appoint a Phing target, click the Browse button  and select the desired target in the Choose Phing Target to Execute dialog box, that opens. • Show Settings - select this check box to have the Run/Debug Configurations dialog box shown every time you launch this run/debug configuration. If this check box is cleared, the run/debug configuration starts silently.
Share configuration	If this check box is selected, the run/debug configurations become available to the other team members. The shared run/debug configurations are kept in separate xml files under <code>.idea\runConfigurations</code> folder, while the local run/debug configurations are kept in the <code>.idea\workspace.xml</code> .

See Also

Concepts:

- [Running and Debugging](#)

Procedures:

- [Debugging](#)
- [PHP-Specific Guidelines](#)

Reference:

- [Deployment](#)
- [Debugger](#)
- [PHP](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wiki/>
- <http://youtrack.jetbrains.com/issues/WI>

Run/Debug Configuration: PHP HTTP Request

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Use this dialog box to configure running and debugging of separate [HTTP Requests](#).

Click [here](#) for the description of the options that are common for all run/debug configurations.

Item	Description
Name	In this text box, specify the name of the current run/debug configuration.
Server	From this drop-down list, select the server access configuration to interact with the Web server where the application is executed.
URL	In this text box, specify the <code>host</code> element of the request in question. Type element relative to the host specified in the debug server configuration . As you type, PhpStorm composes the URL address on-the-fly and displays it below the text box.
Request method	From this drop-down list, choose the relevant request type. The available options are: <ul style="list-style-type: none"> • POST

- GET

Query String In this text box, type the query string of the request, this string will be appended to the request after the ? symbol.

Request Body In this text box, type the data to be sent to the server through the **POST** request.

Note
The text box is available only for request methods of the type **POST**.

Toolbar

Item	Keyboard shortcut	Description
	InsertInsert	Click this button to add new configuration to the list.
	DeleteDelete	Click this button to remove the selected configuration from the list.
	Ctrl+DCommand D	Click this button to create a copy of the selected configuration.
	Edit Defaults	Click this button to edit the default configuration templates. The defaults are used for the newly created configurations.
	UpUp DownDown	Use these buttons to move the selected configuration up and down in the list. The order of configurations in the list defines the order, in which configurations appear in the Run/Debug drop-down list on the main toolbar.

Common options

Item	Description
Defaults	This node in the left-hand pane of the dialog box contains the default run/debug configuration settings. Select the desired configuration to change its default settings in the right-hand pane. The defaults are applied to all newly created run/debug configurations.
Temporary configurations limit	Specify here the maximum number of temporary configurations to be stored and shown in the Select Run/Debug Configuration drop-down list.
Before Launch	In this area, specify the activities to be performed before starting the current configuration. The available options are: <ul style="list-style-type: none"> • Run Phing target - if this check box is selected, the specified Phing target will be executed prior to running or debugging. To appoint a Phing target, click the Browse button  and select the desired target in the Choose Phing Target to Execute dialog box, that opens. • Show Settings - select this check box to have the Run/Debug Configurations dialog box shown every time you launch this run/debug configuration. If this check box is cleared, the run/debug configuration starts silently.
Share configuration	If this check box is selected, the run/debug configurations become available to the other team members. The shared run/debug configurations are kept in separate xml files under <code>.idea\runConfigurations</code> folder, while the local run/debug configurations are kept in the <code>.idea\workspace.xml</code> .

See Also

Concepts:

- [Run/Debug Configuration](#)

Procedures:

- [Debugging a PHP HTTP Request](#)
- [Debugging PHP Applications](#)
- [Debugging](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Run/Debug Configuration: PHP Remote Debug

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Use this dialog box to configure debugging of PHP applications on a remote server.
Click [here](#) for the description of the options that are common for all run/debug configurations.

Item	Description
Name	In this text box, specify the name of the current run/debug configuration.
Server	Use this drop-down list to specify the Web server configuration to use. The list shows all the configurations that are currently available in PhpStorm.
	Click this button to open the Deployment page and view the details of the selected configuration there.
Ide key (session id)	In this text box, specify the key to identify the debugging session.
Debug	Use the controls in this area to configure behaviour of the debugging tool. <ul style="list-style-type: none"> • Break at the first line - select this check box to have the debugging tool stop at the first line of the source code.

Toolbar

Item	Keyboard shortcut	Description
	InsertInsert	Click this button to add new configuration to the list.
	DeleteDelete	Click this button to remove the selected configuration from the list.
	Ctrl+DCommand D	Click this button to create a copy of the selected configuration.
	Edit Defaults	Click this button to edit the default configuration templates. The defaults are used for the newly created configurations.
	UpUp DownDown	Use these buttons to move the selected configuration up and down in the list. The order of configurations in the list defines the order, in which configurations appear in the Run/Debug drop-down list on the main toolbar.

Common options

Item	Description
Defaults	This node in the left-hand pane of the dialog box contains the default run/debug configuration settings. Select the desired configuration to change its default settings in the right-hand pane. The defaults are applied to all newly created run/debug configurations.
Temporary configurations limit	Specify here the maximum number of temporary configurations to be stored and shown in the Select Run/Debug Configuration drop-down list.
Before Launch	In this area, specify the activities to be performed before starting the current configuration. The available options are: <ul style="list-style-type: none"> • Run Phing target - if this check box is selected, the specified Phing target will be executed prior to running or debugging. To appoint a Phing target, click the Browse button  and select the desired target in the Choose Phing Target to Execute dialog box, that opens. • Upload files to Remote Host - select this check box to have files uploaded to a remote host of your choice through FTP or SFTP protocol. To define the folders to be uploaded, click the Browse button  and select the desired folder in the Choose Local Paths to Upload dialog box that opens. • Show Settings - select this check box to have the Run/Debug Configurations dialog box shown every time you launch this run/debug configuration. If this check box is cleared, the run/debug configuration starts silently.
Share configuration	If this check box is selected, the run/debug configurations become available to the other team members. The shared run/debug configurations are kept in separate xml files under <code>.idea\runConfigurations</code> folder, while the local run/debug configurations are kept in the <code>.idea\workspace.xml</code> .

See Also

- Concepts:
- [Running and Debugging](#)
- Procedures:
- [Debugging](#)
 - [PHP-Specific Guidelines](#)
- Reference:
- [Deployment](#)
 - [Debugger](#)
 - [PHP](#)
- Web Resources:
- <http://www.jetbrains.net/devnet/community/wi> 
 - <http://youtrack.jetbrains.net/issues/WI> 

1.0+

Run/Debug Configuration: PHP Web Application

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Use this dialog box to configure running and debugging of PHP applications on a remote server.
Click [here](#) for the description of the options that are common for all run/debug configurations.

Item	Description
Name	In this text box, specify the name of the current run/debug configuration.
Server	Use this drop-down list to specify the Web server configuration to use. The list shows all the configurations that are currently available in PhpStorm.
	Click this button to open the Servers page and view the details of the selected configuration there.
Start URL	In this area, compose the URL address to access the application through. In the Start URL text box, specify the local file that implements the starting page of the application. Type the path to the desired file relative to the folder that is mapped to the root folder of the target host. PhpStorm concatenates the host root URL with the specified relative path and shows the URL address of the application starting page in the read-only field below.
Browser	From this drop-down list, select the Web browser to open the application in.
Debug	Use the controls in this area to configure behaviour of the debugging tool. <ul style="list-style-type: none"> • Break at the first line - select this check box to have the debugging tool stop at the first line of the source code.

Toolbar

Item	Keyboard shortcut	Description
	InsertInsert	Click this button to add new configuration to the list.
	DeleteDelete	Click this button to remove the selected configuration from the list.
	Ctrl+DCommand D	Click this button to create a copy of the selected configuration.
	Edit Defaults	Click this button to edit the default configuration templates. The defaults are used for the newly created configurations.
	UpUp DownDown	Use these buttons to move the selected configuration up and down in the list. The order of configurations in the list defines the order, in which configurations appear in the Run/Debug drop-down list on the main toolbar.

Common options

Item	Description
Defaults	This node in the left-hand pane of the dialog box contains the default run/debug configuration settings. Select the desired configuration to change its default settings in the right-hand pane. The defaults are applied to all newly created run/debug configurations.
Temporary configurations limit	Specify here the maximum number of temporary configurations to be stored and shown in the Select Run/Debug Configuration drop-down list.
Before Launch	In this area, specify the activities to be performed before starting the current configuration. The available options are: <ul style="list-style-type: none"> • Run Phing target - if this check box is selected, the specified Phing target will be executed prior to running or debugging. To appoint a Phing target, click the Browse button  and select the desired target in the Choose Phing Target to Execute dialog box, that opens. • Upload files to Remote Host - select this check box to have files uploaded to a remote host of your choice through FTP or SFTP protocol. To define the folders to be uploaded, click the Browse button  and select the desired folder in the Choose Local Paths to Upload dialog box that opens. • Show Settings - select this check box to have the Run/Debug Configurations dialog box shown every time you launch this run/debug configuration. If this check box is cleared, the run/debug configuration starts silently.
Share configuration	If this check box is selected, the run/debug configurations become available to the other team members. The shared run/debug configurations are kept in separate xml files under <code>.idea\runConfigurations</code> folder, while the local run/debug configurations are kept in the <code>.idea\workspace.xml</code> .

See Also

Concepts:

- [Running and Debugging](#)

Procedures:

- [Creating a PHP Web Application Debug Configuration](#)
- [Debugging](#)
- [PHP-Specific Guidelines](#)

Reference:

- [Deployment](#)
- [Debugger](#)
- [PHP](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

1.0+

Run/Debug Configuration: PHPUnit

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Use this dialog box to create a configuration to be used for running unit tests on PHP applications.

PHP unit testing requires the following prerequisites to be fulfilled:

- The [PHPUnit](#)  tool should be [installed and configured](#)  on your machine.
- The `PEAR` folder should be added to the list of the [project content roots](#).

The dialog box contains two tabs:

- [Configuration](#)
- [Test Groups](#)

Click [here](#) for the description of the options that are common for all run/debug configurations.

Item	Description
Name	In this text box, specify the name of the current run/debug configuration.

Configuration tab

In this tab, specify the unit tests to launch.

Item	Description
All in directory	<p>Select this option to have all the unit tests in a directory launched.</p> <p>In the Directory text box, specify the directory to search the unit test in. Type the path to the directory manually or click the Browse button  and select the desired directory in the Choose Test Directory dialog box, that opens.</p>
All in file	<p>Select this option to have all the unit tests in a file launched.</p> <p>In the File text box, specify the file to search the unit test in. Type the path to the file manually or click the Browse button  and select the desired directory in the Choose Test File dialog box, that opens.</p>
Class or suite	<p>Select this option to have all the unit tests in a test class or test suite launched.</p> <ul style="list-style-type: none"> In the File text box, specify the file to search the class or suite in. Type the path to the file manually or click the Browse button  and select the desired directory in the Choose Test File dialog box, that opens. In the Class text box, specify the desired class. Type the class name manually or click the Browse button  and select the desired class in the tree view, that opens.
Method	<p>Select this option to have a specific test method launched.</p> <ul style="list-style-type: none"> In the File text box, specify the file to search for the test method in. Type the file name manually or click the Browse button  and select the desired file in the tree view, that opens. In the Method text box, specify the desired method.
XML file	<p>Select this check box to run testing according to the configuration defined in a dedicated XML file. In the Use XML configuration file text box, specify the location of the file to use. Type the path manually or click the Browse button  and appoint the desired file in the Choose Test File dialog box, that opens.</p>
Custom working directory	<p>In this text box, specify the test execution directory of your choice. Type the path manually or click the Browse button  and select the desired folder in the Select Path dialog box that opens.</p> <p>Note</p> <p>By default, the working directory for test execution is the one that contains the test.</p>

Test groups tab

In this tab, set additional limits for tests to run.

Item	Description
Include/exclude groups	<p>When this check box is selected, PhpStorm displays a list of all test groups  that are defined in the specified scope.</p> <ul style="list-style-type: none"> To have the tests of a group executed, select the Include check box next to the desired group. To skip the tests of a group, select the Exclude check box next to the desired group.
Custom working directory	<p>In this text box, specify the PHP script execution directory of your choice. Type the path manually or click the Browse button  and select the desired folder in the Select Path dialog box that opens.</p> <p>Note</p> <p>By default, the working directory for PHP script execution is the one that contains the script.</p>

Toolbar

Item	Keyboard shortcut	Description
	InsertInsert	Click this button to add new configuration to the list.
	DeleteDelete	Click this button to remove the selected configuration from the list.
	Ctrl+DCommand D	Click this button to create a copy of the selected configuration.
	Edit Defaults	Click this button to edit the default configuration templates. The defaults are used for the newly created configurations.
	UpUp DownDown	Use these buttons to move the selected configuration up and down in the list. The order of configurations in the list defines the order, in which configurations appear in the Run/Debug drop-down list on the main toolbar.

Common options

Item	Description
Defaults	This node in the left-hand pane of the dialog box contains the default run/debug configuration settings. Select the desired configuration to change its default settings in the right-hand pane. The defaults are applied to all newly created run/debug configurations.
Temporary configurations limit	Specify here the maximum number of temporary configurations to be stored and shown in the Select Run/Debug Configuration drop-down list.
Before Launch	<p>In this area, specify the activities to be performed before starting the current configuration. The available options are:</p> <ul style="list-style-type: none"> Run Phing target - if this check box is selected, the specified Phing target will be executed prior to running or debugging. To appoint a Phing target, click the Browse button  and select the desired target in the Choose Phing Target to Execute dialog box, that opens. Show Settings - select this check box to have the Run/Debug Configurations dialog box shown every time you launch this run/debug configuration. If this check box

is cleared, the run/debug configuration starts silently.

Share configuration If this check box is selected, the run/debug configurations become available to the other team members.

The shared run/debug configurations are kept in separate xml files under `.idea\runConfigurations` folder, while the local run/debug configurations are kept in the `.idea\workspace.xml`.

See Also

Concepts:

- [Run/Debug Configuration](#)

Procedures:

- [Creating Run/Debug Configuration for Tests](#)
- [Testing](#)
- [Enabling PHPUnit Support](#)
- [Configuring Content Roots](#)

Reference:

- [Directories](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

1.0+

Run/Debug Configuration: PHPUnit on Server

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Use this dialog box to configure running unit tests of PHP applications on a remote server.

PHP unit testing requires the following prerequisites to be fulfilled:

- The [PHPUnit](#) tool should be [installed and configured](#) on your machine.
- The `PEAR` folder should be added to the list of the [project content roots](#).

The dialog box contains two tabs:

- [Configuration](#)
- [Test Groups](#)
- [Remote](#)

Click [here](#) for the description of the options that are common for all run/debug configurations.

Item	Description
Name	In this text box, specify the name of the current run/debug configuration.

Configuration tab

In this tab, specify the unit tests to launch.

Item	Description
All in directory	Select this option to have all the unit tests in a directory launched. In the Directory text box, specify the directory to search the unit test in. Type the path to the directory manually or click the Browse button  and select the desired directory in the Choose Test Directory dialog box, that opens.
All in file	Select this option to have all the unit tests in a file launched. In the File text box, specify the file to search the unit test in. Type the path to the file manually or click the Browse button  and select the desired directory in the Choose Test File dialog box, that opens.
Class or suite	Select this option to have all the unit tests in a test class or test suite launched. <ul style="list-style-type: none"> • In the File text box, specify the file to search the class or suite in. Type the path to the file manually or click the Browse button  and select the desired directory in the Choose Test File dialog box, that opens. • In the Class text box, specify the desired class. Type the class name manually or click the Browse button  and select the desired class in the tree view, that opens.
Method	Select this option to have a specific test method launched. <ul style="list-style-type: none"> • In the File text box, specify the file to search for the test method in. Type the file name manually or click the Browse button  and select the desired file in the tree view, that opens. • In the Method text box, specify the desired method.
XML file	Select this check box to run testing according to the configuration defined in a dedicated XML file. In the Use XML configuration file text box, specify the location of the file to use. Type the path manually or click the Browse button  and appoint the desired file in the Choose Test File dialog box, that opens.
Custom working directory	In this text box, specify the test execution directory of your choice. Type the path manually or click the Browse button  and select the desired folder in the Select Path dialog box that opens.

Note

By default, the working directory for test execution is the one that contains the test.

Test groups tab

In this tab, set additional limits for tests to run.

Item	Description
Include/exclude groups	<p>When this check box is selected, PhpStorm displays a list of all test groups that are defined in the specified scope.</p> <ul style="list-style-type: none"> To have the tests of a group executed, select the Include check box next to the desired group. To skip the tests of a group, select the Exclude check box next to the desired group.
Custom working directory	<p>In this text box, specify the PHP script execution directory of your choice. Type the path manually or click the Browse button and select the desired folder in the Select Path dialog box that opens.</p> <p>Note</p> <p>By default, the working directory for PHP script execution is the one that contains the script.</p>

Remote

In this tab, configure deployment of tests to a remote server.

Item	Description
Remove test files after run	Select this check box to have tests removed from the server upon execution.
Show transfer logs	Select this check box to have test deployment logged.
Server paths mappings	Click this button to open the Override Server Paths Mappings dialog box, where you can map local folders to folders on the server with regard to the file system used on the server.
	<p>Tip</p> <p>PhpStorm detects these mappings automatically but still provides you with this possibility to specify them manually.</p>

Toolbar

Item	Keyboard shortcut	Description
	InsertInsert	Click this button to add new configuration to the list.
	DeleteDelete	Click this button to remove the selected configuration from the list.
	Ctrl+DCommand D	Click this button to create a copy of the selected configuration.
	Edit Defaults	Click this button to edit the default configuration templates. The defaults are used for the newly created configurations.
	UpUp DownDown	Use these buttons to move the selected configuration up and down in the list. The order of configurations in the list defines the order, in which configurations appear in the Run/Debug drop-down list on the main toolbar.

Common options

Item	Description
Defaults	This node in the left-hand pane of the dialog box contains the default run/debug configuration settings. Select the desired configuration to change its default settings in the right-hand pane. The defaults are applied to all newly created run/debug configurations.
Temporary configurations limit	Specify here the maximum number of temporary configurations to be stored and shown in the Select Run/Debug Configuration drop-down list.
Before Launch	<p>In this area, specify the activities to be performed before starting the current configuration. The available options are:</p> <ul style="list-style-type: none"> Run Phing target - if this check box is selected, the specified Phing target will be executed prior to running or debugging. To appoint a Phing target, click the Browse button and select the desired target in the Choose Phing Target to Execute dialog box, that opens. Upload files to Remote Host - select this check box to have files uploaded to a remote host of your choice through FTP or SFTP protocol. To define the folders to be uploaded, click the Browse button and select the desired folder in the Choose Local Paths to Upload dialog box that opens. Show Settings - select this check box to have the Run/Debug Configurations dialog box shown every time you launch this run/debug configuration. If this check box is cleared, the run/debug configuration starts silently.
Share configuration	<p>If this check box is selected, the run/debug configurations become available to the other team members.</p> <p>The shared run/debug configurations are kept in separate xml files under <code>.idea\runConfigurations</code> folder, while the local run/debug configurations are kept in the <code>.idea\workspace.xml</code>.</p>

See Also

- Concepts:
- [Run/Debug Configuration](#)

Procedures:

- [Creating Run/Debug Configuration for Tests](#)
- [Testing](#)
- [Enabling PHPUnit Support](#)
- [Configuring Content Roots](#)

Reference:

- [Directories](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Run/Debug Configuration: XSLT

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

The dialog box consists of the following tabs:

- [Settings](#)
- [Advanced](#)

This section provides descriptions of the [configuration-specific items](#), and the [toolbar](#) and [options](#) that are common for all run/debug configurations.

Item	Description
Name	In this text box, specify the name of the current run/debug configuration.

Settings tab

Item	Description
Input	<p>Use the controls in this area to specify the XML file to process and the script to be executed.</p> <ul style="list-style-type: none"> • XSLT Script File - in this text box, specify the path to the XSLT stylesheet file. Type the path manually or click the Browse button  and select the desired file in the Choose XSLT File dialog box, that opens. • Choose XML Input File - from this drop-down list, select the XML input file to be transformed. The list contains all the XML files that have been associated with the chosen stylesheet via the File Associations functionality. To specify a file, which is not on the list, click the Browse button  and select the desired file in the Choose XML File dialog box, that opens.
Output	<p>Use the controls in this area to configure handling of the script output.</p> <ul style="list-style-type: none"> • Show in Default Console - select this option to have the output displayed in the normal run console, together with any warnings and error messages from the XSLT transformer, as well as messages generated by the script, e.g. by <code>xsl:message</code>. • Show in Extra Console Tab - select this option to have the produced output displayed in an extra, XSLT Output, tab. <p>Note</p> <p>This option is selected by default.</p> <ul style="list-style-type: none"> • Highlight Output As - from this drop-down list, select the file type to highlight the output as. • Save to File - select this option to have the output saved directly to a file. In the text box, specify the name of the target file. Type the path to the file manually or click the Browse button  and select the desired file in the Choose Output File dialog box, that opens. If you type the name of a file that does not exist, PhpStorm will create a file and save the output to it. <ul style="list-style-type: none"> ◦ Open File in Editor After Execution - select this check box to have the file with the output opened in the editor after the script is executed successfully. ◦ Open File in Web Browser After Execution - select this check box to have the file with the output opened in the configured Web browser after the script is executed successfully. <p>Warning</p> <p>The specified file will be overwritten without requesting for confirmation.</p>
Parameters	<p>Use the controls in this area to create and manage a list of parameters to be passed to the script.</p> <ul style="list-style-type: none"> • Add  - click this button to create a new entry. • Remove  - click this button to remove the selected entry from the list. • Name - in this text box, specify the name of the parameter. • Value - in this text box, specify the value of the parameter. <p>Warning</p> <p>The field is mandatory. Parameters without values are not passed to the script. Values are not assigned by default.</p>

Advanced tab

In this tab, configure additional options that are not commonly required in run configurations.

Item	Description
Smart Error Handling	

- Clear this check box to have the console display full error messages including their complete stack traces, when an error occurs during execution.
- Select this check box to suppress showing stacktraces and have the console display only the relevant information about errors.

VM Arguments In this text box, specify optional VM arguments to be passed to the VM where the XSLT script is executed. These can be heap size, garbage collection options, file encoding, etc. If the line of VM arguments is too long, click the  button and type the text in the **VM Arguments** dialog box, that opens.

Working Directory In this text box, specify the working directory to use. Type the path manually or click the **Browse** button  and select the desired folder in the **Working Directory** dialog box, that opens.

Note

If no folder is specified in text box, the working directory will be the one where the XSLT script file is located.

Classpath and JDK In this area, specify the environment to run the script in. By default, it is the module the XSLT file belongs to.

- **From Module** - select this option to execute the script in a specific module. From this drop-down list, select the desired module.

Tip

The full classpath to the selected module is included. This can be required if the script uses custom XSLT Extension Functions.

- **Use JDK** - select this option to choose the JDK without including anything module- or project-related into the classpath.

Tip

It can be useful to explicitly choose a specific JDK to test the script with.

Toolbar

Item	Keyboard shortcut	Description
	InsertInsert	Click this button to add new configuration to the list.
	DeleteDelete	Click this button to remove the selected configuration from the list.
	Ctrl+DCommand D	Click this button to create a copy of the selected configuration.
	Edit Defaults	Click this button to edit the default configuration templates. The defaults are used for the newly created configurations.
	UpUp DownDown	Use these buttons to move the selected configuration up and down in the list. The order of configurations in the list defines the order, in which configurations appear in the Run/Debug drop-down list on the main toolbar.

Common options

Item	Description
Name	In this text box, specify the name of the current run/debug configuration.
Defaults	This node in the left-hand pane of the dialog box contains the default run/debug configuration settings. Select the desired configuration to change its default settings in the right-hand pane. The defaults are applied to all newly created run/debug configurations.
Temporary configurations limit	Specify here the maximum number of temporary configurations to be stored and shown in the Select Run/Debug Configuration drop-down list.
Before Launch	In this area, specify the activities to be performed before starting the current configuration. The available options are: <ul style="list-style-type: none"> • Show Settings - select this check box to have the Run/Debug Configurations dialog box shown every time you launch this run/debug configuration. If this check box is cleared, the run/debug configuration starts silently.
Share configuration	If this check box is selected, the run/debug configurations become available to the other team members. The shared run/debug configurations are kept in separate xml files under <code>.idea\runConfigurations</code> folder, while the local run/debug configurations are kept in the <code>.idea\workspace.xml</code> .

See Also

Reference:

- [XSLT File Associations](#)
- [XSLT](#)

External Links:

- [XSLT Support](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Logs Tab

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Run | Edit Configurations - Logs

In this tab, specify which log files generated while running or debugging should be displayed right in the console.

Item	Description
Is Active	Select this check box to activate the selected log entry.
Skip Content	Select this check box to have the previous content of the selected log skipped.
Log File Entry	This read-only field displays the path to the log entry or its alias.
Add	Click this button to open the Edit Log Files Aliases dialog box, where you can select a new log entry and specify an alias for it.
Remove	Click this button to remove the selected log entry from the list.

See Also

Concepts:

- [Run/Debug Configuration](#)

Procedures:

- [Creating and Editing Run/Debug Configurations](#)

Reference:

- [Run/Debug Configurations](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Run/Debug Configuration: JavaScript Debug

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

In this dialog box, create a configuration to be used for debugging JavaScript sources locally and remotely.

Warning

Debugging of the JavaScript applications is supported only in [Firefox](#) and [Google Chrome](#).

This section provides descriptions of the [configuration-specific items](#), and the [toolbar](#) and [options](#) that are common for all run/debug configurations.

Item	Description	Available for
Name	In this text box, specify the name of the current run/debug configuration.	Both
HTML file	Specify here the path to the HTML file where your script is to be debugged.	Local
Browser	From this drop down list, select the browser (Firefox or Chrome), where your JavaScript application will be debugged.	Both
URL to open	In this text box, specify the URL address to access the application through.	Remote
Remote URLs of local files	In this area, map the local files to be involved in debugging to URL addresses. <ul style="list-style-type: none"> • File/Directory - in this read-only field, select the desired local file or directory in the project tree. • Remote URL - in this text box, type the corresponding absolute URL address. 	Remote

Toolbar

Item	Keyboard shortcut	Description
	InsertInsert	Click this button to add new configuration to the list.
	DeleteDelete	Click this button to remove the selected configuration from the list.
	Ctrl+DCommand D	Click this button to create a copy of the selected configuration.
	Edit Defaults	Click this button to edit the default configuration templates. The defaults are used for the newly created configurations.
	UpUp DownDown	Use these buttons to move the selected configuration up and down in the list. The order of configurations in the list defines the order, in which configurations appear in the Run/Debug drop-down list on the main toolbar.

Common options

Item	Description
Defaults	This node in the left-hand pane of the dialog box contains the default run/debug configuration settings. Select the desired configuration to change its default settings in the right-hand pane. The defaults are applied to all newly created run/debug configurations.
Temporary configurations limit	Specify here the maximum number of temporary configurations to be stored and shown in the Select Run/Debug Configuration drop-down list.
Before Launch	In this area, specify the activities to be performed before starting the current configuration. The available options are:

- **Run Phing target** - if this check box is selected, the specified [Phing](#) target will be executed prior to running or debugging. To appoint a Phing target, click the **Browse** button  and select the desired target in the **Choose Phing Target to Execute** dialog box, that opens.
- **Show Settings** - select this check box to have the **Run/Debug Configurations** dialog box shown every time you launch this run/debug configuration. If this check box is cleared, the run/debug configuration starts silently.

Share configuration If this check box is selected, the run/debug configurations become available to the other team members.

The shared run/debug configurations are kept in separate xml files under `.idea\runConfigurations` folder, while the local run/debug configurations are kept in the `.idea\workspace.xml`.

See Also

- Concepts:
- [Breakpoints](#)
- Procedures:
- [Creating and Editing Run/Debug Configurations](#)
- Reference:
- [Debug Tool Window](#)
- Web Resources:
- <http://www.jetbrains.net/devnet/community/wi> 
 - <http://youtrack.jetbrains.net/issues/WI> 

Run/Debug Configuration: JSTestDriver

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

To get access to this dialog box, [enable](#) the **JSTestDriver** [bundled plugin](#).

In this dialog box, create a configuration to be used for running JavaScript unit tests in the browser against a JSTestDriver server. Configurations of this type enable running unit tests based on the [JSTestDriver Assertion](#) , [Jasmine](#) , and [QUnit](#)  frameworks.

This section provides descriptions of the [configuration-specific items](#), and the [toolbar](#) and [options](#) that are common for all run/debug configurations.

The dialog box consists of the following areas:

- [Name](#)
- [Test](#)
- [Server](#)
- [Toolbar](#)
- [Common Options](#)

Name

In this text box, type the name of the current configuration.

Test

In this area, tell PhpStorm where to find the tests to execute and how to get [test runner configuration files](#)  that define which test files to load and in which order. There are three main approaches:

- Specify the location of one or several previously created configuration files.
- Point at the target test file, test case, or test method, and then specify the location of the corresponding configuration file.
- Point at the target test file, test case, or test method, and then have PhpStorm detect its dependencies automatically and generate a configuration file on this base.

The way to find tests and configuration files is defined in the Test drop-down list. This choice determines the set of other controls in the area.

Item	Description	Available for
Test	In this drop-down list, specify how PhpStorm will get test runner configuration files. <ul style="list-style-type: none"> • All configuration files in directory: select this option to use all the test runner configuration files in a specific folder. • Configuration file: select this option to use a specific test runner configuration file. • JavaScript test file: select this option to have tests from a specific file executed and either use the existing configuration file or have it generated automatically. • Case: select this option to run a specific test case and either use the existing configuration file or have it generated automatically. • Method: select this option to run a specific test method and either use the existing configuration file or have it generated automatically. 	
Directory	In this text box, specify the folder to look for test runner configuration files in. Type the path manually or click the Browse button  and select the required folder in the dialog box, that opens.	All configuration files in directory
Matched configuration files	This read-only field shows a list of all the <code>*jstd</code> and <code>JSTestDriver.conf</code> test runner configuration files detected in the specified folder .	All configuration files in directory
Configuration file	In this text box, specify the test runner configuration file to use. Type the path manually or click the Browse button  and select the required file in the dialog box, that opens.	Configuration file
JS test file	In this text box, specify the JavaScript files with tests to be executed. Type the path manually or click the Browse button  and select the required file in the dialog box, that opens.	JavaScript test file Case Method
Case	In this text box, type the name of the target case from the specified JavaScript file .	Case Method

Method	In this text box, type the name of the target method from the specified test case within the specified JavaScript file .	Method
JSTestDriver configuration file: Generated	Choose this option to have PhpStorm detect the dependencies of the specified JavaScript test file, or test case , or method and create a test runner configuration file on this base.	JavaScript test file Case Method
JSTestDriver configuration file: Custom	Choose this option to have PhpStorm use an existing configuration file for running the specified JavaScript test file, or test case , or method . Specify the file location in the JSTestDriver configuration file text box: type the path to it manually or click the Browse button  and select the required file in the dialog box, that opens.	JavaScript test file Case Method

Server

In this area, appoint the test server to run tests against.

Item	Description
At address	Choose this option to have the test execution handles by a remote test server. In the text box, specify the URL address to access the server through.
Running in IDE	Choose this option to have test execution handles through the JSTestDriver server that comes bundled with PhpStorm and can be launched from it.
Test Connection	Click this button to check that the specified test server is accessible.
	Note
	The server must be running. Start the server from PhpStorm or manually, according to the server-specific instructions.

Toolbar

Item	Keyboard shortcut	Description
	InsertInsert	Click this button to add new configuration to the list.
	DeleteDelete	Click this button to remove the selected configuration from the list.
	Ctrl+DCommand D	Click this button to create a copy of the selected configuration.
	Edit Defaults	Click this button to edit the default configuration templates. The defaults are used for the newly created configurations.
	UpUp DownDown	Use these buttons to move the selected configuration up and down in the list. The order of configurations in the list defines the order, in which configurations appear in the Run/Debug drop-down list on the main toolbar.

Common options

Item	Description
Defaults	This node in the left-hand pane of the dialog box contains the default run/debug configuration settings. Select the desired configuration to change its default settings in the right-hand pane. The defaults are applied to all newly created run/debug configurations.
Temporary configurations limit	Specify here the maximum number of temporary configurations to be stored and shown in the Select Run/Debug Configuration drop-down list.
Before Launch	In this area, specify the activities to be performed before starting the current configuration. The available options are: <ul style="list-style-type: none"> • Run Phing target - if this check box is selected, the specified Phing target will be executed prior to running or debugging. To appoint a Phing target, click the Browse button  and select the desired target in the Choose Phing Target to Execute dialog box, that opens. • Show Settings - select this check box to have the Run/Debug Configurations dialog box shown every time you launch this run/debug configuration. If this check box is cleared, the run/debug configuration starts silently.
Share configuration	If this check box is selected, the run/debug configurations become available to the other team members. The shared run/debug configurations are kept in separate xml files under <code>.idea\runConfigurations</code> folder, while the local run/debug configurations are kept in the <code>.idea\workspace.xml</code> .

See Also

Procedures:

- [Running JavaScript Unit Tests in Browser](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi> 
- <http://youtrack.jetbrains.com/issues/WI> 

Run/Debug Configuration: Node JS

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

To get access to this dialog box:

1. [Install](#) and [enable](#) the **NodeJS** plugin from the [JetBrains plugin repository](#) .
2. Download and install the [NodeJS](#)  framework.

In this dialog box, create configurations for running and debugging of **nodeJS** applications locally. "Locally" in the current context means that PhpStorm itself starts the **NodeJS** framework installed on your computer, whereupon initiates a running or debugging session.

This section provides descriptions of the [configuration-specific items](#), and the [toolbar](#) and [options](#) that are common for all run/debug configurations.

Item	Description												
Name	In this text box, type the name of the current configuration.												
Path to Node	In this field, specify the NodeJS installation home. Type the path to the NodeJS executable file manually, or click the Browse button  and select the location in the dialog box, that opens. If you have appointed one of the installations as default , the field displays the path to its executable file.												
Node parameters	In this text box, type the NodeJS-specific command line options to be passed to the NodeJS executable file. The acceptable options are: <table border="0" style="width: 100%;"> <tr> <td style="vertical-align: top;"><code>--debug=<port for connect to debugger remotely></code></td> <td>Specify this option to enable remote debugging of the application without re-starting the NodeJS server.</td> </tr> <tr> <td style="vertical-align: top;"><code>-v, --version</code></td> <td>Print the current version of NodeJS.</td> </tr> <tr> <td style="vertical-align: top;"><code>-e, --eval script</code></td> <td>Evaluate script.</td> </tr> <tr> <td style="vertical-align: top;"><code>--v8-options</code></td> <td>Print v8 command line options.</td> </tr> <tr> <td style="vertical-align: top;"><code>--vars</code></td> <td>Print various compiled-in variables.</td> </tr> <tr> <td style="vertical-align: top;"><code>--max-stack-size=val</code></td> <td>Set max v8 stack size (bytes).</td> </tr> </table>	<code>--debug=<port for connect to debugger remotely></code>	Specify this option to enable remote debugging of the application without re-starting the NodeJS server.	<code>-v, --version</code>	Print the current version of NodeJS.	<code>-e, --eval script</code>	Evaluate script.	<code>--v8-options</code>	Print v8 command line options.	<code>--vars</code>	Print various compiled-in variables.	<code>--max-stack-size=val</code>	Set max v8 stack size (bytes).
<code>--debug=<port for connect to debugger remotely></code>	Specify this option to enable remote debugging of the application without re-starting the NodeJS server.												
<code>-v, --version</code>	Print the current version of NodeJS.												
<code>-e, --eval script</code>	Evaluate script.												
<code>--v8-options</code>	Print v8 command line options.												
<code>--vars</code>	Print various compiled-in variables.												
<code>--max-stack-size=val</code>	Set max v8 stack size (bytes).												
Path to Node App JS file	In this field, specify the full path to the NodeJS file to start running or debugging from.												
Application parameters	In this text box, type the NodeJS-specific arguments to be passed to the starting NodeJS application file through the process.argv  array.												
Working directory	In this field, specify the location of the files referenced from the starting NodeJS application file , for example, <code>includes</code> . If this file does not reference any other files, just leave the field empty. Choose the folder from the drop-down list, or type the path manually, or click the Browse button  and select the location in the dialog box, that opens.												
Environment variables	In this text box, specify the environment variables  for the NodeJS executable file, if applicable. Click the Browse button  to the right of the field and configure a list of variables in the Environment Variables dialog box, that opens: <ul style="list-style-type: none"> • To define a new variable, click the Add toolbar button  and specify the variable name and value. • To discard a variable definition, select it in the list and click the Delete toolbar button . • Click OK, when ready The definitions of variables are displayed in the Environment variables read-only field with semicolons as separators. The acceptable variables are: <table border="0" style="width: 100%; margin-top: 10px;"> <tr> <td style="vertical-align: top;"><code>NODE_PATH</code></td> <td>A <code>:</code>-separated list of directories prefixed to the module search path.</td> </tr> <tr> <td style="vertical-align: top;"><code>NODE_MODULE_CONTEXTS</code></td> <td>Set to <code>1</code> to load modules in their own global contexts.</td> </tr> <tr> <td style="vertical-align: top;"><code>NODE_DISABLE_COLORS</code></td> <td>Set to <code>1</code> to disable colors in the REPL.</td> </tr> </table>	<code>NODE_PATH</code>	A <code>:</code> -separated list of directories prefixed to the module search path.	<code>NODE_MODULE_CONTEXTS</code>	Set to <code>1</code> to load modules in their own global contexts.	<code>NODE_DISABLE_COLORS</code>	Set to <code>1</code> to disable colors in the REPL.						
<code>NODE_PATH</code>	A <code>:</code> -separated list of directories prefixed to the module search path.												
<code>NODE_MODULE_CONTEXTS</code>	Set to <code>1</code> to load modules in their own global contexts.												
<code>NODE_DISABLE_COLORS</code>	Set to <code>1</code> to disable colors in the REPL.												

Toolbar

Item	Keyboard shortcut	Description
	InsertInsert	Click this button to add new configuration to the list.
	DeleteDelete	Click this button to remove the selected configuration from the list.
	Ctrl+DCommand D	Click this button to create a copy of the selected configuration.
	Edit Defaults	Click this button to edit the default configuration templates. The defaults are used for the newly created configurations.
	UpUp DownDown	Use these buttons to move the selected configuration up and down in the list. The order of configurations in the list defines the order, in which configurations appear in the Run/Debug drop-down list on the main toolbar.

Common options

Item	Description
Defaults	This node in the left-hand pane of the dialog box contains the default run/debug configuration settings. Select the desired configuration to change its default settings in the right-hand pane. The defaults are applied to all newly created run/debug configurations.
Temporary configurations limit	Specify here the maximum number of temporary configurations to be stored and shown in the Select Run/Debug Configuration drop-down list.
Before Launch	In this area, specify the activities to be performed before starting the current configuration. The available options are: <ul style="list-style-type: none"> • Run Phing target - if this check box is selected, the specified Phing target will be executed prior to running or debugging. To appoint a Phing target, click the Browse button  and select the desired target in the Choose Phing Target to Execute dialog box, that opens. • Show Settings - select this check box to have the Run/Debug Configurations dialog box shown every time you launch this run/debug configuration. If this check box is cleared, the run/debug configuration starts silently.
Share configuration	If this check box is selected, the run/debug configurations become available to the other team members. The shared run/debug configurations are kept in separate xml files under <code>.idea\runConfigurations</code> folder, while the local run/debug configurations are kept in the <code>.idea\workspace.xml</code> .

See Also

- Concepts:
- [Running and Debugging](#)

Procedures:

- [Running and Debugging Node.JS](#)
- [NodeJS](#)
- [Running](#)
- [Debugging](#)
- [Creating and Editing Run/Debug Configurations](#)

Reference:

- [Run/Debug Configuration: Node JS Remote Debug](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

JavaScript Support:

- [Running and Debugging JavaScript](#)

Run/Debug Configuration: Node JS Remote Debug

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

To get access to this dialog box:

1. [Install](#) and [enable](#) the **NodeJS** plugin from the [JetBrains plugin repository](#).
2. Download and install the [NodeJS](#) framework.

In this dialog box, create configurations for debugging already running [nodeJS](#) applications. This approach gives you the possibility to re-start a debugging session without re-starting the NodeJS server.

This section provides descriptions of the [configuration-specific items](#), and the [toolbar](#) and [options](#) that are common for all run/debug configurations.

Item	Description
Name	In this text box, specify the name to identify the configuration.
Host	In this text box, specify the host where the application is running.
Debug port	In this text box, specify the port to connect to. Do one of the following: <ul style="list-style-type: none"> • Copy the port number from the information message in the Run tool window that controls the running application. • Copy the port number from the Node parameter text box in the Run/Debug Configuration:NodeJS dialog box.

Toolbar

Item	Keyboard shortcut	Description
	InsertInsert	Click this button to add new configuration to the list.
	DeleteDelete	Click this button to remove the selected configuration from the list.
	Ctrl+DCommand D	Click this button to create a copy of the selected configuration.
	Edit Defaults	Click this button to edit the default configuration templates. The defaults are used for the newly created configurations.
	UpUp DownDown	Use these buttons to move the selected configuration up and down in the list. The order of configurations in the list defines the order, in which configurations appear in the Run/Debug drop-down list on the main toolbar.

Common options

Item	Description
Defaults	This node in the left-hand pane of the dialog box contains the default run/debug configuration settings. Select the desired configuration to change its default settings in the right-hand pane. The defaults are applied to all newly created run/debug configurations.
Temporary configurations limit	Specify here the maximum number of temporary configurations to be stored and shown in the Select Run/Debug Configuration drop-down list.
Before Launch	In this area, specify the activities to be performed before starting the current configuration. The available options are: <ul style="list-style-type: none"> • Run Phing target - if this check box is selected, the specified Phing target will be executed prior to running or debugging. To appoint a Phing target, click the Browse button  and select the desired target in the Choose Phing Target to Execute dialog box, that opens. • Show Settings - select this check box to have the Run/Debug Configurations dialog box shown every time you launch this run/debug configuration. If this check box is cleared, the run/debug configuration starts silently.
Share configuration	If this check box is selected, the run/debug configurations become available to the other team members. The shared run/debug configurations are kept in separate xml files under <code>.idea\runConfigurations</code> folder, while the local run/debug configurations are kept in the <code>.idea\workspace.xml</code> .

See Also

Concepts:

- [Running and Debugging](#)

Procedures:

- [Running and Debugging Node.JS](#)
- [NodeJS](#)
- [Running](#)
- [Debugging](#)
- [Creating and Editing Run/Debug Configurations](#)

Reference:

- [Run/Debug Configuration: Node JS](#)

Web Resources:

- <http://www.jetbrains.com/idea/faq/faq-nodejs.html>
- <http://youtrack.jetbrains.com/issues/WI-10000>

Run/Debug Configuration: Nodeunit

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

To get access to this dialog box:

1. [Install](#) and [enable](#) the **NodeJS** plugin from the [JetBrains plugin repository](#).
2. Download and install the [NodeJS](#) framework.
3. Download and install the [nodeunit](#) testing framework

In this dialog box, create configurations to run unit tests for **NodeJS** applications.

This section provides descriptions of the [configuration-specific items](#), and the [toolbar](#) and [options](#) that are common for all run/debug configurations.

Item	Description
Name	In this text box, type the name of the current configuration.
Node path	In this field, specify the NodeJS installation home. Type the path to the NodeJS executable file manually, or click the Browse button  and select the location in the dialog box, that opens. If you have appointed one of the installations as default , the field displays the path to its executable file.
Working directory	In this text box, specify the folder to find tests under. This can be the project root folder or the parent directory for the <code>test</code> folder. Type the path to the NodeJS executable file manually, or click the Browse button  and select the location in the dialog box, that opens.
Run	From this drop-down list, choose the scope of tests to execute. The available options are: <ul style="list-style-type: none"> • All JavaScript test files in the directory: choose this option to have PhpStorm run all the test files in a folder. In the Directory text box below, specify the path to the test folder relative to the working directory. • JavaScript test file: choose this option to have a specific test executed. In the JavaScript test file text box, type the path to the file relative to the working directory.

Toolbar

Item	Keyboard shortcut	Description
	InsertInsert	Click this button to add new configuration to the list.
	DeleteDelete	Click this button to remove the selected configuration from the list.
	Ctrl+DCommand D	Click this button to create a copy of the selected configuration.
	Edit Defaults	Click this button to edit the default configuration templates. The defaults are used for the newly created configurations.
	UpUp DownDown	Use these buttons to move the selected configuration up and down in the list. The order of configurations in the list defines the order, in which configurations appear in the Run/Debug drop-down list on the main toolbar.

Common options

Item	Description
Defaults	This node in the left-hand pane of the dialog box contains the default run/debug configuration settings. Select the desired configuration to change its default settings in the right-hand pane. The defaults are applied to all newly created run/debug configurations.
Temporary configurations limit	Specify here the maximum number of temporary configurations to be stored and shown in the Select Run/Debug Configuration drop-down list.
Before Launch	In this area, specify the activities to be performed before starting the current configuration. The available options are: <ul style="list-style-type: none"> • Run Phing target - if this check box is selected, the specified Phing target will be executed prior to running or debugging. To appoint a Phing target, click the Browse button  and select the desired target in the Choose Phing Target to Execute dialog box, that opens. • Show Settings - select this check box to have the Run/Debug Configurations dialog box shown every time you launch this run/debug configuration. If this check box is cleared, the run/debug configuration starts silently.
Share configuration	If this check box is selected, the run/debug configurations become available to the other team members. The shared run/debug configurations are kept in separate xml files under <code>.idea\runConfigurations</code> folder, while the local run/debug configurations are kept in the <code>.idea\workspace.xml</code> .

See Also

Concepts:

- [Testing](#)
- [NodeJS](#)
- [Creating and Editing Run/Debug Configurations](#)

Procedures:

- [Unit Testing Node.JS](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

JavaScript Support:

- [Running JavaScript Unit Tests in Browser](#)

Select Path

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

This dialog is used in numerous situations, when it is necessary to point to a particular file or directory.

Item	Tooltip and Shortcut	Description
	Home Ctrl+1Command 1	Click this button to navigate to the user home directory.
	Project Ctrl+2Command 2	Click this button to navigate to the project home directory.
	Refresh Ctrl+Alt+YCommand Alt Y	Click this button to refresh the displayed tree.
	Show hidden files and directories	Click this button to have hidden files and directories displayed in the tree.
	Hide Path/Show Path	Click this button to display/hide the text box with the path to the selected file.

See Also

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Scopes

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

A [scope](#) is a set of files to which various operations apply. Using this dialog, you can define scopes for the various PhpStorm actions, for example, [Find Usages](#), or [Code Inspection](#).

In this section:

- [Main toolbar](#)
- [Options](#)
- [Legend of the project tree view](#)

Main toolbar

Item	Description
	Click this button to add a new local or shared scope.
	Click this button to delete the selected scope from the list.
	Click this button to create a copy of the selected scope.
	Click this button to have the selected local scope saved as shared or a selected shared scope as local.
	Use these buttons to move the scopes up and down in the list.

Options

Item	Description
Name	In this text box, specify the scope name.
Pattern	In this text box, specify the pattern to define the current scope using the script-like language described in the Scope Language Syntax Reference subsection. Type or edit the pattern manually or choose the desired files in the Project Tree View - PhpStorm will generate the corresponding pattern.

Warning

Storing empty or incorrect patterns is not allowed. In such cases, you will be prompted with the `Syntax Error` warning.

- Project tree view** The tree view contains all the files available in your project. In the view, select the desired files to be included in the current scope and have the scope definition pattern generated automatically. The message on the toolbar shows the total number of available files and the number of files included in the scope. See also the [color legend](#) below. Use the [toolbar buttons](#) above the pane to change the view presentation.
- Include** Click this button to have the selected element included in the scope. The corresponding expression is automatically generated and added to the expression in the **Pattern** text box.

Tip

If the current element is a folder, the nested subfolders are ignored.
- Include Recursively** Click this button to have the selected folder included in the scope, together with the nested subfolders. The corresponding expression is automatically generated and added to the expression in the **Pattern** text box.
- Exclude** Click this button to have the selected element excluded from the scope. The corresponding expression is automatically added to the **Pattern**. If the current element is a folder, the nested subfolders are ignored.
- Exclude Recursively** Click this button to have the selected folder excluded from the scope, together with the nested subfolders. The corresponding expression is automatically added to the **Pattern** field.
-  Use this drop-down box to define how you want the project files to be displayed in the tree view. The available options are:

 - **Project**
 - **Packages**
-  When the button is pressed, all the packages are displayed as a single-level tree view. This enables you to find a package somewhere deep within the project by its name without going through the entire tree hierarchy.
-  Click this button to have the selected package prefix displayed as a single package.
-  When this button is pressed, source files in the folders are hidden. Otherwise they are displayed explicitly in the tree view.
-  When the button is pressed, items in the tree-view are shown below the corresponding module nodes. Otherwise, the project items are shown below the corresponding package (like a source path with packages).
-  When the button is pressed, the package structure of the scope is displayed.
-  When the button is pressed, the tree shows only the elements that are included in the scope.

Legend of the project tree view

Item	Description
Green	Folders and files included in scope.
Black	Folders and files excluded from scope.
Blue	Folders that contain both excluded and included files and subfolders.

See Also

- Concepts:
- [Scope](#)
- Procedures:
- [Inspecting Source Code](#)
- Reference:
- [Scope Language Syntax Reference](#)
 - [Inspections](#)
 - [File Colors](#)
- Web Resources:
- <http://www.jetbrains.com/devnet/community/wi>
 - <http://youtrack.jetbrains.com/issues/WI>

Specify Code Duplication Analysis Scope

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Code | Locate Duplicates

Use this dialog box to launch the search for duplicated code fragments in the specified scope.

Item	Description
Whole project	Select this option to perform analysis for the whole project.
Module <name>	Select this option to have PhpStorm analyze the module that is currently selected in the Project tool window.
File <name>	Select this option to have PhpStorm analyze the file that is currently selected in the Project tool window or opened in the editor.
Custom scope	Select this option to define a custom scope to apply analysis to. Choose a pre-defined scope from the drop-down list or click the Browse button  and define a custom scope using the Scopes dialog box.

Tip

Use a special [language](#) to define a scope.

Include Test Sources Select this check box to perform analysis on the test sources.

See Also

Procedures:

- [Analyzing Duplicates](#)

Reference:

- [Duplicates Tool Window](#)
- [Code Duplication Analysis Settings](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Specify Inspection Scope Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Code | **Inspect Code**

Use this dialog box to define the scope to apply inspection to and the profile against which the source code should be inspected.

Item	Description
Whole project	Select this option to perform analysis for the whole project.
File <name>	Select this option to have PhpStorm analyze the file that is currently selected in the Project tool window or opened in the editor.
Directory <name>	Select this option to have PhpStorm analyze the folder that is currently selected in the Project tool window.
Uncommitted files	Select this option to have PhpStorm analyze only files that have not been committed to the version control system. Use the drop-down list to further limit the analysis scope. The available options are: <ul style="list-style-type: none"> • All - select this option to have files from all changelists analyzed. • Default - select this option to have PhpStorm analyze only files from the Default changelist.
Custom scope	Select this option to define a custom scope to apply analysis to. Choose a pre-defined scope from the drop-down list or click the Browse button  and define a custom scope using the Scopes dialog box.
Tip	Use a special language to define a scope.
Include Test Sources	Select this check box to perform analysis on the test sources.

See Also

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Settings Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: 

File | **Settings**

Ctrl+Alt+S Meta Comma



PhpStorm | **Preferences**

Ctrl+Alt+S Meta Comma



This section provides descriptions of the main [controls](#) of the dialog. For project-specific options refer to the section [Project Settings](#); for the options that pertain to your workspace, refer to [IDE Settings](#).

Item	Description
Search	Enter a search keyword in the text area. While typing the search string, the list of options in the dialog reduces to the matching occurrences.
	Click this button to clear the search area.
OK	Apply changes and close the dialog box.
Cancel	Discard changes and close the dialog box.
Apply	Apply changes and leave the dialog box opened.

[Help](#) [Show reference page.](#)

See Also

Procedures:

- [Configuring IDE Settings](#)
- [Configuring Project Settings](#)

Reference:

- [Project Settings](#)
- [IDE Settings](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
 - <http://youtrack.jetbrains.net/issues/WI> 
-

Project Settings

Previous | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: 

File | Settings | Project Settings <project name>

Ctrl+Alt+S Meta Comma



PhpStorm | Preferences | Project Settings <project name>

Ctrl+Alt+S Meta Comma



In this section:

- [Code Style](#)
- [Command Line Tool Support](#)
- [Directories](#)
- [Deployment](#)
- [File Colors](#)
- [File Encodings](#)
- [JavaScript Libraries](#)
- [PHP](#)
- [Language Injections](#)
- [Schemas and DTDs](#)
- [Spelling](#)
- [SQL Dialects](#)
- [Template Data Languages](#)
- [Tasks](#)
- [Version Control](#)
- [XSLT File Associations](#)
- [Inspections](#)

See Also

Procedures:

- [Accessing Project Settings](#)
- [Configuring IDE Settings](#)
- [Configuring Project Settings](#)

Reference:

- [Project Structure](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
 - <http://youtrack.jetbrains.net/issues/WI> 
-

Code Style

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: 

File | Settings | Code Style

Ctrl+Alt+S Meta Comma



PhpStorm | Preferences | Code Style

Ctrl+Alt+S Meta Comma



Use this page to customize the code style at the `global` level or at the `project` level. Settings at the global level are automatically applied every time PhpStorm generates, refactors, or reformats your code.

Settings at the project level apply to the current project only.

Common options

Item	Description	Level
Use	Click this drop-down list to choose the level of configuring the code style settings. The available options are: <ul style="list-style-type: none"> Global settings: Select this option to use global code style options for the project. Per project settings: Select this option to use your own code style settings configurable at the level of the current project. 	
Export	Click this button to save the current scheme under the specified name in the <code>config\codeStyles</code> folder under the PhpStorm home directory.	Per project settings
Scheme name	From this drop-down list, select the desired scheme.	Global settings
Save as	Click this button to save the current global scheme with a new name.	Global settings
Delete	Click this button to delete the current scheme.	Global settings
	<p>Tip</p> <p>The default scheme cannot be deleted.</p>	
Copy to Project	Click this button to create a copy of the current global scheme to the project level. After creating the copy, PhpStorm suggests to switch to this new scheme at the project level.	Global settings
Set From	This button is only available for language-specific code style settings. Click this button to reveal the list of the languages code styles, and select one to copy style settings from.	Global settings

PhpStorm helps configure the following code styles:

- [General](#)
- [Code Style. CSS](#)
- [Code Style. CoffeeScript](#)
- [Code Style. HTML](#)
- [Code Style. JavaScript](#)
- [Code Style. PHP](#)
- [Code Style. SQL](#)
- [Code Style. XML](#)

Note

The **Preview** section for each language immediately shows results when you change settings.

See Also

Concepts:

- [Project and IDE Settings](#)

Procedures:

- [Configuring Code Style](#)

Reference:

- [Code Style](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

General

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

File | Settings | Code Style | General

Ctrl+Alt+S Meta Comma



PhpStorm | Preferences | Code Style | General

Ctrl+Alt+S Meta Comma



Use this page to configure [line separator and indentation](#) options for the various languages. The results are displayed in the **Preview** pane. On [reformatting source code](#), PhpStorm will apply the specified indentation behavior.

Item	Description
Line Separator (for new files)	Use this drop-down list to specify which line separator is to be used in files created by PhpStorm. The available options are: <ul style="list-style-type: none"> System dependent - choose this option to use the default selection. Unix - choose this option to use the Unix line separator.

- **Windows** - choose this option to use the Windows line separator.
- **Mac** - choose this option to use the Mac line separator.

Right Margin (columns) In this text box, specify the number of columns to be used to display pages in the editor.

Wrap when typing reaches right margin Select this check box to ensure that edited text always fits in the specified right margin.

Default Indent Options

Use Tab Character If this check box is selected, tab characters are used:

- On pressing the `Tab` key
- For indentation
- For code reformatting

Otherwise, spaces are used instead of tabs.

Smart Tabs If this check box is selected, PhpStorm inserts tabs for indentation and reformatting, but fine alignment to a necessary column is done **only** via spaces. This is done in order to preserve visual representation of the source code, when the **Tab Size** is changed.

If this check box is not selected, then spaces are used for alignment if necessary.

If you need to use only tabs, then clear all the alignment options in the [Wrapping and Braces](#) page.

This check box is available if the **Use Tab Character** check box is selected.

Tab Size In this text box, specify the number of spaces included within a tab.

Indent In this text box, specify the number of spaces (or tabs if the **Use Tab Character** check box is selected) to be inserted for each indent level.

Continuation indent In this text box, specify the number of spaces (or tabs if the **Use Tab Character** check box is selected) to be inserted at the next line in case of a construct break.

See Also

Procedures:

- [Configuring Code Style](#)

Reference:

- [Code Style](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Code Style. CSS

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

File | Settings | Code Style | CSS

Ctrl+Alt+S Meta Comma



PhpStorm | Preferences | Code Style | CSS

Ctrl+Alt+S Meta Comma



Use this page to configure code style options for CSS files which PhpStorm will apply on reformatting. View the results in the **Preview** pane.

In this section:

- [Tabs and Indents](#)
- [Other](#)

Tabs and indents

Item	Description
Use Tab Character	<p>If this check box is selected, tab characters are used:</p> <ul style="list-style-type: none"> • On pressing the <code>Tab</code> key • For indentation • For code reformatting <p>Otherwise, spaces are used instead of tabs.</p>
Smart Tabs	<p>If this check box is selected, PhpStorm inserts tabs for indentation and reformatting, but fine alignment to a necessary column is done only via spaces. This is done in order to preserve visual representation of the source code, when the Tab Size is changed.</p> <p>If this check box is not selected, then spaces are used for alignment if necessary.</p> <p>This check box is available if the Use Tab Character check box is selected.</p>
Tab Size	In this text box, specify the number of spaces included in a tab.

- Indent** In this text box, specify the number of spaces (or tabs if the **Use Tab Character** check box is selected) to be inserted for each indent level.
- Continuation indent** In this text box, specify the number of spaces (or tabs if the **Use Tab Character** check box is selected) to be inserted at the next line in case of a construct break.

Other

In this tab, specify the alignment, braces and spaces options to be applied on reformatting.

Item	Description
Braces placement	Use this drop-down list to specify where PhpStorm should place the opening braces of selectors. The available options are: <ul style="list-style-type: none"> At the end of line Next line
Align values	Use this drop-down list to specify how PhpStorm should align attributes and values. The available options are: <ul style="list-style-type: none"> Do not align: select this option to specify alignment on the first character of an attribute name. On value: select this option to specify alignment on the first character of the value of an attribute. On colon
Blank lines between blocks	In this text box, specify the minimum number of sequential blank lines to be retained after reformatting.
Align closing brace with properties	If this check box is selected, the closing brace of the selector will be placed under the list of properties. If this check box is not selected, the closing brace of the selector will be placed under the selector.
Keep single-line blocks	If this check box is selected, the blocks with a single property will be confined to one line. If this check box is not selected, each property will be placed to its own line.
Spaces	Select the check boxes in this area to add a space after the colon delimiting key and value, and before the opening brace of the selector.

See Also

- Procedures:
- [Markup Languages and Style Sheets](#)
 - [Reformatting Source Code](#)

- Reference:
- [Code Style](#)

- Web Resources:
- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>

Code Style. CoffeeScript

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

File | Settings | Code Style | CoffeeScript

Ctrl+Alt+S Meta Comma



PhpStorm | Preferences | Code Style | CoffeeScript

Ctrl+Alt+S Meta Comma



Use this page to configure formatting options for CoffeeScript files. View the result in the **Preview** pane.

In this section:

- [Tabs and Indents](#)
- [Spaces](#)
- [Other](#)

Tabs and indents

Item	Description
Use Tab Character	If this check box is selected, tab characters are used: <ul style="list-style-type: none"> On pressing the <code>Tab</code> key For indentation For code reformatting Otherwise, spaces are used instead of tabs.
Smart Tabs	If this check box is selected, PhpStorm inserts tabs for indentation and reformatting, but fine alignment to a necessary column is done only via spaces. This is done in order to preserve visual representation of the source code, when the <code>Tab Size</code> is changed. If this check box is not selected, then spaces are used for alignment if necessary. This check box is available if the Use Tab Character check box is selected.
Tab Size	In this text box, specify the number of spaces included in a tab.

- Indent** In this text box, specify the number of spaces (or tabs if the **Use Tab Character** check box is selected) to be inserted for each indent level.
- Continuation indent** In this text box, specify the number of spaces (or tabs if the **Use Tab Character** check box is selected) to be inserted at the next line in case of a construct break.

Spaces

Use this tab to specify where you want spaces in your code. To have PhpStorm automatically insert a space at a location, select the check box next to this location in the list. The results are displayed in the **Preview** pane.

Other

Item	Description
Align object properties	From the drop-down list, select the type of objects' alignment: <ul style="list-style-type: none"> Do not align: the attributes in sequential lines will be not aligned. On colon: the attributes in sequential lines will be aligned against the colon. On value: the attributes in sequential lines will be aligned against the value.
Use semicolon to terminate statements	Select this check box to have statements terminated with a semicolon.

See Also

Procedures:

- [Reformatting Source Code](#)

Language and Framework-Specific Guidelines:

- [CoffeeScript Support](#)

Reference:

- [Code Style](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Code Style. HTML

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

File | Settings | Code Style | HTML

Ctrl+Alt+S Meta Comma



PhpStorm | Preferences | Code Style | HTML

Ctrl+Alt+S Meta Comma



Use this page to configure code style options for HTML, JSP, and GSP files. The results are displayed in the **Preview** pane.

For the descriptions of **common options**, refer to the section [Code Style](#). For the description of **indentation settings**, refer to section [General](#).

Item	Description
Keep line breaks	Select this check box to have PhpStorm honor line breaks when reviewing HTML files in the editor.
Keep line breaks in text	Select this check box to have PhpStorm honor line breaks in attributes (for example, lengthy descriptions) when reviewing HTML files in the editor.
Keep blank lines	In this text box, specify the minimum number of sequential blank lines to be retained after reformatting.
Wrap attributes	Use this drop-down list to determine how attribute lines should be wrapped. The available options are: <ul style="list-style-type: none"> Do not wrap - if this option is selected, no special wrapping style is applied to the code. Wrap if long - select this option to have lines going beyond the right margin wrapped with proper indentation. Chop down if long - select this option to have elements in lists that go beyond the right margin wrapped to give one element per line with proper indentation. Wrap always - select this option to have all elements in lists wrapped to give one element per line with proper indentation.
Wrap text	Select this check box to have long lines wrapped according to the code style settings.
Align attributes	Select this check box to have attributes in sequential lines aligned.
Align text	Select this check box to have PhpStorm align the text that occupies several lines within a tag.
Keep white spaces	Select this check box to suppress replacing actual white spaces with tabs.
Spaces	In this area, define the use of spaces for attributes and tag names. <ul style="list-style-type: none"> Around "=" in attribute - select this check box to have spaces added around the "=" symbol in attributes. After tag name - select this check box to have spaces added after tag names. In empty tag - select this check box to have spaces added in empty tags.
Insert new line before	This display field shows a list of tags before which a new line should be inserted. Use the button next to the field or press Shift+Enter/Shift Enter to

Remove new line before	open the Insert New Line Before Tags dialog box, where you can edit the list of tags. This display field shows a list of tags before which a break line should be removed. Use the button  next to the field or press <code>Shift+EnterShift Enter</code> to open the Remove Line Breaks Before Tags dialog box, where you can edit the list of tags.
Do not indent children of	This display field shows a list of tags whose children should not be indented. Use the button  next to the field or press <code>Shift+EnterShift Enter</code> to open the Do Not Indent Children Of dialog box, where you can edit the list of tags.
Or if tag size more than	In this text box, specify the minimum length of a tag in lines starting from which its children are not indented.
Inline elements	This display field shows a list of tags that are presented in the source code in the same line with the other tags. If a tag is removed from the list, the editor automatically moves it to a new line, when you add such tag to the source code. Use the button  next to the field or press <code>Shift+EnterShift Enter</code> to open the Inline Elements dialog box, where you can edit the list of tags.
Keep white spaces inside	This display field shows a list of tags inside which you want the editor to preserve white spaces <i>as is</i> , without any changes. Use the button  next to the field or press <code>Shift+EnterShift Enter</code> to open the Keep Whitespaces Inside dialog box, where you can edit the list of tags.
Don't break if inline content	This display field shows a list of tags that are not to be wrapped if their content is inlined. Use the button  next to the field or press <code>Shift+EnterShift Enter</code> to open the Don't Wrap If Inline Content Only dialog box, where you can edit the list of tags.

See Also

Procedures:

- [Markup Languages and Style Sheets](#)

Reference:

- [Code Style. XML](#)
- [Code Style](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Code Style. JavaScript

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

File | Settings | Code Style | JavaScript

Ctrl+Alt+S Meta Comma



PhpStorm | Preferences | Code Style | JavaScript

Ctrl+Alt+S Meta Comma



Use this page to configure formatting options for JavaScript files. View the result in the **Preview** pane.

In this section:

- [Tabs and Indents](#)
- [Spaces](#)
- [Wrapping and Braces](#)
- [Blank Lines](#)
- [Other](#)

Tabs and indents

Item	Description
Use Tab Character	<p>If this check box is selected, tab characters are used:</p> <ul style="list-style-type: none"> • On pressing the <code>TabTab</code> key • For indentation • For code reformatting <p>Otherwise, spaces are used instead of tabs.</p>
Smart Tabs	<p>If this check box is selected, PhpStorm inserts tabs for indentation and reformatting, but fine alignment to a necessary column is done only via spaces. This is done in order to preserve visual representation of the source code, when the Tab Size is changed.</p> <p>If this check box is not selected, then spaces are used for alignment if necessary.</p> <p>This check box is available if the Use Tab Character check box is selected.</p>
Tab Size	In this text box, specify the number of spaces included in a tab.
Indent	In this text box, specify the number of spaces (or tabs if the Use Tab Character check box is selected) to be inserted for each indent level.
Continuation indent	In this text box, specify the number of spaces (or tabs if the Use Tab Character check box is selected) to be inserted at the next line in case of a construct break.

Spaces

Use this tab to specify where you want spaces in your code. To have PhpStorm automatically insert a space at a location, select the check box next to this location in the list. The results are displayed in the **Preview** pane.

Wrapping and braces

In this tab, customize the code style options, which PhpStorm will apply on [reformatting the source code](#). The left-hand pane contains the list of exceptions (**Keep when reformatting**), and placement and alignment options for the various code constructs (lists, statements, operations, annotations, etc.). The right-hand pane shows preview.

Note

Alignment takes precedence over [indentation options](#).

Keep when reformatting

Use the check boxes to configure exceptions that PhpStorm will make when reformatting the source code.

Wrapping options

The wrapping style applies to the various code constructs, specified in the left-hand pane (for example, method call arguments, or assignment statements).

Item	Description
Wrapping style	<p>From this drop-down list, select the desired wrapping style:</p> <ul style="list-style-type: none"> Do not wrap - when this option is selected, no special wrapping style is applied. <p>Note</p> <p>If this option is selected, the nested alignment and braces settings are ignored.</p> <ul style="list-style-type: none"> Wrap if long - select this option to have lines going beyond the right margin wrapped with proper indentation. Chop down if long - select this option to have elements in lists that go beyond the right margin wrapped so that there is one element per line with proper indentation. Wrap always - select this option to have all elements in lists wrapped so that there is one element per line with proper indentation.

Alignment options

Item	Description
<character(s)> on next line	Select this check box to have the specified character or characters moved to the next line when the lines are wrapped.
New line after <character>	Select this check box to have the code after the specified character moved to a new line.
Place on new line	Use this check box to have the corresponding statements or characters moved to the next line.
Align when multiline	If this check box is selected, a code construct starts at the same column on each next line. Otherwise, the position of a code construct is determined by the current indentation level.
Special else if treatment	<p>If this check box is selected, <code>else if</code> statements are located in the same line.</p> <p>Otherwise, <code>else if</code> statements are moved to the next line to the corresponding indent level.</p>
Indent case branches	If this check box is selected, the <code>case</code> statement is located at the corresponding indent level. Otherwise, <code>case</code> statement is placed at the same indent level with <code>switch</code> .

Braces placement options

Item	Description
Braces placement style	<p>Use this drop-down list to specify the position of the opening brace in <code>class declarations</code>, <code>method declarations</code>, and other types of declarations. The available options are:</p> <ul style="list-style-type: none"> End of line - select this option to have the opening brace placed at the declaration line end. Next line if wrapped - select this option to have the opening brace placed at the beginning of the line after the multiline declaration line. Next line - select this option to have the opening brace placed at the beginning of the line after the declaration line. Next line shifted - select this option to have the opening brace placed at the line after the declaration line being shifted to the corresponding indent level. Next line each shifted - select this option to have the opening brace placed at the line after the declaration line being shifted to the corresponding indent level, and have the next line shifted to the next indent level as well.
Force braces	<p>From this drop-down list, choose the braces introduction method for <code>if</code>, <code>for</code>, <code>while</code>, and <code>do () while</code> statements. The available options are:</p> <ul style="list-style-type: none"> Do not force - select this option to suppress introducing braces automatically. When multiline - select this option to have braces introduced automatically, if a code construct occupies more than one line. Always - select this check box to have braces always introduced automatically.

Blank lines

Use this tab to define where and how many blank lines you want PhpStorm to retain and insert in your code after reformatting. For each type of location, specify the number of blank lines to be inserted. The results are displayed in the **Preview** pane.

Item	Description
Keep Maximum Blank Lines	In this area, specify the number of blank lines to be kept after reformatting in the specified locations.

Other

Item	Description
Align object properties	<p>From the drop-down list, select the type of objects' alignment:</p> <ul style="list-style-type: none"> Do not align: the attributes in sequential lines will be not aligned.

- On `colon`: the attributes in sequential lines will be aligned against the colon.
- On `value`: the attributes in sequential lines will be aligned against the value.

Use semicolon to terminate statements

Select this check box to have statements terminated with a semicolon.

See Also

Procedures:

- [Reformatting Source Code](#)

Reference:

- [Code Style](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Code Style. PHP

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

File | Settings | Code Style | PHP

Ctrl+Alt+S Meta Comma



PhpStorm | Preferences | Code Style | PHP

Ctrl+Alt+S Meta Comma



Use this page to configure code style options for PHP files. The results are displayed in the Preview pane.

In this section:

- [Tabs and Indents](#)
- [Spaces](#)
- [Wrapping and Braces](#)
- [Blank Lines](#)
- [PHPDoc](#)
- [Other](#)

Tabs and indents

Item	Description
Use Tab Character	<p>If this check box is selected, tab characters are used:</p> <ul style="list-style-type: none"> • On pressing the <code>Tab</code> key • For indentation • For code reformatting <p>Otherwise, spaces are used instead of tabs.</p>
Smart Tabs	<p>If this check box is selected, PhpStorm inserts tabs for indentation and reformatting, but fine alignment to a necessary column is done only via spaces. This is done in order to preserve visual representation of the source code, when the <code>Tab Size</code> is changed.</p> <p>If this check box is not selected, then spaces are used for alignment if necessary.</p> <p>This check box is available if the <code>Use Tab Character</code> check box is selected.</p>
Tab Size	In this text box, specify the number of spaces included in a tab.
Indent	In this text box, specify the number of spaces (or tabs if the <code>Use Tab Character</code> check box is selected) to be inserted for each indent level.
Continuation indent	In this text box, specify the number of spaces (or tabs if the <code>Use Tab Character</code> check box is selected) to be inserted at the next line in case of a construct break.

Spaces

Use this tab to specify where you want spaces in your code. To have PhpStorm automatically insert a space at a location, select the check box next to this location in the list. The results are displayed in the Preview pane.

Wrapping and braces

In this tab, customize the code style options, which PhpStorm will apply on [reformatting the source code](#). The left-hand pane contains the list of exceptions (**Keep when reformatting**), and placement and alignment options for the various code constructs (lists, statements, operations, annotations, etc.). The right-hand pane shows preview.

Note

Alignment takes precedence over [indentation options](#).

Keep when reformatting

Use the check boxes to configure exceptions that PhpStorm will make when reformatting the source code.

Wrapping options

The wrapping style applies to the various code constructs, specified in the left-hand pane (for example, method call arguments, or assignment statements).

Item	Description
Wrapping style	<p>From this drop-down list, select the desired wrapping style:</p> <ul style="list-style-type: none"> • <code>Do not wrap</code> - when this option is selected, no special wrapping style is applied. <p>Note</p> <p>If this option is selected, the nested alignment and braces settings are ignored.</p> <ul style="list-style-type: none"> • <code>Wrap if long</code> - select this option to have lines going beyond the right margin wrapped with proper indentation. • <code>Chop down if long</code> - select this option to have elements in lists that go beyond the right margin wrapped so that there is one element per line with proper indentation. • <code>Wrap always</code> - select this option to have all elements in lists wrapped so that there is one element per line with proper indentation.

Alignment options

Item	Description
<character(s)> on next line	Select this check box to have the specified character or characters moved to the next line when the lines are wrapped.
New line after <character>	Select this check box to have the code after the specified character moved to a new line.
Place on new line	Use this check box to have the corresponding statements or characters moved to the next line.
Align when multiline	If this check box is selected, a code construct starts at the same column on each next line. Otherwise, the position of a code construct is determined by the current indentation level.
Special else if treatment	<p>If this check box is selected, <code>else if</code> statements are located in the same line.</p> <p>Otherwise, <code>else if</code> statements are moved to the next line to the corresponding indent level.</p>
Indent case branches	If this check box is selected, the <code>case</code> statement is located at the corresponding indent level. Otherwise, <code>case</code> statement is placed at the same indent level with <code>switch</code> .

Braces placement options

Item	Description
Braces placement style	<p>Use this drop-down list to specify the position of the opening brace in <code>class</code> declarations, <code>method</code> declarations, and other types of declarations. The available options are:</p> <ul style="list-style-type: none"> • <code>End of line</code> - select this option to have the opening brace placed at the declaration line end. • <code>Next line if wrapped</code> - select this option to have the opening brace placed at the beginning of the line after the multiline declaration line. • <code>Next line</code> - select this option to have the opening brace placed at the beginning of the line after the declaration line. • <code>Next line shifted</code> - select this option to have the opening brace placed at the line after the declaration line being shifted to the corresponding indent level. • <code>Next line each shifted</code> - select this option to have the opening brace placed at the line after the declaration line being shifted to the corresponding indent level, and have the next line shifted to the next indent level as well.
Force braces	<p>From this drop-down list, choose the braces introduction method for <code>if</code>, <code>for</code>, <code>while</code>, and <code>do () while</code> statements. The available options are:</p> <ul style="list-style-type: none"> • <code>Do not force</code> - select this option to suppress introducing braces automatically. • <code>When multiline</code> - select this option to have braces introduced automatically, if a code construct occupies more than one line. • <code>Always</code> - select this check box to have braces always introduced automatically.

Blank lines

Use this tab to define where and how many blank lines you want PhpStorm to retain and insert in your code after reformatting. For each type of location, specify the number of blank lines to be inserted. The results are displayed in the **Preview** pane.

Item	Description
Keep Maximum Blank Lines	In this area, specify the number of blank lines to be kept after reformatting in the specified locations.
Minimum Blank Lines	In the text boxes in this area, specify the number of blank lines to be present in the specified locations.
Warning	These settings do not influence the number of blank lines before the first and after the last item.

PHPDoc

In this tab, configure the code style to be applied inside [PHPDoc](#) comments.

Item	Description
Align parameter names	Select this check box to have the <code>&<paramname></code> elements aligned.
Keep blank lines	Select this check box to suppress removing blank lines automatically.
Blank lines around parameters	Select this check box to have a blank line inserted above and below the section with <code>@param</code> tags.
Blank line before the first tag	Select this check box to have an blank line inserted above the first PHPDoc tag.

Align tag comments Select this check box to have the `description` elements aligned.

Other

Item	Description
Indent code in PHP tags	Select this check box to have the code enclosed in <code><?php></code> tags indented against the opening <code><?php</code> tag.
Align key-value pairs	Select this check box to have the <code>=></code> separators in key-value assignments aligned.
Align consecutive assignments	Select this check box to have the <code>=</code> operators in two or more consecutive assignments aligned.

See Also

Procedures:

- [PHP-Specific Guidelines](#)
- [Configuring Code Style](#)

Reference:

- [Code Style](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Code Style. SQL

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

File | Settings | Code Style | SQL

Ctrl+Alt+S Meta Comma



PhpStorm | Preferences | Code Style | SQL

Ctrl+Alt+S Meta Comma



Use this page to configure formatting options that apply to SQL injections. The results are displayed in the Preview pane.

In this section:

- [Tabs and Indents](#)
- [Other](#)

Tabs and indents

Item	Description
Use Tab Character	<p>If this check box is selected, tab characters are used:</p> <ul style="list-style-type: none"> • On pressing the <code>Tab</code> key • For indentation • For code reformatting <p>Otherwise, spaces are used instead of tabs.</p>
Smart Tabs	<p>If this check box is selected, PhpStorm inserts tabs for indentation and reformatting, but fine alignment to a necessary column is done only via spaces. This is done in order to preserve visual representation of the source code, when the Tab Size is changed.</p> <p>If this check box is not selected, then spaces are used for alignment if necessary.</p> <p>This check box is available if the Use Tab Character check box is selected.</p>
Tab Size	In this text box, specify the number of spaces included in a tab.
Indent	In this text box, specify the number of spaces (or tabs if the Use Tab Character check box is selected) to be inserted for each indent level.
Continuation indent	In this text box, specify the number of spaces (or tabs if the Use Tab Character check box is selected) to be inserted at the next line in case of a construct break.
Indent subdefinitions / indent subclauses	Select these check boxes to add indentation to elements.

Other

Item	Description
Keep line breaks	Select this check box to have PhpStorm honor line breaks when reviewing SQL injected code in the editor.
Keep blank lines	In this text box, specify the minimum number of sequential blank lines to be retained after reformatting.
Spaces	<p>In this area, specify where you want spaces to be inserted in the SQL injected code. To have PhpStorm automatically insert a space at a location, select the corresponding check box. The available options are:</p> <ul style="list-style-type: none"> • Around operators

See Also

Procedures:

- [Using Language Injections](#)
- [Data Sources](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Code Style. XML

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

File | Settings | Code Style | XML

Ctrl+Alt+S Meta Comma



PhpStorm | Preferences | Code Style | XML

Ctrl+Alt+S Meta Comma



Use this page to configure code style options for XML files. The results are displayed in the Preview pane.

In this section:

- [Tabs and Indents](#)
- [Other](#)

Tabs and indents

Item	Description
Use Tab Character	<p>If this check box is selected, tab characters are used:</p> <ul style="list-style-type: none"> • On pressing the <code>Tab</code> key • For indentation • For code reformatting <p>Otherwise, spaces are used instead of tabs.</p>
Smart Tabs	<p>If this check box is selected, PhpStorm inserts tabs for indentation and reformatting, but fine alignment to a necessary column is done only via spaces. This is done in order to preserve visual representation of the source code, when the <code>Tab Size</code> is changed.</p> <p>If this check box is not selected, then spaces are used for alignment if necessary.</p> <p>This check box is available if the <code>Use Tab Character</code> check box is selected.</p>
Tab Size	In this text box, specify the number of spaces included in a tab.
Indent	In this text box, specify the number of spaces (or tabs if the <code>Use Tab Character</code> check box is selected) to be inserted for each indent level.
Continuation indent	In this text box, specify the number of spaces (or tabs if the <code>Use Tab Character</code> check box is selected) to be inserted at the next line in case of a construct break.

Other

Item	Description
Keep line breaks	Select this check box to have PhpStorm honor line breaks when reviewing XML files in the editor.
Keep line breaks in text	Select this check box to have PhpStorm honor line breaks in attributes (for example, lengthy descriptions) when reviewing XML files in the editor.
Keep blank lines	In this text box, specify the minimum number of sequential blank lines to be retained after reformatting.
Wrap attributes	<p>Use this drop-down list to determine how attribute lines should be wrapped. The available options are:</p> <ul style="list-style-type: none"> • <code>Do not wrap</code> - if this option is selected, no special wrapping style is applied to the code. • <code>Wrap if long</code> - select this option to have lines going beyond the right margin wrapped with proper indentation. • <code>Chop down if long</code> - select this option to have elements in lists that go beyond the right margin wrapped to give one element per line with proper indentation. • <code>Wrap always</code> - select this option to have all elements in lists wrapped to give one element per line with proper indentation.
Wrap text	Select this check box to have long lines wrapped according to the code style settings.
Align attributes	Select this check box to have attributes in sequential lines aligned.
Keep white spaces	When this check box is selected, the editor preserves all whitespaces within tags. The same refers also to the indents, and line breaks.
Spaces	<p>In this area, define the use of spaces for attributes and tag names.</p> <ul style="list-style-type: none"> • <code>Around "=" in attribute</code> - select this check box to have spaces added around the "=" symbol in attributes. • <code>After tag name</code> - select this check box to have spaces added after tag names. • <code>In empty tag</code> - select this check box to have spaces added in empty tags.

See Also

Procedures:

- [Markup Languages and Style Sheets](#)

Reference:

- [Code Style. HTML](#)
- [Code Style](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Command Line Tool Support

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

File | Settings | Command Line Tool Support

Ctrl+Alt+S Meta Comma



PhpStorm | Preferences | Command Line Tool Support

Ctrl+Alt+S Meta Comma



The page shows a list of all PHP-specific and custom command line tools that are currently available at the PhpStorm level.

Warning

Tools with [inconsistencies in the definition files](#) are marked with the Invalid description icon .

Use this page to enable integration with third-party PHP command line tools and define your one ones.

Item	Description
Enabled	When this check box is selected, the commands defined within the corresponding framework can be executed from PhpStorm.
Alias	In this text box, specify the character string to use in command calls instead of the full path to the tool. By default, PhpStorm assigns the following aliases: <ul style="list-style-type: none"> • <code>s</code> for Symfony. • <code>zf</code> for Zend Framework.
Tool Path	In this text box, specify the location of the command-line tool.
Add	Click this button to open the Choose Framework to Add dialog box and select the type of command-line tool to enable integration with. The available options are: <ul style="list-style-type: none"> • Zend Framework Tool • Symfony • Custom Framework Depending on your choice, PhpStorm opens one of the following dialog boxes for specifying the location of the selected tool: <ul style="list-style-type: none"> • Zend Framework • Symfony • Framework Settings
Remove	Click this button to have the selected tool removed from the list.
Open definition in editor	Click this button to have PhpStorm open the <code>.xml</code> file with commands of the selected tool in the editor.
Show console in	In this area, specify where you want to enter commands. The available options are: <ul style="list-style-type: none"> • Pop-up - choose this option to have the Command Line Tools Input pane opened in a separate pop-up window and type commands there. • Tool window - choose this option to enter commands in the Input field in the bottom of the dedicated Command Line Tools Console tool window.

See Also

Procedures:

- [Enabling a Command Line Tool](#)
- [Using Command Line Tools](#)
- [PHP-Specific Guidelines](#)

Reference:

- [Command Line Tools Console Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Zend Framework

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

The dialog box opens when you click the **Add** button and choose **Zend Framework Tool** in the **Choose Framework to Add** dialog box. Use the dialog box to configure [Zend Framework](#) support in PhpStorm.

Item	Description
Path to zf tool	In this text box, specify the location of the <code>zf.bat</code> file. Type the path manually or click the Browse button  and choose the desired location in the Select Path dialog box that opens. PhpStorm parses the contents of the specified file for Zend Framework commands.

See Also

Procedures:

- [Enabling a Command Line Tool](#)
- [Using Command Line Tools](#)
- [PHP-Specific Guidelines](#)

Reference:

- [Command Line Tool Support](#)
- [Command Line Tools Console Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Framework Settings

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

The dialog box opens when you click the **Add** button and choose **Custom Framework** in the **Choose Framework to Add** dialog box. Use the dialog box to create your own command line tool.

Item	Description
Framework Name	In this text box, specify the name of the new tool.
Tool path	In this text box, specify the location of the new tool.
Alias	In this text box, type the character string to use in command calls instead of the full path to the tool.
Description	In this text box, provide a brief explanation of the tool functionality.

See Also

Procedures:

- [Enabling a Command Line Tool](#)
- [Using Command Line Tools](#)
- [PHP-Specific Guidelines](#)

Reference:

- [Command Line Tool Support](#)
- [Command Line Tools Console Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Directories

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Use this dialog box to configure content roots of a project.

Item	Tooltip	Description
	Add Content Root	Click this button to open the Select Path dialog box, where you can select the desired folder to be added to the source files as a new root.
	Remove Content Entry	Click this button to have the selected root removed from the list.
	Excluded	Click this button to mark the selected folder as an excluded root.
	Resource Root	Click this button to mark the selected folder as the Web resource root.
	Include	Click this button to have the exclusion mark removed from the selected excluded content root.

See Also

Concepts:

- [Contents](#)

Procedures:

- [Configuring Content Roots](#)
- [Configuring Folders Within a Content Root](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Deployment

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

File | Settings | Deployment

Tools | Deployment | Configuration

Ctrl+Alt+S Meta Comma



PhpStorm | Preferences | Deployment

Tools | Deployment | Configuration

Ctrl+Alt+S Meta Comma



On this page, configure access to local and remote hosts to upload files to.

The page contains:

- The [left-hand pane](#) that shows a list of all the Web server configurations available in PhpStorm. When you select a configuration, the right-hand pane shows the configuration details.
- [Connection tab](#)
- [Mappings tab](#)

Note

Deploying applications on remote hosts, managing their contents, and synchronizing with deployed applications are supported via the `Remote Hosts Access` bundled plugin, which is by default enabled. If not, activate it in the [Plugins](#) page of the [Settings](#) dialog box.

Toolbar and common options

Use the toolbar buttons to manage the list of configurations.

Item	Tooltip and shortcut	Description
	Add InsertInsert	Click this button to open the Add Server dialog box and define a new configuration there.
	Delete DeleteDelete	Click this button to remove the selected configuration from the list.
	Copy Ctrl+D Command D	Click this button to copy the settings of the selected configuration.
	Use as Default	Click this button to have PhpStorm apply the settings of the selected configuration by default during automatic upload of changed files.

See Also

Procedures:

- [Remote Hosts](#)

Reference:

- [Remote Host Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Deployment: Connection Tab

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Use this tab to choose the [way to access the Web server](#) and specify the [connection settings](#).

Item	Description
------	-------------

- Name The text box shows the configuration name specified in the [Add Server](#) dialog box. Edit the configuration name, if necessary.
- Access type From this drop-down list, choose the way to access the server. The available options are:
- **FTP** - choose this option to have PhpStorm access the server via the [FTP file transfer protocol](#).
 - **SFTP** - choose this option to have PhpStorm access the server via the [SFTP](#) file transfer protocol.
 - **Mounted folder** - choose this option to have PhpStorm access the server via a [mounted folder](#).
 - **Local** - choose this option to create a configuration with a Web server running on your local machine.

Upload/download project files

In this area, specify the settings for accessing the server to upload and download files to and from.

Warning

The set of controls in the area depends on the chosen server access type.

Item	Description	Available for
FTP/SFTP host	In this text box, specify the host name of the FTP/SFTP server to upload the files to.	FTP, SFTP
Port	In this text box, specify the port to use. The default values are: <ul style="list-style-type: none"> • 21 for FTP • 22 for SFTP 	FTP, SFTP
Root path	In this text box, specify the root folder for uploading/downloading files relative to the home folder of the FTP/SFTP server. Type the path manually, click the Browse button  and select the desired folder in the Choose Root Path dialog box that opens, or click the Autodetect button.	FTP, SFTP
Autodetect	Click this button to have PhpStorm detect the user home folder settings on the FTP/SFTP server and set up the root path according to them.	FTP, SFTP
User name	In this text box, type your user name for authentication to the server.	FTP, SFTP
Log in as anonymous	Select this check box to enable anonymous access to the server with your email address as password.	FTP, SFTP
Auth type	From this drop-down list, select the client authentication method. The available options are: <ul style="list-style-type: none"> • Password - select this option to use standard authentication through a password. • Key pair (OpenSSH) - select this option to use SSH authentication via a key pair. <p>To apply this authentication method, you need to have your private key on the client machine and your public key on the remote server you connect to. See http://wiki.gnup.com/wiki/How_To_Set_Up_Authorized_Keys for details.</p> <p>Warning</p> <p>PhpStorm supports private keys generated using the OpenSSH utility.</p>	SFTP
Password	In this text box, type your password for authentication to the server.	FTP, SFTP
Private key file	In this text box, specify the location of your private key file.	SFTP
Passphrase	In this text box, specify your authentication passphrase.	SFTP
Save password	Select this check box to have PhpStorm remember the specified password.	FTP, SFTP
Save passphrase	Select this check box to have PhpStorm remember the specified passphrase.	SFTP
Test FTP/SFTP connection	Click this button to check that the specified settings ensure successful connection via FTP/SFTP.	FTP, SFTP
Mounted folder	In this text box, type the path to the mounted folder which is associated with the target server folder.	Mounted folder
Advanced options	Click this button to specify additional uploading settings in the Advanced Options dialog box that opens.	FTP, SFTP
Web server root URL	In this text box, specify the URL address of the Web server root folder .	All
Open	Click this button to make sure that the specified server root URL address is accessible and points at the correct Web page.	All

See Also

Language and Framework-Specific Guidelines:

- [Remote Hosts](#)

Reference:

- [Deployment](#)
- [Remote Host Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Deployment: Mappings Tab

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Use this tab to define mappings between local folders, the folders on the remote server to deploy them to, and the paths relative to the [server root URL](#).

Item	Description
Use this server as default	Click this button to have PhpStorm apply the settings of the selected configuration by default during automatic upload of changed files.
Local Path	In this text box, specify the absolute path to the desired local folder in the project tree. Type the path manually or click the Browse button  and select the desired location in the Choose Local Path dialog box that opens.
Deployment Path	In this text box, specify the folder to upload the selected local folder to. Type the path relative to the specified FTP/SFTP server root path or the specified mounted folder.
	<p>Note</p> <p>The text box is available for FTP, SFTP, and Mounted Folder server configurations.</p>
Web Path	In this text box, specify the path relative to the server document root which is specified in the URL text box.
Add	Click this button to have a new line added to the list of mappings.
Remove	Click this button to remove the selected mapping from the list.

See Also

Language and Framework-Specific Guidelines:

- [Remote Hosts](#)

Reference:

- [Deployment](#)
- [Remote Host Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Deployment: Excluded Paths Tab

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Use this tab to configure a list of local and remote folders that you do not want to be involved in deployment.

Item	Description
Add local path	Click this button to have an empty line added to the list and specify the location of the folder to be protected against deployment. Type the path manually or click the Browse button  and choose the required folder in the Select Path dialog box that opens.
Add deployment path	Click this button to have an empty line added to the list. Click the Browse button  . The Select remote excluded path dialog box that opens shows the data on the host accessed through the selected server configuration. Select the required folder.
Remove path	Click this button to remove the selected item from the list.
	<p>Note</p> <p>The button is only available when a line is selected.</p>

See Also

Language and Framework-Specific Guidelines:

- [Excluding Files and Folders from Deployment](#)
- [Remote Hosts](#)

Reference:

- [Deployment](#)
- [Remote Host Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Add Server Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

The dialog box opens when you click the **Add** toolbar button  on the [Deployment](#) page.

Use the dialog box to create new Web server configurations.

Item	Description
Name	In this text box, type the name of the new Web server configuration.

Protocol From this drop-down list, choose the way to access the server. The available options are:

- **FTP** - choose this option to have PhpStorm access the Web server via the FTP [file transfer protocol](#).
- **SFTP** - choose this option to have PhpStorm access the Web server via the [SFTP](#) file transfer protocol.
- **Mounted folder** - choose this option to have PhpStorm access the Web server on a mounted drive.
- **Local** - choose this option to create a configuration with a Web server running on your local machine.

See Also

Procedures:

- [Remote Hosts](#)
- [PHP-Specific Guidelines](#)

Reference:

- [Deployment](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Options

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

[File](#) | [Settings](#) | [Deployment - Options for Windows and Linux](#)
[PhpStorm](#) | [Preferences](#) | [Deployment - Options for Mac OS](#)
[Tools](#) | [Deployment](#) | [Options](#)

Use this page to specify additional configuration settings for deploying your application to remote hosts.

Tip

The options specified in this dialog box apply to all defined server configurations.

Item	Description
Exclude items by name	In this text box, specify patterns for the names of files and folders that you do not need to be deployed. Use semicolons as delimiters. Wildcards are welcome. The exclusion is applied recursively. This means that if a matching folder has subfolders, the contents of these subfolders are not deployed either.
Operations logging	Use this drop-down list to specify how much detailed logging you need to have. The available options are: <ul style="list-style-type: none"> • Errors only - select this option to have the log show only errors occurred during upload. • Brief - select this option to have all events reflected in the log but without details. • Detailed - select this option to have more details on the upload shown in the log, for example, full file paths.
Stop operation on the first error	Select this check box to have data transfer stopped as soon as an error occurs.
Overwrite up-to-date files	Do one of the following: <ul style="list-style-type: none"> • Select this check box to have all the files uploaded no matter whether they have been changed since the previous upload or not. • Clear the check box to upload only files that have not been changed since the previous upload.
Preserve files timestamps	Select this check box to prevent resetting timestamps of files on upload.
Delete target item if the source one is missing	If this check box is selected, a previously uploaded file will be removed if the file with this name is not involved in the current upload.
Create empty directories	Select this check box to have an empty directory on the server created automatically if a new local directory has been created since the last upload in the source folder.
Prompt when overwriting or deleting local items	Select this check box to have PhpStorm ask you for confirmation before overwriting or deleting local items for synchronization during download.
Upload changed files automatically to the default server	Select this check box to have PhpStorm automatically upload a file to the default server every time the file is updated. Note The default server configuration is appointed on the Deployment page by selecting the desired configuration in the list and clicking the Use as Default toolbar button  .
Override default permissions on files	Select this check box to change the default permissions assigned to uploaded files on remote hosts. Click the Browse button  to open the Files Default Permissions dialog box, where you can manage access to uploaded files on remote hosts by assigning permissions .
Override default permissions on folders	Select this check box to change the default permissions assigned to uploaded folders on remote hosts. Click the Browse button  to open the Folders Default Permissions dialog box, where you can manage access to uploaded folders on remote hosts by assigning permissions .
Warn when uploading over newer file	Use this drop-down list to define the version-control policy to apply when uploading files to remote hosts. Depending on this choice, PhpStorm either checks whether any changes have been made to the corresponding files on the remote host since you downloaded them or just overwrites the remote files. <ul style="list-style-type: none"> • No - choose this option to have the file on the remote host overwritten with its local copy silently. All the changes made to the remote file

since your last synchronization will be abandoned.

- **Compare timestamp and size** - if you choose this option, PhpStorm performs two checks:
 1. Compares the sizes of the local and remote files.
 2. Compares the remote file timestamp set at the moment of the last synchronization with the current remote file timestamp.

If the files differ in their size or the remote file timestamps differ, PhpStorm opens a [Difference Viewer for Files](#), where you can explore and integrate the differences.

Note

This type of check depends on the timezone setting. If the timezone setting on your local machine is different from that on the remote host, the check may be successful even though the file versions actually differ.

- **Compare content** - when this option is chosen, PhpStorm compares the content of the local and remote files. If any diversions are detected, PhpStorm opens a [Difference Viewer for Files](#), where you can explore and integrate the differences.

See Also

Procedures:

- [PHP-Specific Guidelines](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Advanced Options Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

The dialog box opens when you click the **Advanced Options** button in the [Connection](#) tab of the [Deployment](#) page. Use this dialog box to customize deployment by specifying additional upload settings.

Item	Description	Available for
Passive mode	Select this check box to set the client to the passive mode .	FTP
Show and process hidden files	When this check box is selected: <ol style="list-style-type: none"> 1. Hidden files and directories are shown in the Remote Host Tool Window. 2. Hidden files and directories are involved in diff and synchronization operations. <p>The name of a hidden file or directory starts with a dot (.).</p>	FTP
Compatibility mode	Select this check box to ensure compatibility in child file naming with your FTP server.	FTP
	<p>Tip</p> <p>This option is helpful if the remote FTP server reports the following error:</p> <pre>Invalid descendant file name <file name></pre> <p>Note</p> <p>Selecting this option may slow down synchronization with the server.</p>	
Retrieve accurate files timestamps	Use this drop-down list to specify the MDTM FTP command calling policy to retrieve the last-modified time of a given file on the remote host. The available options are: <ul style="list-style-type: none"> • Always - select this option to have MDTM called for every file shown in the Remote Host tool window. • On copy - select this option to have MDTM called in the following cases: <ul style="list-style-type: none"> ◦ To check whether a file is up to date when the Overwrite up-to-date files check box in the Options dialog box is cleared. ◦ To preserve the actual time stamp of a file during download. • Never - select this option to suppress calling MDTM. 	FTP, SFTP
Limit concurrent connections	Select this check box to have PhpStorm restrict the number of connections to be supported simultaneously and specify the maximum number of allowed connections in the text box.	FTP, SFTP

See Also

Language and Framework-Specific Guidelines:

- [Customizing Upload](#)

Reference:

- [Deployment](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Files/Folders Default Permissions Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

The dialog box opens when you select the **Override default permissions on files** or **Override default permissions on folders** check box in the [Options](#) dialog box and click the **Browse** button  next to it.

Use the dialog box to re-assign default server [permissions](#) to owners of files or folders, groups of owners, and other users.

The following identifiers are used for permission types:

- **R** stands for **Read**.
- **W** stands for **Write**.
- **X** stands for **Execute**

Item Description

Owner In this row, specify what an [owner](#) of a file or folder may do by selecting the check boxes below the corresponding identifiers.

Group In this row, specify what a group of owners of a file or folder may do by selecting the check boxes below the corresponding identifiers.

Others In this row, specify what any one else may do by selecting the check boxes below the corresponding identifiers.

Octal In this text box, specify the [octal representation](#) of the specified set of permissions. By default, PhpStorm calculates and re-calculates the value of the field as you select or clear the desired check boxes. You can also specify the octal value manually.

See Also

Procedures:

- [Remote Hosts](#)
- [PHP-Specific Guidelines](#)

Reference:

- [Deployment](#)
- [Advanced Options Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

File Colors

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac 

File | Settings | File Colors

Ctrl+Alt+S Meta Comma



PhpStorm | Preferences | File Colors

Ctrl+Alt+S Meta Comma



Use this dialog box to set background colors for displaying names of project files, folders, and packages of specific scopes. The settings apply to the following UI elements:

- The headers of editor tabs.
- [Navigation lists](#) when one searches for files or classes by their names.

Item Description

Enable File Colors	Select this check box to apply background color settings to navigation lists .
Enable Colors in Editor Tabs	Select this check box to apply background color settings to the headers of editor tabs.
Highlight Non-Project Files	Select this check box to apply background color settings to the headers of editor tabs with files that are outside the project content roots . To specify the background color, click the color indication to the right of the file and choose the desired color in the hexadecimal color table.
Manage Scopes	Click this button to open the Scopes dialog box, where you can define custom scopes for various actions.
Local Colors	Use the controls in this area to configure the color-scope associations to be applied locally. <ul style="list-style-type: none"> • Scope - this read-only field shows the scope to apply the color setting to. • Color - this read-only field shows the color to be applied to the corresponding scope. • Add - click this button to open the Add Color Label dialog box, where you can configure a new color-scope association. • Remove - click this button to remove the selected color-scope association. • Share - click this button to have the selected scope-color association shared among the members of the team.

Note

The selected association will be accordingly moved to the list in the [Shared Colors](#) area.

- **Move up/Move Down** - click these buttons to resort the color-scope associations and thus determine the order in which they are applied.

Shared Colors Use the controls in this area to configure the color-scope associations to be shared among all the members of the team.

- **Scope** - this read-only field shows the scope to apply the color setting to.

- **Color** - this read-only field shows the color to be applied to the corresponding scope.
- **Add** - click this button to open the [Add Color Label](#) dialog box, where you can configure a new color-scope association.
- **Remove** - click this button to remove the selected color-scope association.
- **Unshare** - click this button to have the selected scope-color association applied only locally.

Note

The selected association will be accordingly moved to the list in the [Local Colors](#) area.

- **Move up/Move Down** - click these buttons to resort the color-scope associations and thus determine the order in which they are applied.

See Also

Concepts:

- [Scope](#)

Procedures:

- [Configuring Project Settings](#)
- [Configuring IDE Settings](#)
- [Navigating to Class, File or Symbol by Name](#)

Reference:

- [Editor](#)
- [Scopes](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Add / Edit Color Label Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

[File](#) | [Settings](#) | [File Colors for Windows and Linux](#)
[PhpStorm](#) | [Preferences](#) | [File Colors for Mac OS](#)

The dialog box opens when you click the **Add** button in the [File Colors](#) dialog box. Use this dialog box to configure a new color-scope association.

Item	Description
Scope	<p>From this drop-down list, select the desired scope. The default scopes are:</p> <ul style="list-style-type: none"> • Problems • All • Production • Tests <p>Tip</p> <p>To use custom scopes, click the Manage Scopes button in the File Colors dialog box and define the desired scopes in the Scopes dialog box, that opens.</p>
Color	In this area, select the desired color label to use as the background for displaying the names of files, folders, or packages within the specified scope.
Share Configuration	Select this check box to have the defined scope-color association shared among the members of the team.

See Also

Concepts:

- [Scope](#)

Procedures:

- [Configuring IDE Settings](#)
- [Configuring Project Settings](#)
- [Navigating to Class, File or Symbol by Name](#)

Reference:

- [File Colors](#)
- [Scopes](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

File Encodings

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: 

File | Settings | File Encodings

Ctrl+Alt+S Meta Comma



PhpStorm | Preferences | File Encodings

Ctrl+Alt+S Meta Comma



Use this dialog to configure encoding options for a project and for the entire IDE.

Item	Description
IDE Encoding	Select encoding to be used for PhpStorm.
File/Directory	This column displays the project tree view.
Default encoding	This column displays encoding for a file or directory, if applicable. If encoding is defined within a file, it cannot be configured, and is shown in grey font. If encoding is configurable, click the Default Encoding column for a selected file or directory, and choose encoding from the drop-down list. Encoding information embedded in a file overrides the selected one; encoding information for the nested files or directories overrides that for the outer directories or the whole project.
Autodetect UTF-encoded files	Check this option, if you want PhpStorm to change file encoding to UTF, if at least one character encoded in UTF-8 or UTF-16 is detected in a file.
Default encoding for properties files	Use this drop-down list to define encoding for the properties files in a project. According to the Java API, the <code>load(InputStream / store(OutputStream, String)</code> methods of the <code>java.util.Properties</code> class, use ISO 8859-1 encoding for input/output stream. It is advisable to use this encoding unless you have special reasons to change it. In this case, you can select the desired encoding from the drop-down list; in particular, you can use encoding defined for the whole project.
Transparent native-to-ascii conversion	Select this option to show in properties files the national characters (non-ISO 8859-1), stored as escape sequences. For more details refer to Configuring Encoding for Properties Files .

See Also

Concepts:

- [Encoding](#)

Procedures:

- [Configuring Individual File Encoding](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

JavaScript. Libraries

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

File | Settings | JavaScript - Libraries

Ctrl+Alt+S Meta Comma



PhpStorm | Preferences | JavaScript - Libraries

Ctrl+Alt+S Meta Comma



Use this page to set up additional JavaScript libraries to expand the basic assistance provided through the JavaScript [plugin](#). Note that the JavaScript libraries are global.

Item	Description
Add	Click this button to create a new JavaScript library in the New Library dialog box.
Edit	Click this button to change the name and contents of the selected library in the Edit Library dialog box.
Remove	Click this button to delete the selected library.
Download	Click this button to open the Download Library dialog box, where you can have PhpStorm download and install one of the popular JavaScript-related libraries. The dialog box shows a list of available libraries with indications of their versions and URL addresses.

New library / edit library dialog box

Item	Description
Name	Specify the library name. In this section, set up the library contents.
Files	
Attach	Click this button to select a JavaScript file or directory from the file system.
Detach	Click this button to remove the selected file or directory from a library.
Name	This read-only column shows the name of the selected library file or the names of relevant library files from the selected directory.
Type	Click the column to show the drop-down list of the available versions of library files or directories: debug or release.

Tip

PhpStorm enables you to create a library containing just one .js file, if this file is located on the Internet and can be accessed over HTTP. If you refer to a JavaScript library that is not yet available locally, but is available online, use the Download library [intention action](#):



The library will be placed to the user home directory, and will appear in the list of configured libraries in the JavaScript - Libraries page of the Settings dialog.

See Also

Concepts:

- [Scope](#)

Procedures:

- [Configuring JavaScript Libraries](#)
- [Viewing Inline Documentation](#)

Language and Framework-Specific Guidelines:

- [JavaScript-Specific Guidelines](#)

Reference:

- [JavaScript. Usage Scope](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

JavaScript. Usage Scope

Previous | Next | [See Also](#) | [Comments](#) | Shortcuts: Mac

File | Settings | JavaScript - Libraries - Usage Scope

Ctrl+Alt+S Meta Comma



PhpStorm | Preferences | JavaScript - Libraries - Usage Scope

Ctrl+Alt+S Meta Comma



Use this page to define the scopes where the JavaScript libraries should apply for code completion, highlighting and navigation. The scopes may cover the whole project, directories and even individual files. This helps make JavaScript code completion more precise, and avoid too long suggestion lists.

Item	Description
File/Directory	This column displays the project tree view.
Library	This column displays libraries for a file or directory, if applicable. If a library can be specified for a certain node of the project tree view, click the Library column for a selected file or directory, and choose the desired library from the list of available libraries. If JavaScript libraries are not applicable to a particular node, 'N/A' is shown in grey font.

See Also

Concepts:

- [Scope](#)

Procedures:

- [Configuring JavaScript Libraries](#)

Reference:

- [JavaScript. Libraries](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

PHP

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

File | Settings | PHP

PhpStorm | Preferences | PHP

Note

The page is available only when the PHP bundled plugin is enabled. By default the plugin is active. If not, enable it in the Plugin Configuration Wizard or the Plugins page of the Settings dialog box.

Use this dialog box to configure PHP development and unit testing support in the project level by choosing one of the PHP installations configured at the PhpStorm level.

Item	Tooltip and Shortcut	Description
Interpreter		From this drop-down list, choose the installation to use in the current project. The list contains all the PHP installations configured at the PhpStorm level.
	Reload	Click this button to make sure that the configuration you have chosen points at the relevant installation. If no PHP executable is detected at the specified directory, PhpStorm displays the corresponding error message.
	Show phpinfo	Click this button to examine the installation details and view the full list of installation settings in a separate window. Actually, PhpStorm checks the installation and displays the result of executing the phpinfo command.
	Shift+Enter Shift Enter	Click this button next to the Interpreter drop-down list to create a new PhpStorm-wide PHP installation configuration in the Interpreters dialog box, that opens.
Include path		<p>This area shows a list of paths to PHP-related items below the PHP home directory. The specified include paths will be used:</p> <ul style="list-style-type: none"> By the require(), include(), fopen(), file(), readfile(), and file_get_contents() functions when looking for files to use. By PhpStorm when resolving references to included files. <p>Use the Add and Remove buttons to manage the contents of the list. Use the Up and Down buttons to change the order of items in the list.</p>

See Also

Reference:

- [Interpreters](#)

PHP Support:

- [Enabling PHP Support](#)
- [PHP-Specific Guidelines](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wiki/>
- <http://youtrack.jetbrains.com/issues/WI>

Interpreters

Previous | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

File | Settings | PHP

PhpStorm | Preferences | PHP

The dialog box opens when you click the Browse button next to the Interpreter drop-down list in the Development environment section of the PHP page.

Note

The page is available only when the PHP bundled plugin is enabled. By default the plugin is active. If not, enable it in the Plugin Configuration Wizard or the Plugins page of the Settings dialog box.

Use this dialog box to configure PHP installations at the PhpStorm level.

Item	Tooltip and Shortcut	Description
Name		In this text box, type the identifier to distinguish the installation from others, for example, php_installation_<version>.
PHP Home		In this text box, specify the folder where the PHP engine, the configuration file, and other PHP-related items are located. the path manually or click the Browse button and choose the desired folder in the Select Path dialog box that opens. PhpStorm detects the PHP version installed on your computer.
	Reload	Click this button to check that the specified PHP home directory actually contains a PHP executable file. If no PHP executable is detected at the specified location, PhpStorm displays the corresponding error message.
	Show phpinfo	This read-only field displays the current PHP installation version.
Debugger		From this drop-down list, select the debugging engine to use. The available options are: <ul style="list-style-type: none"> XDebug Zend Debugger
Configuration options		Use this text box to customize the configuration settings of the installation by compose a string of configuration directives to be passed through the <code>-d</code> command line option and thus add new entries to the php.ini file.
		Click this button to open the Configuration Directives dialog box and create a list of new php.ini entries there.

Configuration directives dialog

Item	Tooltip and Shortcut	Description
Name		In this text box, type the name of the new entry.
Value		In this text box, type the value of the new entry.
	Add Alt+InsertCommand N	Click this button to have a new line added to the list and specify the name and value of a new entry there.
	Remove Alt+DeleteMeta Delete	Click this button to remove the selected entry from the list.
	Up / Down Alt+UpCommand Up / Alt+DownCommand Down	Use these buttons to move the selected entry up or down lin the list. The order of entries in the list determine the order in which they are passed through the -d command line option .

See Also

Reference:

- [PHP](#)

PHP Support:

- [Enabling PHP Support](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Smarty

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

[File](#) | [Settings](#) | [PHP](#) | [Smarty](#)

[PhpStorm](#) | [Preferences](#) | [PHP](#) | [Smarty](#)

The dialog box opens when you click the **Browse** button  next to the **Interpreter** drop-down list in the **Development environment** section of the [PHP](#) page.

Note

The page is available only when the **PHP** bundled plugin is enabled. By default the plugin is active. If not, [enable](#) it in the [Plugin Configuration Wizard](#) or the [Plugins](#) page of the [Settings](#) dialog box.

Use this dialog box to specify the delimiters to enclose [Smarty](#) tags in.

Item	Description
Smarty left delimiter	In this text box, specify the desired opening delimiter. By default, the field shows the opening curly brace {.
Smarty right delimiter	In this text box, specify the desired closing delimiter. By default, the field shows the closing curly brace }.
Use Smarty 3 whitespace policy	When this check box is selected, delimiters followed by whitespace will not be parsed as template tags.
	Note
	The check box is available when Smarty 3.0 or higher is used.

See Also

Procedures:

- [PHP-Specific Guidelines](#)

Reference:

- [PHP](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Servers

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: 

[File](#) | [Settings](#) | [PHP](#) | [Debug](#)

[PhpStorm](#) | [Preferences](#) | [PHP](#) | [Debug](#)

Note

The page is available only when the PHP bundled plugin is enabled. By default the plugin is active. If not, [enable](#) it in the [Plugin Configuration Wizard](#) or the [Plugins](#) page of the [Settings](#) dialog box.

On this page, configure access to local and remote Web servers to debug PHP applications on.

The [left-hand pane](#) of the page shows a list of all the Web server debug configurations available in PhpStorm. When you select a configuration, the right-hand pane shows the [configuration details](#).

Toolbar and common options

Use the toolbar buttons to manage the list of configurations.

Item	Tooltip and shortcut	Description
	Add InsertInsert	Click this button to define a new configuration.
	Delete DeleteDelete	Click this button to remove the selected configuration from the list.

Configuration details

In this area, specify the connection parameters and mappings to be used during debugging sessions.

Item	Description
Name	In this text box, type the name of the server debug configuration.
Host	In this text box, type the name of the host where the target application is deployed.
Port	In this text box, type the port to connect to the specified host through. If you are using localhost on your machine, this setting should correspond with the port specified in the configuration file of the local Web server where the application will be executed or debugged.
Debugger	From this drop-down list, select the debug engine to use. The available options are: <ul style="list-style-type: none"> XDebug Zend Debugger
Use path mappings	<ul style="list-style-type: none"> Clear this check box to have PhpStorm suggest mappings between files and folders deployed on the server and their local copies. When you start a debugging session, PhpStorm will try to detect the local copies of the application files on the server. The suggestions are displayed in a dialog box where PhpStorm asks you to confirm or edit suggested mappings. Select this check box, to specify correspondence between files on the server and their local copies manually. Then map files and folders on the server to their local copies using the Path on server and File/Directory fields respectively.
File / Directory	This read-only field displays the local folders of the current project in a tree view. Select a file or a folder to be used as the local copy.
Path on server	In this field, specify the file or folder on the target server to which the selected local file or folder corresponds. Type the path manually or select it from the drop-down list. When this file or a file from this folder is being processed during a debugging session, PhpStorm opens its local copy chosen in the File / Directory field.

See Also

Procedures:

- [PHP Debugging Session](#)
- [Creating a PHP Web Application Debug Configuration](#)

Reference:

- [Debug](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Debug

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

[File](#) | [Settings](#) | [PHP](#) | [Debug](#)

[PhpStorm](#) | [Preferences](#) | [PHP](#) | [Debug](#)

Note

The page is available only when the PHP bundled plugin is enabled. By default the plugin is active. If not, [enable](#) it in the [Plugin Configuration Wizard](#) or the [Plugins](#) page of the [Settings](#) dialog box.

Use this page to configure the behaviour of the XDebug and Zend Debugger debugging tools.

Item	Description
Values tooltip delay	In this text box, specify the time period to pass between you hover the mouse pointer over a variable or another code element and a tooltip with the value of this element appears. This setting helps you avoid the situation when tooltips appear immediately as you move the mouse pointer through the code.
Safe evaluation mode	<ul style="list-style-type: none"> When this check box is selected, PhpStorm checks that the expression or code fragment to be evaluated does not contain any undefined elements and informs you about any discrepancies detected. If the check box is cleared, an exception appears if PhpStorm encounters any undefined elements during evaluation.

- Local debug** In this area, customize local debugging, when the processed files are on your machine.
 - **Pass required configuration options through command line (still need to enable debug extension manually):** select this check box to have debugger configuration options passed through a command line.
- Remote debug** In this area, customize remote debugging, which means that the paths to the files processed by the server differ from the paths to the files in the project while the project files are local copies of the originals on the server.
 - **Validate breakpoints with "file_exists()"** - select this check box to have PhpStorm check whether the file on server where the PhpStorm attempts to register a breakpoint is [correctly mapped](#) to an existing local file. If the file is missing or the mapping is incorrect, PhpStorm shows a warning and opens a dialog box for specifying correct mappings.

If the check box is cleared, the debugging session will be canceled as soon as the debugging tool reaches a breakpoint with a missing or incorrect mapping.
 - **Ignore external connections through unregistered server configurations** - select this check box to have PhpStorm ignore connections received from hosts and through ports that are not registered as [server configurations](#). When this check box is selected, PhpStorm does not attempt to create a server configuration automatically.

- XDebug** Use the controls in this area to configure debugging using the XDebug tool.
 - **Debug port:** in this text box, specify the port for PhpStorm and the XDebug engine to communicate through. This must be exactly the same port number as specified in the `php.ini` file:


```
xdebug.remote_port = <port_number>
```

By default, Xdebug listens to port 9000.
 - **Can accept external connections:** select this check box to enable PhpStorm to accept any incoming connections from XDebug engines through the port specified in the **Debug port** text box.
 - **Break at first line (for external connections):** select this check box to have XDebug stop as soon as connection between it and PhpStorm is established (instead of running automatically until the first breakpoint is reached).
 - **Force break at the first line when no path mapping is specified:**
 - **Force break at the first line when the script is outside the project:**

- Zend Debugger** Use the controls in this area to configure debugging using the Zend Debugger tool.
 - **Debug port** - in this text box, specify the port for PhpStorm and the Zend Debugger engine to communicate through.
 - **Can accept external connections** - select this check box to enable PhpStorm to accept any incoming connections from Zend Debugger engines through the port specified in the **Debug port** text box.
 - **Settings broadcasting port** - in this text box, specify the port through which the debugger settings are passed to the debugging toolbar in the browser.

Tip

If starting the Zend Debugger tool fails with the message "Port is busy", specify a port number of your choice higher than 10000.

Use debugger bookmarklets to initiate debugger from your favorite browser Follow this link to open the [Zend Debugger & XDebug bookmarklets](#) page where you can generate bookmarklets through which you will [start/stop a debugging session by controlling the debugger cookie](#).

See Also

Procedures:

- [Configuring a Debugging Engine](#)
- [PHP Debugging Session](#)
- [Running](#)
- [Debugging](#)
- [PHP-Specific Guidelines](#)
- [Creating and Editing Run/Debug Configurations](#)
- [Creating and Saving Temporary Run/Debug Configurations](#)

Reference:

- [PHP](#)

External Links:

- <http://blogs.jetbrains.com/webide/2011/03/configure-php-debugging-in-phpstorm-2-0/>

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Skipped Paths

Previous | [Next](#) | [See Also](#) | [Comments](#)

[File](#) | [Settings](#) | [PHP](#) | [Skipped Paths](#)

[PhpStorm](#) | [Preferences](#) | [PHP](#) | [Skipped Paths](#)

Note

The page is available only when the `PHP` bundled plugin is enabled. By default the plugin is active. If not, [enable](#) it in the [Plugin Configuration Wizard](#) or the [Plugins](#) page of the [Settings](#) dialog box.

On this page, specify the files and folders to be processed but excluded from debugging.

Item	Description
Notify about skipped paths	Select this check box to have PhpStorm inform you every time a file or folder is skipped during a debugging session.
Skipped paths	This list box shows the files and folders within the current project that will be just processed without debugging.
Add	Click this button to have a new line added to the list where you can specify the file or folder to skip. Tpe the path manually or click the Browse button  and choose the desired items in the dialog box that opens.
Remove	Click this button to have the selected item removed from the list.

See Also

Procedures:

- [Configuring a Debugging Engine](#)
- [Debugging](#)

Reference:

- [Debug](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

XDebug Proxy

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

[File](#) | [Settings](#) | [PHP](#) | [Debug](#)

[PhpStorm](#) | [Preferences](#) | [PHP](#) | [Debug](#)

Note

The page is available only when the `PHP` bundled plugin is enabled. By default the plugin is active. If not, [enable](#) it in the [Plugin Configuration Wizard](#) or the [Plugins](#) page of the [Settings](#) dialog box.

On this page, enable, disable, and re-configure your access to debugging PHP applications in the multiuser mode via an [Xdebug proxy server](#) .

Item	Description
IDE key	In this text box, specify the name for the Proxy server to identify connections from your IDE. This should be the value of the <code>xdebug.idekey</code> setting in your currently active <code>php.ini</code> configuration file.
Host	In this text box, specify the host on which the Xdebug proxy server resides.
Port	In this text box, specify the port which PhpStorm will listen to during a proxy debugging session.

See Also

Procedures:

- [PHP Debugging Session](#)
- [Multiuser Debugging Via XDebug Proxies](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Language Injections

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

[File](#) | [Settings](#) | [Language Injection](#) for Windows and Linux

[PhpStorm](#) | [Preferences](#) | [Language Injection](#) for Mac OS

Use this page to configure the behavior of the [IntelliLang plugin](#).

The page consists of one tab [Injection](#).

See Also

Concepts:

- [Using Language Injections](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Injection

Previous | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

[File](#) | [Settings](#) | [Language Injection](#) for Windows and Linux
[PhpStorm](#) | [Preferences](#) | [Language Injection](#) for Mac OS

Use this tab to manage the list of available language injections and configure the language injection feature for text, attributes, and parameters.

The tab consists of two areas:

- [Injection Entries](#)
- [Toolbar](#)

Injection entries

The list shows all injections currently available in PhpStorm.

To enable or disable an injection, select or clear the check box next to it.

You can also enable or disable a number of injections at once. To do that, select the required injections in the list and click **Enable Selected Injections**  or **Disable Selected Injections**  on the toolbar.

Item	Description
Display Name	This read-only field shows the name of the injection and the library that implements it.
Language	This read-only field shows the language of the injection.

Use the toolbar buttons to manage the contents of the list.

Toolbar

Item	Tooltip and Shortcut	Description
	Add InsertInsert	Click this button to create a new entry. From the pop-up list that opens select the new entry type: <ul style="list-style-type: none"> • XML Attribute Injection • XML Tag Injection Configure the entry in the Language Injection Settings dialog box that opens.
	Remove DeleteDelete	Click this button to have the selected entry removed from the list.
	Duplicate	Click this button to have PhpStorm create a copy of the selected injection entry. By re-configuring this copy you can create a new injection based on the existing one.
	Edit EnterEnter	Click this button to re-configure the selected entry in the Language Injection Settings dialog box that opens.
	Enable Selected Injections	Click this button to enable the injections that are currently selected in the list.
	Disable Selected Injections	Click this button to disable the injections that are currently selected in the list.
	Move to Project/ Make Global Shift+SpaceShift Space	Use this button to toggle between the project and global states of the selected injection configurations. <ul style="list-style-type: none"> • Character strings configured as project injections are treated as source code only within the current project. • Character strings configured as global injections are treated as source code at the PhpStorm level, that is, within any PhpStorm project.
	Export	Click this button to have one or several selected injection configurations saved in a file. Upon clicking this button, the Export Selected Injections to File dialog box opens. <ul style="list-style-type: none"> • To have the configurations appended to an existing file, select the required file. • To have the configurations saved in a new file, specify the file name and choose the file type from the drop-down list.
	Import	Click this button to have an injection configuration from another PhpStorm installation imported. The Select Path dialog box opens, where you can select the <code>IntelliLang.xml</code> file from the desired location. The dialog box that opens displays the entries contained in the configuration file. Remove the unnecessary entries from the list using the Delete button.
	Note	This will not affect the contents of the configuration file.
	Tip	This selective import-feature makes it easy to share certain configurations in a team without losing any local entries as it happens when settings are imported via the core Importing Settings feature.
	Warning	To prevent inconsistent data, import is only possible if the existing configuration has not been changed or has been saved using the Apply button.

See Also

Concepts:

- [Using Language Injections](#)

Reference:

- [Language Injections](#)
- [Language Injection Settings Dialog: Java Parameter](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Language Injection Settings Dialog: XML Tag

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

[File](#) | [Settings](#) | [Language Injection for Windows and Linux](#)
[PhpStorm](#) | [Preferences](#) | [Language Injection for Mac OS](#)

The dialog box opens when you click the **Add** button  or select an entry and click the **Edit** button  in the **Injection** tab and choose **XML Tag** on the context menu.

Item	Description
Display Name	<p>In this text box, type the name that will identify the injection configuration in the list.</p> <p>Note</p> <p>The text box is available only during configuration creation.</p>
Language	<p>In this area, specify the language of the injection. If necessary, define the context to be automatically made up when the injection is opened in the editor.</p> <ul style="list-style-type: none"> • ID - from this drop-down list, select the ID of the language of the injection. • Prefix - in this text box, specify the character string to be automatically added before the injection. • Suffix - in this text box, specify the character string to be automatically added after the injection. <p>Note</p> <p>The Prefix and Suffix text boxes are optional.</p>
XML Tag	<p>In this area, define the XML tag which indicates that the text enclosed in this tag should be treated as the selected language.</p> <ul style="list-style-type: none"> • Local Name - in this text box, specify the tag name without a namespace prefix. Use regular expressions to specify multiple tag names (<code>name1 name2</code>), case-insensitive names (<code>(?i) tagname</code> matches <code>tagname</code> as well as <code>TagName</code>), etc. <p>Warning</p> <p>Be sure not to enter any space characters as they would be significant for the match.</p> <ul style="list-style-type: none"> • Namespace - in this text box, specify the namespace URI of the XML tag. <p>Note</p> <p>The field is optional.</p>
Apply to all text fragments recursively	<p>Select this check box to have the injection applied to embedded fragments recursively.</p>
Advanced	<p>In this area, specify additional settings to enable more fine-grained control over the injection process.</p> <ul style="list-style-type: none"> • Value Pattern - in this text box, type a regular expression that determines the part of the XML text's value to inject the language into. By using the first capturing group of the pattern as the target for injection, you can configure the procedure to have the language injected only into values that match a certain pattern or into multiple parts that match the pattern. Examples: <code>[\$#]\{ (.*) \}</code> matches the pattern used by the JSP/JSF Expression Language. <code>^javascript:(.*)</code> matches the <code>javascript</code> protocol that can be used in hyperlink-hrefs to execute JavaScript code. • XPath Condition - in this text box, specify an XPath expression to address the injection-target more precisely. The context in which the expression is evaluated is the surrounding XML tag. <p>Note</p> <p>It is possible to use the XPath extension functions that are provided by the Jaxen XPath engine, e.g. <code>lower-case()</code>. Also, there are three additional functions that can be used to determine the current file's name, extension, and file type: <code>file-name()</code>, <code>file-ext()</code> and <code>file-type()</code>. Alternatively, a list of available functions can be retrieved through standard code completion.</p>

Warning

For performance reasons, it is recommended that you keep these expressions as simple as possible. Especially expressions that cause the whole document to be scanned, such as `//foo/bar` might cause performance problems with large files.

- **Single File** - select this check box to specify an exact file to apply the injection to.

See Also

Concepts:

- [Using Language Injections](#)

Reference:

- [Language Injections](#)
- [Injection](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Language Injection Settings Dialog: XML Attribute

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

[File](#) | [Settings](#) | [Language Injection for Windows and Linux PhpStorm](#) | [Preferences](#) | [Language Injection for Mac OS](#)

The dialog box opens when you click the **Add** button  or select an entry and click the **Edit** button  in the [Injection](#) tab and choose **XML Attribute Injection** on the context menu.

Item	Description
Display Name	<p>In this text box, type the name that will identify the injection configuration in the list.</p> <p>Note</p> <p>The text box is available only during configuration creation.</p>
Language	<p>In this area, specify the language of the injection. If necessary, define the context to be automatically made up when the injection is opened in the editor.</p> <ul style="list-style-type: none"> • ID - from this drop-down list, select the ID of the language of the injection. • Prefix - in this text box, specify the character string to be automatically added before the injection. • Suffix - in this text box, specify the character string to be automatically added after the injection. <p>Note</p> <p>The Prefix and Suffix text boxes are optional.</p>
XML Tag	<p>In this area, specify the XML tags in which the attributes are impacted by the defined configuration.</p> <ul style="list-style-type: none"> • Local Name - in this text box, specify the tag name without a namespace prefix. Use regular expressions to specify multiple tag names (<code>(name1 name2)</code>, case-insensitive names (<code>(?i) tagname</code> matches <code>tagname</code> as well as <code>TagName</code>), etc. <p>Warning</p> <p>Space characters are not allowed as they affect the match result.</p> <ul style="list-style-type: none"> • Namespace - in this text box, specify the namespace URI of the XML tag. <p>Note</p> <p>Both fields are optional. However, if the Local Name text box is empty the configuration will apply to any attribute that matches the configured name, regardless of its containing XML tag.</p>
XML Attribute	<p>In this area, define the XML tag attribute which indicates that the text enclosed in a tag with such attribute should be treated as the selected language.</p> <ul style="list-style-type: none"> • Local Name - in this text box, specify the attribute name without a namespace prefix. Use regular expressions: For example, to match HTML event handler attributes, type <code>on.*</code> in the text box. <p>Note</p> <p>The field is optional, unless the Local Name text box in the XML Tag area is empty. If the attribute local name is not specified, the configuration applies to all attributes of the enclosing tag.</p> <ul style="list-style-type: none"> • Namespace - in this text box, specify the namespace URI of the attribute.

- Advanced In this area, specify additional settings to enable more fine-grained control over the injection process.
- **Value Pattern** - in this text box, type a regular expression that determines the part of the attribute's value to inject the language into. By using the first capturing group of the pattern as the target for injection, you can configure the procedure to have the language injected only into values that match a certain pattern or into multiple parts that match the pattern.
 - **XPath Condition** - in this text box, specify an XPath expression to address the injection-target more precisely. The context in which the expression is evaluated is the attribute itself.
 - **Single File** - select this check box to specify an exact file to apply the injection to.

See Also

- Concepts:
- [Using Language Injections](#)
- Reference:
- [Language Injections](#)
 - [Injection](#)
- Web Resources:
- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>

Language Injection Settings: Generic JavaScript

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

[File](#) | [Settings](#) | [Language Injection for Windows and Linux](#)
[PhpStorm](#) | [Preferences](#) | [Language Injection for Mac OS](#)

The dialog box opens when you click the **Add** button  or select an entry and click the **Edit** button  in the [Injection](#) tab and choose **Generic JS** on the context menu.

Warning

PhpStorm comes with a set of predefined injection configurations which is quite sufficient to ensure high productivity and comfortable environment. Therefore it is strongly recommended that you use the predefined injection configurations and avoid creating new ones.

Item	Description
Display Name	In this text box, type the name that will identify the injection configuration in the list. Note The text box is available only during configuration creation.
Language	In this area, specify the language of the injection. If necessary, define the context to be automatically made up when the injection is opened in the editor . <ul style="list-style-type: none"> • ID - from this drop-down list, select the ID of the language of the injection. • Prefix - in this text box, specify the character string to be automatically added before the injection. • Suffix - in this text box, specify the character string to be automatically added after the injection. Note The Prefix and Suffix text boxes are optional.
Places Pattern	In this text box, type the rules that define the context where you want PhpStorm recognize literals as injections. Note The rules are built from Program Structure Interface Patterns and are actually chained calls of methods of a Groovy-like internal PhpStorm language. The Program Structure Interface  shows the structure of a file as %product\$ treats it. For more information on the syntax used, refer to Custom Plugin Development  guide. Warning These rules are PhpStorm internals, and it is strongly recommended that you use the predefined injection configurations and avoid creating new ones.
Advanced	In this area, specify additional settings to narrow the context where the injection is applicable and thus the enable more fine-grained control over the injection process. <ul style="list-style-type: none"> • Value Pattern - in this text box, type a regular expression that determines the context to inject the language into. By using the first capturing group of the pattern as the target for injection, you can configure the procedure to have the language injected only into values that match a certain pattern or into multiple parts that match the pattern. For example, <code>^javascript:(.*)</code> matches the <code>javascript</code> protocol that can be used in hyperlink-hrefs to execute JavaScript code. • Single File - select this check box to specify an exact file to apply the injection to. Type the name of the file in the Value Pattern.

See Also

Concepts:

- [Supported Languages](#)
- [Using Language Injections](#)

Reference:

- [Language Injections](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Language Injection Settings: Generic PHP

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

[File](#) | [Settings](#) | [Language Injection for Windows and Linux](#)
[PhpStorm](#) | [Preferences](#) | [Language Injection for Mac OS](#)

The dialog box opens when you click the **Add** button  or select an entry and click the **Edit** button  in the [Injection](#) tab and choose **Generic PHP** on the context menu.

Warning

PhpStorm comes with a set of predefined injection configurations which is quite sufficient to ensure high productivity and comfortable environment. Therefore it is strongly recommended that you use the predefined injection configurations and avoid creating new ones.

Item	Description
Display Name	In this text box, type the name that will identify the injection configuration in the list. Note The text box is available only during configuration creation.
Language	In this area, specify the language of the injection. If necessary, define the context to be automatically made up when the injection is opened in the editor . <ul style="list-style-type: none"> • ID - from this drop-down list, select the ID of the language of the injection. • Prefix - in this text box, specify the character string to be automatically added before the injection. • Suffix - in this text box, specify the character string to be automatically added after the injection. Note The Prefix and Suffix text boxes are optional.
Places Pattern	In this text box, type the rules that define the context where you want PhpStorm recognize literals as injections. Note The rules are built from Program Structure Interface Patterns and are actually chained calls of methods of a Groovy-like internal PhpStorm language. The Program Structure Interface  shows the structure of a file as %product\$ treats it. For more information on the syntax used, refer to Custom Plugin Development  guide. Warning These rules are PhpStorm internals, and it is strongly recommended that you use the predefined injection configurations and avoid creating new ones.
Advanced	In this area, specify additional settings to narrow the context where the injection is applicable and thus the enable more fine-grained control over the injection process. <ul style="list-style-type: none"> • Value Pattern - in this text box, type a regular expression that determines the context to inject the language into. By using the first capturing group of the pattern as the target for injection, you can configure the procedure to have the language injected only into values that match a certain pattern or into multiple parts that match the pattern. For example, <code>^javascript:(.*)</code> matches the <code>javascript</code> protocol that can be used in hyperlink-hrefs to execute JavaScript code. • Single File - select this check box to specify an exact file to apply the injection to. Type the name of the file in the Value Pattern.

See Also

Concepts:

- [Supported Languages](#)
- [Using Language Injections](#)

Reference:

- [Language Injections](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Schemas and DTDs

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: 

File | Settings | Schemas and DTDs

Ctrl+Alt+S Meta Comma



PhpStorm | Preferences | Schemas and DTDs

Ctrl+Alt+S Meta Comma



Use this dialog box to map URIs of external resources with local files and create a list of URIs that should not be highlighted as errors.

Item	Description
Configure External Resources	In this area, map URIs with files and folders.
URI	This read-only field shows the URI of the resource.
Location	This read-only field shows the full path to the resource.
Project	Check this option, if the selected mapping should be used within current project only. If the check box is cleared, this mapping is available in all projects.
Add	Click this button to open the External Resource dialog box, where you can create a new mapping between a URI and a file or a folder.
Edit	Click this button to open the External Resource dialog box, where you can edit the selected mapping.
Remove	Click this button to delete the selected URI from the list.
Configure Ignored Resources	In this area, create a list of resources that should be ignored. URIs from this list will not be highlighted as errors any more.
URI	This read-only field shows the URIs of resources to be ignored
Add	Click this button to open the External Resource dialog box, where you can specify the URI of an ignored resource .
Edit	Click this button to open the External Resource , where you can change URI the of the selected ignored resource.
Remove	Click this button to delete the selected URI from the list.

Note

The URI becomes available for mapping to a local file.

Default HTML Language Level Select the language to be used for editing the HTML/XHTML files where doctype is not specified explicitly. These settings affect code completion, syntax and error highlighting etc. If the option **Other doctype** is selected, you have to specify the desired doctype same way as it is done in HTML files.

See Also

Procedures:

- [Markup Languages and Style Sheets](#)

External Links:

- [Doc type declaration](#) 

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Spelling

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: 

File | Settings | Spelling

Ctrl+Alt+S Meta Comma



PhpStorm | Preferences | Spelling

Ctrl+Alt+S Meta Comma



Use this dialog box to create your own spelling dictionaries and thus expand the basic spelling support provided by PhpStorm by default.

Item	Description
Accepted words tab	Use this tab to configure the list of words, which should be skipped by the Typo inspection.
Add	Click this button to open the Add New Word dialog box, where you can type a new entry. CamelCase or snake_case words cannot be used. If you try to add a word that is already included in one of the spelling dictionaries, this attempt will be denied.
Delete	Click this button to remove selected word from the list. Such word will be marked by the inspection as a typo.
Dictionaries tab	Use this tab to configure the dictionaries to be used for spellchecking.

User dictionaries path	This area displays the list of directories that contain user-defined dictionary files (textual files with the <code>.dic</code> extension, containing words delimited with newline).
Add	Click this button to open the Select Path dialog box and point to the folder with the dictionaries you want to use for spellchecking. Fully qualified path is added to the User dictionaries path list , and all <code>*.dic</code> files encountered thereunder are added to the Dictionaries list .
Remove	Click this button to remove selected path from the list. So doing, the words included in dictionaries under this path will be recognized as typos.
Dictionaries	This area shows all dictionaries that come bundled with PhpStorm, and those detected under the specified dictionary paths. By default, all dictionaries are enabled. To disable a dictionary, clear its check box.

See Also

Procedures:

- [Spellchecking](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

SQL Dialects

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

File | Settings | SQL Dialects

Ctrl+Alt+S Meta Comma



PhpStorm | Preferences | SQL Dialects

Ctrl+Alt+S Meta Comma



Use this dialog to specify which SQL dialects are used in the project.

The dialect may be set at the project level, and also at the level of a directory or a file.

If a dialect is not specified explicitly, it is inherited from a higher hierarchical level.

The dialects specified explicitly are shown in black. The inherited values are shown in gray italic.

Note

For this dialog to be available, the SQL Support [plugin must be enabled](#).

Item	Description
File/Directory	This column shows the hierarchy of files and folders in the project. Each row corresponds to a directory or a file.
SQL Dialect	Specify the SQL dialect to be used. Click the cell of interest and select the necessary option from the list. In addition to particular dialects, you can select: <ul style="list-style-type: none"> • Clear. This will clear the corresponding cell. As a result, an option from a higher hierarchical level will be inherited. • keywords only. This means that no particular dialect is specified. As a result, the coding assistance will be limited only to highlighting the SQL keywords in the corresponding file or files.

See Also

Concepts:

- [Supported Languages](#)

Language and Framework-Specific Guidelines:

- [Data Sources](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Template Data Languages

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

File | Settings | Template Data Languages for Windows and Linux

PhpStorm | Preferences | Template Data Languages for Mac OS

Use this page to define template data languages to be used for the entire project, its directories and individual files.

Item	Description
------	-------------

File/Directory	This column displays the project tree view.
Template data language	This column displays template languages for a file or directory. Click a line that corresponds to the desired file or directory in this column and select template language for PhpStorm to use when parsing template framework files from the drop-down list.

See Also

Procedures:

- [Specifying Template Data Languages for Templates](#)

Reference:

- [File Types](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Tasks

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

File | Settings | Tasks

Ctrl+Alt+SMeta Comma



PhpStorm | Preferences | Tasks

Ctrl+Alt+SMeta Comma



Use this page to set up the general options for [tasks and context management](#).

Item	Description
Task history length	Specify the number of tasks to be stored in history.
Enable issue cache	<ul style="list-style-type: none"> • Select this check box to have PhpStorm synchronize with the issue tracking system in the background on a regular basis, No matter whether you actually request on information from your issue tracker or not, PhpStorm will connect to your issue tracking system according to the specified frequency and refresh the cached issues. The advantage of this approach is that when you need to switch to a task, the up-to-date information is already at your disposal so you do not need to wait till PhpStorm establishes connection with the tracker and retrieves the information. <p>Tip</p> <p>This configuration is especially recommended when working with rather "slow" issue tracking systems.</p> <ul style="list-style-type: none"> • Clear this check box to have PhpStorm PhpStorm connect to the issue tracking system only when you actually need information on issues, clear the Enable issue cache check box.
Update (N) issues each (M) minutes	Specify how often PhpStorm should synchronize with your issue tracking system and refresh the cached issues.

See Also

Procedures:

- [Enabling Integration with an Issue Tracking System](#)
- [Managing Tasks and Context](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Version Control

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

File | Settings | Version Control

Ctrl+Alt+SMeta Comma



PhpStorm | Preferences | Version Control

Ctrl+Alt+SMeta Comma



In the dialog boxes below this node, specify the [common settings](#) for version control activities that are applied to the project files regardless of the version control system used.

The following dialog boxes are grouped below this node:

- [Confirmation](#)
- [Background](#)

- [Ignored Files](#)
- [Issue Navigation](#)
- [Changelist Conflicts](#)
- [VCSs](#)

In the **Version Control** dialog box, specify which version control systems will be used for specific directories or the entire project.

Item	Description
Directory	This read-only field shows the path to a directory or the project root.
VCS	Use this drop-down list to select a version control system to put the specified directory under.
	Note The drop-down list displays only the version control systems that are installed and enabled in the PhpStorm plugin repository.
	Click this button to open the Version Control Configurations dialog box and update the configuration settings of the selected VCS.
Add	Click this button to add a directory mapping to the list. The Add VCS Directory Mapping dialog box opens where you can specify the required directory, select a VCS for it, and open the Version Control Configurations dialog box to configure the specified VCS, if necessary.
Edit	Click this button to edit the selected directory mapping. The Edit VCS Directory Mapping dialog box opens where you can update the selected mapping and configure the specified VCS, if necessary.
Remove	Click this button to remove the selected directory mapping from the list.
Store on shelf base revision texts of files under DVCS	Select this check box to have PhpStorm always shelve the base revisions of files that are under Git or Mercurial version control. By default, PhpStorm always "remembers" the last commit hash [↗] . However, this information is not sufficient if the history has been changed since the last commit as a result of running the Rebase operation. In this case, having a copy of the base revision may help.
	Note This check box is relevant only for integration with distributed version control systems, such as, Git and Mercurial .
Show changed in last <number> days	Select this check box to have color indication of file status applied during stacktrace analysis and debugging . The names of the files that have been changed within a certain period will be highlighted accordingly. Specify the period to observe in the spin box.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Version Control with PhpStorm](#)
- [Configuring Version Control Options](#)

Reference:

- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> [↗]
- <http://youtrack.jetbrains.net/issues/WI> [↗]

Confirmation

Previous | [Next](#) | [See Also](#) | [Comments](#)

[File](#) | [Settings](#) | [Version Control](#) | **Confirmation**

[PhpStorm](#) | [Preferences](#) | [Version Control](#) | **Confirmation**

In this dialog box, specify whether you want PhpStorm to ask you for confirmation before performing specific version control related actions.

Item	Description
When files are created	In this section, specify whether and how to put a file created in PhpStorm under version control. The following options are available: <ul style="list-style-type: none"> • Show options before adding to version control: When this option is selected, newly created files are put under version control after you specify the version control options in the dialog box that opens. • Add silently: When this option is selected, newly created files are automatically put under version control without displaying any messages. • Do not add: When this option is selected, newly created files remain unversioned and you can put them under version control later.
When files are deleted	In this section, specify whether and how to remove a file from the version control system when the file is removed in PhpStorm. The following options are available: <ul style="list-style-type: none"> • Show options before removing from version control: When this option is selected, locally removed files are also removed from the specified VCS but first a dialog box for selecting version control options is displayed. • Remove silently: When this option is selected, all locally removed files are removed from the specified VCS without asking for confirmation. • Do not remove: When this option is selected, locally removed files remain under version control.
When empty changelist	In this section,

becomes inactive

Display Option dialogs when these commands are invoked In this section, specify whether you want PhpStorm to ask you for confirmation in association with specific commands. To enable showing a prompt before performing an action, select the relevant check box. The following options are available:

- Add
- Checkout
- Update
- Status
- Edit
- Undo Checkout
- Get Latest version

Note

Check boxes are enabled or disabled depending on the assigned version control system.

Show Clear Read-Only Status dialog box

Select this check box to have PhpStorm explicitly require cancellation of the read-only status when you open a file in the editor and try to modify it.

The read-only status can be cleared in two ways:

- Using the current VCS: the file is added to a [changelist](#).
- Using a file system: the file is not added to the changelist.

Note

This option is relevant for those version control systems that separate check-out from opening for editing.

Suggest to move uncommitted changes to another changelist

When this option is selected, on committing a changelist to the repository with some files excluded from the commit, PhpStorm prompts to move these files to another changelist.

Force non-empty check-in comments

Select this check box to suppress committing changes without supplying them with corresponding comments.

Create changelist on failed commit

Use this drop-down list to define the PhpStorm behavior in case of failed commits. The available options are:

- **Yes:** if this option is selected, the `Failed commit` changelist will be automatically created, and the respective files will be moved to this changelist.
- **No:** if this option is selected, the files that failed to commit will remain in their original changelist.
- **Ask:** if this option is selected, on failed commit PhpStorm will display a dialog box asking whether the files should be moved to another changelist, or remain in the original one.

See the section [Resolving Conflicts](#).

See Also

Concepts:

- [Version Control with PhpStorm](#)

Reference:

- [Version Control Reference](#)

Version Control:

- [Version Control with PhpStorm](#)
- [Resolving Conflicts](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Background

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

[File](#) | [Settings](#) | [Version Control](#) | [Background](#)

[PhpStorm](#) | [Preferences](#) | [Version Control](#) | [Background](#)

In this dialog box, enable performing specific version control related operations in the background without any activities from your side.

Item	Description
Background Operations	<p>In this area, specify the version control related operations to be performed in the background. To enable running an operation in the background, select the relevant check box:</p> <ul style="list-style-type: none"> • Perform update from VCS in background • Perform commit to VCS in background • Perform checkout from VCS in background • Perform Edit/Checkout in background • Perform Add/Remove in background • Perform revert in background

- Enable background processes Select this check box to have PhpStorm refresh cached changelists and detect "changed on server" conflicts in the background.
- Changelists to cache initially Use this spin box to specify the number of changelists to be stored in the cache.
- Refresh changes every ... minutes Use this spin box to specify how often the VCS should check for new changes and refresh the cache.

Note

The spin box is only enabled when the **Enable background processes** check box is selected.

"Changed on server" conflicts

In this area, specify whether you want PhpStorm to check whether a checked out files have been updated by someone else. To enable background synchronization with the server, select the **Check every ... minutes**. In the spin box, specify how often you want synchronization to take place.

Note

This area is only enabled when the **Enable background processes** check box is selected.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Version Control with PhpStorm](#)

Reference:

- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Ignored Files

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

[File](#) | [Settings](#) | [Version Control](#) | [Ignored Files](#)

[PhpStorm](#) | [Preferences](#) | [Version Control](#) | [Ignored Files](#)

Use this dialog box to create a list of files and directories that you do not want to put under version control. These can be file names associated with VCS administration, backup files, and any other artifacts that you want to remain unversioned. You can also specify patterns that define files or directories to ignore.

Item	Description
Add	Click this button to add an item to the list. The Ignore Unversioned Files dialog box opens where you can type an exact path to a file or directory to be ignored or specify a pattern that defines the names of files and directories to be ignored.
Edit	Click this button to edit the selected path or pattern in the Ignore Unversioned Files dialog box.
Remove	Click this button to remove the selected path or pattern from the list.

See Also

Procedures:

- [Configuring Ignored Files](#)

Reference:

- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Ignore Unversioned Files

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

[File](#) | [Settings](#) | [Version Control](#) | [Ignored Files](#)

[PhpStorm](#) | [Preferences](#) | [Version Control](#) | [Ignored Files](#)

The dialog box opens when you select a row in the **Ignored Files** dialog box and click the **Add** or **Edit** button.

Use this dialog to configure rules that define which files should be ignored by version control systems.

Select the relevant option and fill in the text box next to it.

Item	Description
------	-------------

Ignore specified file In this text box, specify the fully qualified name of the file to be ignored. You can type the file name manually or click the  button and select the desired file in the **Select File to Ignore** dialog box.

Tip
Basic code completion is available.

Ignore all files under In this text box, specify the fully qualified name of the directory to be ignored. You can type the directory name manually or click the  button and select the desired file in the **Select Directory to Ignore** dialog box.

Tip
Basic code completion is available.

Ignore all files matching In this text box, specify a pattern that defines the names of files to be ignored.

Tip
Wildcards are welcome.

See Also

Procedures:

- [Configuring Ignored Files](#)

Reference:

- [Version Control Reference](#)
- [Configure Ignored Files Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Issue Navigation

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

[File](#) | [Settings](#) | [Version Control](#) | [Issue Navigation](#)

[PhpStorm](#) | [Preferences](#) | [Version Control](#) | [Issue Navigation](#)

Use this dialog box to create a list of issue patterns and navigation links to a bug tracking system. PhpStorm will search for the specified patterns in check-in comments and use the corresponding links to navigate to the corresponding issues in the bug tracking system.

Item	Description
Issue	This read-only field shows an issue pattern.
Link	This read-only field shows the link to navigate from the issue pattern in the current row to the issue in the bug tracking system.
Add	Click this button to create a new issue navigation pattern and link. The Add Issue Navigation Link dialog box opens where you can specify: <ul style="list-style-type: none"> • A regular expression to define the issue ID. • A regular expression to define the navigation link to the issue. <p>Tip Please, refer to Regular Expressions Syntax Reference for details on using special characters in regular expressions.</p>
Add JIRA pattern	Click this button to create a new JIRA pattern. The Create JIRA Issue Navigation Pattern dialog box is opened where you can specify the URL to your JIRA installation. The regular expression that defines the pattern is added automatically.
Add YouTrack pattern	Click this button to create a new pattern for our bug tracking system YouTrack . In the dialog box that is opened, specify the URL to your YouTrack installation. The regular expression that defines the pattern is added automatically.
Edit	Click this button to update the selected issue navigation link.
Delete	Click this button to remove the selected issue navigation link from the list.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Version Control with PhpStorm](#)

Reference:

- [Version Control Reference](#)
- [Regular Expression Syntax Reference](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Changelist Conflicts

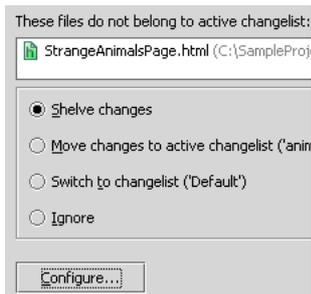
[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

[File](#) | [Settings](#) | [Version Control](#) | [Changelist Conflicts](#)

[PhpStorm](#) | [Preferences](#) | [Version Control](#) | [Changelist Conflicts](#)

Use this page to configure protection of files belonging to inactive changelists, from occasional conflicts.

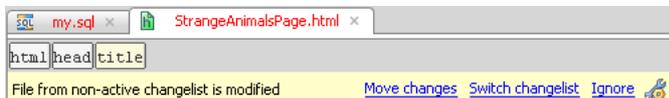
Item	Description
Enable changelist conflict tracking	If this check box is selected, PhpStorm makes it possible to protect files in inactive changelists. Such protection can be performed on the various levels enabled by the options listed below.
Show conflict resolving dialog	If this check box is selected, PhpStorm shows Resolve Changelist Conflict dialog box on an attempt to modify a file.



You must define the way to resolve conflict, before being allowed to proceed with changes. The possible ways to resolve a conflict are:

- [Shelve](#) changes.
- [Move](#) file to the active changelist.
- [Switch](#) changelists to make the current changelist active.
- [Ignore](#) conflict. In this case, the file in question will be added to the list of files with ignored conflicts.

Highlight files with conflicts	If this check box is selected, PhpStorm shows a yellow stripe on top of a file from an inactive changelist, when such file has been modified.
--------------------------------	---



In this stripe, you can choose one of the possible ways to resolve conflict:

- [Move](#) file to the active changelist.
- [Switch](#) changelists.
- [Ignore](#) conflict.

Names of the files belonging to inactive changelists are shown in red font in the editor tabs and in the Project view.

Highlight files from non-active changelists	If this check box is selected, names of the files belonging to inactive changelists are shown in light-blue font in the editor tabs and in the Project view.
---	--

Files with ignored conflicts	This area shows the list of files, for which Ignore option has been selected in the Resolve Changelist Conflict dialog box, or in the editor stripe.
------------------------------	---

Clear	Click this button to remove all files from the list of files with ignored conflicts.
-------	--

See Also

Concepts:

- [Changelist](#)

Procedures:

- [Shelving Changes](#)
- [Moving Items Between Changelists in the Changes Tool Window](#)
- [Assigning an Active Changelist](#)

Reference:

- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

VCSs

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

[File](#) | [Settings](#) | [Version Control](#) | [<VCS name>](#)

[PhpStorm](#) | [Preferences](#) | [Version Control](#) | [<VCS name>](#)

In these pages, specify the settings that will apply to the version control system which is used in your project. The available pages are:

- [Git](#)
- [Mercurial](#)
- [Perforce](#)
- [Subversion](#)
- [CVS](#)
- [TFS](#)

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Version Control with PhpStorm](#)

Reference:

- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Git

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

[File](#) | [Settings](#) | [Version Control](#) | [Git](#)

[PhpStorm](#) | [Preferences](#) | [Version Control](#) | [Git](#)

Use this page to specify the version control settings to be applied to the directories of your project that are under [Git](#) control.

Item	Description
Path to Git executable	In this text box, specify the path to the Git executable file. Type the path manually or click the Browse button  to open the Select Path - Git Configuration dialog box and select the location of the Git executable file in the directories tree.
Test	Click this button to check whether the specified settings ensure establishing connection.
SSH executable	Use this drop-down list to specify the SSH version to be used with Git. The available options are: <ul style="list-style-type: none"> • PhpStorm ssh - select this option to have the implementation provided by PhpStorm used. • Native - select this option to have the native implementation used. <p>Warning</p> <p>On some platforms, using the native ssh implementation may cause hang-up problems. Besides, you may need to configure a platform-specific <code>ssh-askpass</code> to receive prompts for passwords.</p>
File Conversion	Use this area, configure processing of line separators in source files after commit. The available options are: <ul style="list-style-type: none"> • Do not convert - select this option to leave sources as is. • Convert to project line separators - select this option to have line separators converted according to the project code style settings. • Ask before conversion - if this option is selected, the list of files to be converted is shown before conversion is applied and you are asked to choose the conversion method.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Version Control with PhpStorm](#)
- [Using Git Integration](#)

Reference:

- [Version Control Reference](#)
- [Git Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Mercurial

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

[File](#) | [Settings](#) | [Version Control](#) | [Mercurial](#)

[PhpStorm](#) | [Preferences](#) | [Version Control](#) | [Mercurial](#)

Use this page to specify the version control settings to be applied to the directories of your project that are under [Mercurial](#) control.

Item	Description
Changesets	<p>In this area, specify whether you want PhpStorm to detect incoming and outgoing changes in the background mode.</p> <ul style="list-style-type: none"> • Check for incoming changesets - select this check box to have PhpStorm check for incoming changesets every 5 minutes. • Check for outgoing changesets - select this check box to have PhpStorm check for outgoing changesets every 5 minutes.
Path to hg executable	<p>In this area define how PhpStorm should find the Mercurial executable file.</p> <ul style="list-style-type: none"> • Autodetect hg in PATH - when this option is selected, PhpStorm tries to find and use the path to the Mercurial executable file specified at the system level in the <code>Path</code> system environment variable. • Specify executable path - select this option to specify the location of the Mercurial executable file explicitly. If necessary, click the Browse button  and select the directory in the dialog box that opens.
Run hg as 'bash -c <path to hg>	<p>Select this check box to start <code>bash</code> script and have Mercurial-related actions converted into commands of format <code>bash -c hg <command></code>.</p> <p>This option is helpful if, due to non-standard installation of Mercurial and Python, running Mercurial requires definitions of some environment variables and specific configuration settings that are defined in the <code>bash</code> configuration file <code>.bashrc</code> or <code>.bash_profile</code>. If you run <code>bash</code> and then use the <code>bash</code> console, these variables and settings are read from the configuration file, so Mercurial starts successfully and its commands are executed in the <code>bash</code> console. However, PhpStorm itself does not read the <code>bash</code> configuration file with required environment variables and settings during start-up, so attempts to run Mercurial from PhpStorm without <code>bash</code> already running fail with the error message "Hg executable invalid" or similar.</p> <p>When this check box is selected, PhpStorm first starts <code>bash</code>, reads the required environment variables and configuration settings, and runs Mercurial successfully.</p> <p>Note</p> <ol style="list-style-type: none"> 1. Alternatively, you can define the required environment variables and configuration settings in another file and make this file accessible for PhpStorm. 2. The check box is not available in Windows.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Version Control with PhpStorm](#)
- [Using Mercurial Integration](#)

Reference:

- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Perforce

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

[File](#) | [Settings](#) | [Version Control](#) | [Perforce](#)

[PhpStorm](#) | [Preferences](#) | [Version Control](#) | [Perforce](#)

Use this page to specify the version control settings to be applied to those directories of your project that are under Perforce control

Item	Description
Perforce is online	Select this check box to work with Perforce in the online mode.
Switch to offline mode automatically if Perforce is unavailable	Select this check box to ave PhpStorm automatically go offline as soon as Perforce becomes unavailable and display the corresponding message.
Config Settings	<p>In this area, specify which credentials you want to use for connecting to the Perforce server. The available options are:</p> <ul style="list-style-type: none"> • Use P4CONFIG or default connection - choose this option to connect to the Perforce server using the <code>P4CONFIG</code> or the default Perforce connection. <p>Note</p> <ol style="list-style-type: none"> 1. When this option is selected, the <code>Port</code>, <code>Client</code>, <code>User</code>, and <code>Password</code> text boxes below are disabled.

2. Using P4CONFIG makes it trivial to switch between Perforce settings for different projects, when necessary.

- **Use connection parameters** - choose this option to specify the connection credentials manually in the text boxes below:
 1. **Port** - in this text box, type the port which your Perforce client will listen to.
 2. **Client** - in this text box, type the name of your Perforce client.
 3. **User** - in this text box, type your user name to authenticate to the server.
 4. **Password** - in this text box, type your password to authenticate to the server.
- **Charset** - from this drop-down list, select the character set to applied when typing connection credentials.

Dump Perforce Commands to:	Select this check box to have PhpStorm create a file <code>P4.output</code> and store the output of Perforce commands in it.
Use login authentication	When this check box is selected, Perforce requires a login to authenticate a user.
Try to login silently	Select this check box to enable logging in without authentication.
Test Connection	Click this button to check whether the specified settings ensure establishing connection to the Perforce server.
Path to P4 executable	In this text box, specify the path to the Perforce Command Line Client's executable file <code>P4</code> . Click the Browse button  to open the Select Path - P4 Configuration dialog box and select the executable file in the directories tree.
Path to P4V executable	In this text box, specify the path to the Perforce Visual Client's executable file <code>P4V</code> . Click the Browse button  to open the Select Path - P4 Configuration dialog box and select the executable file in the directories tree.
Show branching history ...	Select this check box to enable displaying the branch history of a specified file, including all file branch points, edits, and merges.
Show integrated changelists in committed changes	Select this check box to have PhpStorm point at committed changes that are also integrated to other changelists and provide information on the target changelists that received the content in question.
Server timeout	In this text box, specify the time period in seconds after when the Perforce client cancels its attempts to establish connection to the server.
Enable Perforce Jobs Support	When this check box is selected, user interface for attaching and detaching Perforce jobs to change lists is provided in the Changes tool window and in the Commit Changes dialog box.

See Also

- Concepts:
- [Version Control with PhpStorm](#)

- Procedures:
- [Using Perforce Integration](#)

- Reference:
- [Version Control Reference](#)
 - [Changes Tool Window](#)
 - [Commit Changes Dialog](#)

- Web Resources:
- <http://www.jetbrains.net/devnet/community/wi> 
 - <http://youtrack.jetbrains.net/issues/WI> 

Subversion

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

[File](#) | [Settings](#) | [Version Control](#) | [Subversion](#)

[PhpStorm](#) | [Preferences](#) | [Version Control](#) | [Subversion](#)

Use this page to specify the version control settings to be applied to the directories of your project that are under Subversion control

Item	Description
Update administrative information only in changed subtrees	When this check box is selected, only those parts of a working copy are locked during update that are about to be affected.
Use system default Subversion configuration directory	Select this check box to store Subversion configuration files in the system default directory <code>user_home\Application Data\Subversion</code> .
Subversion configuration directory	In this text box, specify a Subversion configuration directory different from the default one. Click the Browse button  to open the Select Configuration Directory dialog box.
	<p>Tip</p> <p>The text box is available only when the Use system default Subversion configuration directory check box is not selected.</p>
Clear Auth Cache	Click this button to remove all stored credentials for <code>http</code> , <code>svn</code> , and <code>svn+ssh</code> protocols from the authentication cache.
Use PhpStorm general proxy settings as default for Subversion	Select this check box to have Subversion use the default proxy settings .
Edit Network Options	Click this button to open the Edit Subversion Options Related to Network Layers dialog box where you can update the network settings to be used in Subversion integration. The network settings are stored in the <code>servers</code> Subversion runtime configuration file.
Detect nested working copies	When this check box is selected, PhpStorm automatically detects and checks out the external working copies as soon as you check out the original working copy, provided that the nested working copy structure  has been defined.

- Check svn:mergeinfo in target subtree when preparing for merge Select this check box to have PhpStorm investigate the merge tracking information for the target branch before merging to prevent duplicates.
- Maximum number of revisions to look back in annotations Select this check box to limit the number of revisions to look back when calculating annotations. In the text box, specify the maximum number of revisions.
- Show merge source in annotations Select this check boxes to have merge sources included in annotations.
- Ignore whitespace differences in annotations Select this check box to have white spaces ignored during annotating and thus get more meaningful annotations and cast out senseless ones.

See Also

- Concepts:
- [Version Control with PhpStorm](#)
- Procedures:
- [Using Subversion Integration](#)
- Reference:
- [Version Control Reference](#)

- Web Resources:
- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>

Edit Subversion Options Related to Network Layers Dialog

Previous | Next | [See Also](#) | [Comments](#) | Shortcuts: Mac

[File](#) | [Settings](#) | [Version Control](#) | [Subversion](#)

[PhpStorm](#) | [Preferences](#) | [Version Control](#) | [Subversion](#)

The dialog box opens when you click the **Edit Network Options** button on the [Subversion](#) page of the [Settings](#) dialog box. In this dialog box, specify the Subversion network settings stored in the [servers](#) Subversion runtime configuration file.

The dialog box contains two tabs:

- System file** - this tab displays the default network configuration settings specified by the system administrator.
- User file** - in this tab, customize the default network configuration settings.

The dialog box consists of two panes:

- On the left-hand pane, add, edit, and remove configuration profiles. Network configuration settings are arranged into profiles of two types:
 - Group** - settings from such profile apply to a specific group, defined by a glob pattern.
 - Global** - settings from this profile are applied to all servers that do not match any glob pattern.
- On the right-hand pane, specify the settings for the selected configuration profile.

Toolbar options

Item	Tooltip and shortcut	Description
	Add InsertInsert	Click this button to have a new configuration profile added to the list.
	Delete DeleteDelete	Click this button to remove the selected profile from the list.
	Copy Ctrl+C Command C or Ctrl+Insert Command Insert	Click this button to have a copy of the selected profile created.

HTTP proxy settings

Item	Description
URL Patterns	In this text box, type the patterns that define the URL addresses of repositories to be accessed via proxy. Use commas to separate patterns.
Exceptions	In this text box, type the patterns that define the URL addresses of repositories to be accessed directly, without using proxy. Use commas to separate patterns.
Server	In this text box, specify the name or IP address of the proxy server to use.
Port	In this text box, specify the port number the proxy server listens to.
Connection timeout	In this text box, specify the time period in seconds after when the Subversion client cancels its attempts to establish connection to the server.
User	In this text box, type the user name or login to authenticate at the specified proxy server.
Password	In this text box, type the password that matches the specified login or user name.

SSL settings

Item	Description
Comma separated paths to CAs certificate files	In this text box, specify the paths to files that contain certificates of the Certificate Authority (CAs) files that are accepted by the Subversion client when accessing the repository.

SSL client certificate file	In this text box, specify the location of the SSL client certificate file. Type the path to the file manually or click the Browse button  and choose the location in the Select PATH dialog box that opens.
SSL client certificate passphrase	In this text box, type the SSL client certificate passphrase to use.
Trust default CAs	Select this check box to have the Subversion integration trust the set of default Certificate Authority files shipped with OpenSSL .

Repositories

Item	Description
Repositories	This list displays the URL addresses of the previously accessed repositories.
Test connection	Click this button to make sure that connection to the selected repository can be established successfully according to the settings specified in the dialog box.
	Tip
	When you click this button, PhpStorm displays the Authentication Required dialog box.

See Also

- Concepts:
- [Version Control with PhpStorm](#)

- Procedures:
- [Using Subversion Integration](#)
 - [Configuring HTTP Proxy](#)

- Reference:
- [Version Control Reference](#)
 - [Subversion](#)

- Web Resources:
- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>

CVS

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

[File](#) | [Settings](#) | [Version Control](#) | [CVS](#)

[PhpStorm](#) | [Preferences](#) | [Version Control](#) | [CVS](#)

Use this page to specify the version control settings to be applied to the directories of your project that are under CVS control.

Item	Description
Updating	In this area, define the PhpStorm behavior when merge conflicts occur during update from the server. <ul style="list-style-type: none"> Show dialog - select this option to have PhpStorm display a dialog box where you can examine, analyze, and resolve possible conflicts before updating. Skip merging for all project or module files merged with conflicts - select this option to suppress updating files where conflicts occurred during merge. Get latest repository versions silently - select this option to have your local files in question automatically updated with their latest repository versions.
Use read-only flag for not edited file	Select this check box to have the read-only status assigned to a file automatically after the check out, update, or commit operations.
Show CVS server output	Select this check box to have the server output of CVS commands displayed in the CVS Output tool window.
Default keyword substitution for text files	Use this drop-down list to specify the keyword expansion mode. CVS uses the keyword substitution mode of a file to differentiate binary files from ASCII files and to indicate what type of keyword substitution is applied when files are committed and checked out. The available options are: <ul style="list-style-type: none"> keyword&value (-kkv) keyword, value&locker (-kkv1) keyword only (-kk) original string (-ko) binary (-kb) value only (-kv)
Global Settings	Click this button to open the Global CVS Settings dialog box.

See Also

- Concepts:
- [Supported Version Control Systems](#)

- Procedures:
- [Using CVS Integration](#)

Reference:

- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

TFS

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

[File](#) | [Settings](#) | [Version Control](#) | [TFS](#)

[PhpStorm](#) | [Preferences](#) | [Version Control](#) | [TFS](#)

Use this page to specify the version control settings to be applied to those directories of your project that are under TFS control.

Item	Description
Servers and workspaces	<p>In this area, configure access to workspaces and servers to use.</p> <ul style="list-style-type: none"> • Manage: click this button to open the Manage TFS Servers and Workspaces dialog box where you can create a list of servers and workspaces you need to have access to. • Use PhpStorm HTTP Proxy settings for TFS: when this check box is selected, TFS servers are accessed through the Proxy server using the PhpStorm default Proxy settings.
Passwords	<p>In this area, handle the passwords for accessing TFS servers and workspaces.</p> <ul style="list-style-type: none"> • Reset stored passwords: click this button to discard the stored passwords.
Check-in policies compatibility	<p>A check-in policy is a rule that is executed before every check-in to ensure that the selected changeset is OK to commit. Standard policies are stored on the server and are executed on the client machines.</p> <p>Custom policies are implemented as custom plugins to PhpStorm. The IDs of these plugins are stored on the server, while the policies themselves are applied locally. Therefore, to enable the use of a policy in a team, all the team members should install the corresponding plugin.</p> <p>Use the controls in this area to configure how PhpStorm should treat third party check-in policies. These settings are applied to all PhpStorm projects by default unless they are overridden for a specific project.</p> <ul style="list-style-type: none"> • Evaluate Team Explorer policies: select this check box to have the <code>Microsoft Team Explorer</code> policy definitions installed and executed on the client machine. • Evaluate Teamprise policies: select this check box to have the <code>Teamprise</code> policy definitions installed and executed on the client machine. • Warn about not installed policies: select this check box to have warnings displayed in case the specified policy definition is not installed.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Using TFS Integration](#)

Reference:

- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Manage TFS Servers and Workspaces

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

[File](#) | [Settings](#) | [Version Control](#) | [TFS](#)

[PhpStorm](#) | [Preferences](#) | [Version Control](#) | [TFS](#)

The dialog box opens when you click the **Manage servers and workspaces** button in the [TFS](#) dialog box. Use this dialog box to handle the list of TFS servers and [workspaces](#) you have access to.

Item	Description
Server/Workspace	This read-only field shows the URL addresses of TFS servers you have access to and workspaces available on these TFS servers.
Workspace comment	This read-only field shows the descriptions of workspaces on the servers you have access to.
Team Servers	<p>Use the buttons in this area to manage the list of available servers and workspaces and configure access to them.</p> <ul style="list-style-type: none"> • Add: click this button to open the Add Team Foundation Server dialog box where you can specify the parameters for establishing connection to a TFS server. • Remove: click this button to remove the selected server from the list. • Reload workspaces: click this button to have the list of available workspaces refreshed. • TFS Proxy: click this button to open the Set TFS proxy for server... dialog box where you can specify the parameters for accessing the selected server via Proxy. • Check-in Policies: click this button to open the Edit Check-in Policies dialog box where you can manage the list of check-in policies to be applied.

Workspaces Use the buttons in this area manage the list of available workspaces and update the workspaces, when applicable.

- **Create:** click this button to open the [Create Workspace](#) dialog box for creating a new workspace.
- **Edit:** click this button to open the [Edit Workspace](#) dialog box for editing the selected workspace.
- **Delete:** click this button to remove the selected workspace from the list.

Close Click this button to save the settings, close the dialog box, and return to the [TFS](#) dialog box.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Version Control with PhpStorm](#)

Reference:

- [Version Control Reference](#)
- [TFS](#)
- [Create Workspace](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Add Team Foundation Server

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

[File](#) | [Settings](#) | [Version Control](#) | [TFS](#)

[PhpStorm](#) | [Preferences](#) | [Version Control](#) | [TFS](#)

The dialog box opens when you click the **Add** button in the [Manage TFS Servers and Workspaces](#) dialog box. Use this dialog box to specify and edit the parameters for establishing connection to TFS servers.

Item	Description
Server	In this text box, type the URL address of the TFS server you want to connect to.
Use system credentials	<ul style="list-style-type: none"> • Select this check box to use your credentials for logging on your operating system when authenticating to TFS. • Clear this check box to define special credentials for authenticating to the TFS server.
User name	In this text box, type your TFS user name.
Domain	In this text box, type the name of the network domain where the TFS server is located.
Password	In this text box, type your TFS password.
Store password	Select this check box to have PhpStorm remember the specified password.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Version Control with PhpStorm](#)

Reference:

- [TFS](#)
- [Manage TFS Servers and Workspaces](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Create Workspace

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

[File](#) | [Settings](#) | [Version Control](#) | [TFS](#)

[PhpStorm](#) | [Preferences](#) | [Version Control](#) | [TFS](#)

The dialog box opens when you click the **Create** button or select a workspace and click the **Edit** button in the [Manage TFS Servers and Workspaces](#) dialog box.

Use this dialog box to define a new workspace or update the existing one by selecting the necessary folders on the TFS server and mapping them to local folders.

Item	Description
Name	In this text box, specify the name of the new workspace.

Comment	In this text box, describe briefly what this workspace is intended for.
Server	This read-only field displays the URL address of the TFS server on which the new workspace will be created and which is selected in the Manage TFS Servers and Workspaces dialog box.
Owner	This read-only field displays your TFS user name.
Computer	This read-only field displays the name of your computer in the network domain.

Working folders

In this area, define mappings between the necessary folders on the TFS server and local folders.

Tip

You can map a folder including all its subfolders recursively or map each subfolder separately.

Item	Description
Status	From this drop-down list, select the status of a new mapping.
Local path	In this text box, specify the path to the local folder. Use the  button, if necessary.
Server path	In this text box, specify the path to the corresponding folder on the server.
Add	Click this button to create a new mapping.
Remove	Click this button to remove the selected mapping from the list.

See Also

- Concepts:
- [Version Control with PhpStorm](#)

- Procedures:
- [Version Control with PhpStorm](#)

- Reference:
- [TFS](#)
 - [Manage TFS Servers and Workspaces](#)

- Web Resources:
- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>

Edit Check-in Policies Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

[File](#) | [Settings](#) | [Version Control](#) | [TFS](#)

[PhpStorm](#) | [Preferences](#) | [Version Control](#) | [TFS](#)

The dialog box opens when you select an entry in the **Server/Workspace** list and click the **Check-in Policies** button in the [Manage TFS Servers and Workspaces](#) dialog box.

A check-in policy is a rule that is executed before every check-in to ensure that the selected changeset is OK to commit. **Standard policies** are stored on the server and are executed on the client machines.

Custom policies are implemented as [custom plugins](#) to PhpStorm. The IDs of these plugins are stored on the server, while the policies themselves are applied locally. Therefore, to enable the use of a policy in a team, all the team members should install the corresponding plugin.

Use this dialog box to manage the list of the custom project policies to be applied when checking in to the selected workspace and to override the default PhpStorm-wide policies for the project, if necessary.

In this section:

- [Check-in Policies](#)
- [Compatibility](#)

Check-in policies

Item	Description
Team Project	From this drop-down list, select the name of the project to specify the policies for.
Policy Type	This read-only field shows the available policies.
Description	This read-only field shows brief descriptions of policies.
Enabled	When this check box is selected, the policy next to it is mandatory during check-in.
Add	Click this button to open the Add Check-in Policy dialog box where you can define a new check-in policy.
Edit	Click this button to open the Edit Check-in Policy dialog box where you can re-define the selected check-in policy.
Remove	Click this button to remove the selected check-in policy from the list.

Compatibility

Use the controls in this area to suppress applying the default PhpStorm-wide check-in policy settings to the current project.

Item	Description
Override default settings for team project <project name>	Select the check box to discard the default policy settings within the scope of the current project and re-define the settings by selecting or clearing the corresponding check boxes below. <ul style="list-style-type: none"> Evaluate Team Explorer policies: select this check box to have the <code>Microsoft Team Explorer</code> policy definitions installed and executed on the client machine. Evaluate Teamprise policies: select this check box to have the <code>Teamprise</code> policy definitions installed and executed on the client machine. Warn about not installed policies: select this check box to have warnings displayed in case the specified policy definition is not installed.

See Also

- Concepts:
- [Version Control with PhpStorm](#)

- Procedures:
- [Version Control with PhpStorm](#)

- Reference:
- [TFS](#)
 - [Manage TFS Servers and Workspaces](#)
 - [Create Workspace](#)

- Web Resources:
- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>

XSLT File Associations

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

[File | Settings | XSLT File Associations for Windows and Linux](#)
[PhpStorm | Preferences | XSLT File Associations for Mac OS](#)

Use this dialog box to associate an XSLT stylesheet file with XML files. This is necessary to enable error highlighting and enhanced completion for element and attribute names in XSLT node-selections

Item	Description
 Project XSLT Files	The pane shows all the XSLT files in a project tree view, grouped by modules and their content roots.
 Associated Files	The pane shows all the XML file associated with the selected XSLT file.
	Click this button to open the Select Path dialog box, where you can choose an XML file to associate with the selected XSLT file.
Note	
	The button is available only when an XSLT file is selected in the Project XSLT Files pane.
	Click this button to cancel the association between the XML file selected in the Associated Files pane and the XSLT file selected in the Project XSLT Files pane.

See Also

- External Links:
- [File Associations](#)

- Web Resources:
- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>

2.0+

Inspections

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

[File | Settings | Inspections](#)

Ctrl+Alt+S Meta Comma



[PhpStorm | Preferences | Inspections](#)

Ctrl+Alt+S Meta Comma



Use this dialog box to configure [code inspections and inspection profiles](#). The page is divided into the following areas:

- [General Profiles Options](#)
- [Current Profile Toolbar](#)

- [Scopes Toolbar](#)
- [Search Area](#)
- [Inspection Options](#)

General profiles options

Item	Description
Profile	From this drop-down list, select the name of the profile to configure.
Add	Click this button to add a new profile. The profile is created based on the <code>default profile</code> .
Copy	Click this button to create a new profile based on the current profile.
Delete	Click this button to delete the selected profile.
Import	Click this button to import the profile from an <code>xml</code> file.
Export	Click this button to export the selected profile as an <code>xml</code> file.
Activate	Click this button to activate the selected profile. The active profile is applied to the project files, thus if you need to switch to another profile, you need to activate it. If this button is disabled, that means the selected profile is active now, or defined as IDE-level profile.
	Note
	If you modify inactive profile and then click OK , or Apply , your changes are saved, but profile does not become active.
Share profile	Select this check box to make the selected profile available for your team, i.e to make it the project-level profile. If you create IDE-level profile for your own purposes, clear this check box.

Current profile toolbar

Item	Shortcut	Description
	Ctrl+Add Command Add or Ctrl+Equals Command Equals	Click this button to have all inspection nodes expanded.
	Ctrl+Subtract Command Subtract or Ctrl+Minus Command Minus	Click this button to have all inspection nodes collapsed.
	Ctrl+RCommand R	Click this button to abandon the modifications and have the current profile reset to defaults.
		Click this button to have all the check boxes of the profile cleared and thus disable all the profile inspections.
		Click this button to lock the current profile and prevent any changes to it if PhpStorm is updated and new inspections appear. You can only change such profile manually.

Scopes toolbar

Item	Shortcut	Description
	InsertInsert	Click this button to specify the scope for the selected inspection.
		Note
		If an inspection is enabled and no scope is specified for it, the inspection is applied to all project sources.
	DeleteDelete	Click this button to delete the selected scope for the current inspection.
		Use these buttons to arrange the order of scopes for the selected inspection.

Search area

Item	Description
	Use this text box to search through the list of inspections. As you type a search string, the matching inspections are highlighted. To finalize the search, press <code>Enter</code> . The used search strings are memorized in the history list.
	Click this button to clear the search history.

Inspection options

Item	Description
Description	This read-only field shows the description of the selected inspection.
Options	Use the controls in this area to configure the inspection severity , and the other options, if they are available. <ul style="list-style-type: none"> • Severity - from this drop-down list, select the desired severity to assign to the current inspection. The default options are: <ul style="list-style-type: none"> ◦ As typo ◦ As server problem ◦ As info ◦ As warning ◦ As error • - click this button to open the Severities Editor dialog box, where you can change the color scheme of the selected severity, add and delete severity levels, and

change their priority by re-arranging them in the list.

- Additional options for individual inspections.

Note

This area is only available when the check box next to the desired inspection is selected.

See Also

Concepts:

- [Scope](#)
- [Code Inspection](#)

Procedures:

- [Customizing Profiles](#)
- [Defining Scope-Profile Combination](#)
- [Configuring Inspection Severities](#)

Reference:

- [Scopes](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

IDE Settings

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac 

File | Settings | IDE Settings

Ctrl+Alt+S Meta Comma



PhpStorm | Preferences | IDE Settings

Ctrl+Alt+S Meta Comma



- [Debugger](#)
- [Appearance](#)
- [Editor](#)
- [External Diff Tools](#)
- [External Tools](#)
- [File Templates](#)
- [File Types](#)
- [General](#)
- [HTTP Proxy](#)
- [GitHub](#)
- [Images](#)
- [Intentions](#)
- [Keymap](#)
- [Live Templates](#)
- [Menus and Toolbars](#)
- [Notifications](#)
- [Plugins](#)
- [Passwords](#)
- [Quick Lists](#)
- [TODO](#)
- [Updates](#)
- [Usage Statistics](#)
- [Web Browsers](#)
- [XPath Viewer](#)
- [XSLT](#)

See Also

Procedures:

- [Configuring IDE Settings](#)
- [Accessing the IDE Settings](#)

Getting Started:

- [Configuring IDE Settings](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Debugger

Previous | [Next](#) | [See Also](#) | [Comments](#)

[File](#) | [Settings](#) | [Debugger for Windows and Linux](#)
[PhpStorm](#) | [Preferences](#) | [Debugger for Mac OS](#)

In this section:

- [Debugger. JavaScript](#)

See Also

Concepts:

- [Running and Debugging](#)

Procedures:

- [Debugging](#)

Reference:

- [Debug Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Debugger. JavaScript

Previous | Next | [See Also](#) | [Comments](#)

Use this tab to configure JavaScript debug options.

Item	Description
Value tooltip delay (ms)	Define the delay in milliseconds between the moment when the mouse pointer hovers over an object in editor, and the moment when a tooltip with the object's value is displayed.
Show DOM properties	Select, whether you want to view DOM properties in the tab, or not.
Show function values	Select, whether you want to view function values in the tab, or not.
Show only user-defined functions	Select, whether you want to view user-defined functions only in the tab, or not.
Do not step into scripts	Select this check box to suppress stepping into the specified scripts while debugging. Use Add/Remove buttons to populate the list of the scripts to be skipped.

See Also

Concepts:

- [Running and Debugging](#)
- [Types of Breakpoints](#)

Procedures:

- [Debugging](#)

Reference:

- [Debug Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Appearance

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

[File](#) | [Settings](#) | [IDE Settings](#) | [Appearance](#)

Ctrl+Alt+S Meta Comma



[PhpStorm](#) | [Preferences](#) | [IDE Settings](#) | [Appearance](#)

Ctrl+Alt+S Meta Comma



Use this page to change the overall look and feel of your IDE.

- [UI Options](#)
- [Transparency](#)

- [Window Options](#)

UI options

Option	Description
Cyclic scrolling in list	Select this check box to enable scrolling through a list by jumping from the last item to the first one and vice versa.
Show icons in quick navigation	Select this check box to have the module and module icon shown in the quick navigation pop-up menu (Ctrl+Ctrl/Ctrl+Shift+Ctrl+Shift/Ctrl+Shift+Alt+NCommand Shift Alt N).
Automatically position mouse cursor on default button	Select this check box to have the mouse pointer placed at the default button when a dialog box opens. If the check box is not selected, the pointer location does not change.
Look and feel	Use this drop-down list to select a Look and Feel configuration. The list contains all the Look & Feel configurations registered with the UI manager.
Override default fonts by (not recommended)	<p>Note</p> <p>This check box is not available, when the default look and feel is selected in the Look and Feel field.</p> <p>If you choose a non-default look and feel, select this check box to enable specifying font family and size to be used instead of the default one.</p> <p>Note</p> <p>When first installed, PhpStorm takes Windows default font size and style.</p>

Transparency

Note

The transparency mode is available only for Windows 2000/Windows XP or higher.

Option	Description
Use transparent mode for floating windows	<p>Select this check box if you want floating tool windows to become semi-transparent when inactive (i.e. not in focus).</p> <p>Note</p> <p>The Delay and Ratio fields are enabled only after you select the Use transparent mode... check box.</p>
Delay (ms)	In this text box, specify the time in milliseconds after which an inactive floating tool window becomes transparent.
Ratio	Specify the transparency ratio for the floating tool windows using the sliding scale.
	<p>Note</p> <p>The higher the ratio, the more transparent and less visible the tool windows are.</p>

Window options

Option	Description
Animate windows	<p>Select this check box to have undocked tool windows slide with the animation effect.</p> <p>Note</p> <p>This option applies only when a tool window is undocked.</p>
Show memory indicator	Select this check box to show the Memory Indicator on the Status Bar .
Show tool window bars	Select this check box to display tool window bars.
Show tool window numbers	<p>Select this check box to show tool window quick access numbers on the tool window buttons.</p> <p>You can use the Alt+numberAlt number shortcuts regardless of this setting and change the shortcuts in the Keymaps dialog.</p> <p>Note that the tool window mnemonics show up only when the corresponding keybindings have the format Alt+n, where n is an integer number in the range from 1 to 9. In the case of a different keyboard shortcut, the mnemonics are not displayed.</p>
Disable mnemonics in menu	Select this check box to hide underlining of hot keys in the PhpStorm menus.

See Also

- Concepts:
- [Guided Tour Around PhpStorm User Interface](#)
 - [PhpStorm Tool Windows](#)
- Procedures:
- [Configuring IDE Settings](#)

Reference:

- [Tool Windows Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Editor

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

[File](#) | [Settings](#) | [IDE Settings](#) | [Editor](#)

Ctrl+Alt+S Meta Comma



[PhpStorm](#) | [Preferences](#) | [IDE Settings](#) | [Editor](#)

Ctrl+Alt+S Meta Comma



Use the Editor page of the Settings dialog box to configure the Editor's behaviour and customize its view.

Item	Description
Advanced mouse usages	
Honor "CamelHumps" word settings when selecting using double click	Select this check box to have PhpStorm invoke the CamelHumps selection when words are selected by double-clicking. This feature works only if the Use 'CamelHumps' words option is enabled.
Change font size (Zoom) with Ctrl+MouseWheel	If this check box is selected, the particular editor font size can be changed by rolling the mouse wheel while holding the CtrlCtrl button. The font size is changed only in the current editor tab, editors opened later are not affected.
Enable Drag'n'Drop functionality in editor	If this check box is selected, you can drag code fragments in the editor. Refer to the section Using Drag-and-Drop in the Editor .
Scrolling	
Smooth scrolling	Select this check box to enable smooth scrolling in the editor.
Prefer scrolling editor canvas to keep caret line centered	Click this option to choose scrolling editor canvas and keeping the caret in place. Keeping the caret in place and scrolling the editor canvas can be helpful in course of debugging session . As you step through the lines of code, the editor canvas scrolls, while the line at caret is always in the center of the screen.
Prefer moving caret line to minimize editor scrolling	Click this option to choose moving the caret. When you step through the lines of code during the debugging session , the caret moves down, and the editor canvas doesn't scroll until the caret line reaches the bottom of the screen.
Limits	
Maximum number of contents to keep in clipboard	In this text box, specify how many code blocks can be kept in clipboard.
Recent file limit	In this text box, specify how many file names can be included in the list of recent files.
Console commands history size	In this text box, specify how many console commands will be included in the console history and can be browsed through.
Enable in place refactorings	Select or clear this check box to enable or disable in-place refactorings for %language%. The <code>in-place</code> in connection with the refactorings means specifying all or most of the information necessary for the refactoring by typing, right in the editor. All the affected code fragments are highlighted and change as your type. If appropriate, additional refactoring options are selected in corresponding option boxes. The in-place refactoring mode is available for the following refactorings: <ul style="list-style-type: none"> • Introduce Constant • Introduce Field • Introduce Variable • Rename If the option is off (i.e. the check box is not selected), the refactoring settings for all of the refactorings are specified in the corresponding dialogs.
Strip trailing spaces on Save	From this drop-down list, select the mode in which PhpStorm will handle trailing spaces in the end of lines on file saving: <ul style="list-style-type: none"> • <code>Modified lines</code> - Strips trailing spaces only in the end of modified lines. • <code>All</code> - Strips trailing spaces in all lines. • <code>None</code> - Does not strip trailing spaces.
Ensure blank line before end of file on Save	Select this check box to have PhpStorm automatically add an empty line in the end of a file during the save procedure.
Formatting	
Show "Reformat Code" dialog	If this check box is selected, the Reformat Code dialog box will appear every time you try to reformat code, where you can specify the scope of reformatting. Otherwise, PhpStorm will reformat code silently.
Show "Optimize Imports" dialog	If this check box is selected, the Optimize Imports dialog box will appear every time you try to optimize import statements. Otherwise, PhpStorm will optimize imports silently.

Virtual Space

- Use soft wraps in the editor** If this check box is selected, the `soft wraps` (or `word wraps`) are used in the editor.
Horizontal scroll bar in general is not shown, when this option is enabled. However, in certain cases, when a line cannot be "soft-wrapped", the horizontal scroll bar still appears (for example, a line consists of a single string that is wider than the visible area.)
- Use soft wraps in the console** If this check box is selected, the `soft wraps` (or `word wraps`) are used in the console.
- Use custom soft wrap indent** Select this check box to use custom indentation for soft wraps on resizing the editor. Specify the indent value in the text field to the right.
This field is only available when at least one of the **Use soft wraps...** check boxes is selected.
- Show all soft wraps** If this check box is selected, the soft wrap characters `␣` `␣` will be always shown at the end of each line, and at the beginning of each next line.
Otherwise, the soft wrap characters will be shown in the active logical line only.
- Allow placement of caret after end of line** If this check box is cleared, the caret never rests after the last symbol in a line.
- Allow placement of caret inside tabs** Select this check box to allow placing the caret inside tab characters. The reason is that each tab character shows in the editor as a set of 'virtual' space characters.
- Show virtual space at file bottom** If this check box is selected, the currently edited line (even if it is the final line) can be scrolled to the top of the screen. PhpStorm adds the necessary amount of virtual lines.

Highlight on Caret Movement

- Highlight matched brace** Select this check box to have PhpStorm highlight pairs of opening/closing braces when you position the caret right before the opening or right after the closing one. It also works for HTML and XML tags.
- Highlight current scope** Select this check box to have PhpStorm highlight the available scope for the code typed in the current caret location.
- Highlight usages of element at caret** Select this check box to have PhpStorm highlight all usages of the element at which the caret is currently positioned.

Error Highlighting

- Error stripe mark min height (pixels)** In this text box, specify the minimum size of the error and warning stripes.
- Autoreparse delay (ms)** In this text box, specify the time period after when PhpStorm starts reparsing the entered text.
- 'Next Error' action goes to errors first** Select this check box to have PhpStorm pass through the errors and then through the warnings.
Clear this check box to have PhpStorm pass through the errors and warnings sequentially.

See Also

Procedures:

- [Advanced Editing Procedures](#)

Getting Started:

- [Familiarize Yourself with PhpStorm Editor](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Editor. Smart Keys

Previous | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

[File](#) | [Settings](#) | [IDE Settings](#) | [Editor - Smart Keys](#)

Ctrl+Alt+S Meta Comma



[PhpStorm](#) | [Preferences](#) | [IDE Settings](#) | [Editor - Smart Keys](#)

Ctrl+Alt+S Meta Comma



Use this page to enable or disable specific smart keys and to define which actions you want to be invoked automatically.

Item	Description
Home	When this check box is selected, on pressing <code>HomeHome</code> , the caret is positioned at the first non-space character of the current line. Pressing <code>HomeHome</code> subsequently moves the caret from the <i>Smart Home position</i> to the first column and back.
End (on blank line)	When this check box is selected, on pressing <code>EndEnd</code> in an empty line, the caret is positioned with the indent, which PhpStorm assumes to be reasonable in the current code point (indentation is based on the current Code Style Settings).
Insert pair bracket	Select this check box to have PhpStorm automatically add a closing round or square bracket for each typed opening round or square bracket, respectively.
Insert pair quote	Select this check box to have PhpStorm automatically add a closing single or double quote for each typed opening single or double quote, respectively.
Use 'CamelHumps' words	Select this check box to have PhpStorm discern separate words within CamelHump names. Words within a name should start with a capital letter or an underscore. This option impacts some editor actions, for example: <ul style="list-style-type: none"> • Caret Move (<code>Ctrl+ArrowCtrl Arrow</code>) • Caret Move with Selection (<code>Shift+Ctrl+ArrowShift Ctrl Arrow</code>)

- **Select Word at Caret** (Ctrl+W/Command W)
- **Delete to Word Start/End** (Ctrl+BackSpace/Alt BackSpace and Ctrl+Delete/Alt Delete respectively)
- **Double-clicking**

Surround selection on typing quote or brace If this check box is selected, the selected text on typing a quote, double-quote or brace, will be surrounded with these characters. If this check box is not selected, then the typed quotes, double-quotes or braces will replace the selection.

Enter Use this area to define the actions to be invoked by pressing Enter/Enter.

- **Smart Indent** - select this check box to have PhpStorm add a new line and position the caret in it, with the indent that PhpStorm assumes to be reasonable in the current point of code (indentation is based on the current [Code Style Settings](#)).
- If the check box is cleared, upon pressing Enter/Enter in a blank line, PhpStorm adds a new line and positions the caret at the current non-space character column.
- **Insert pair '}'** - select this check box to have PhpStorm automatically position a closing brace } at the proper column when Enter/Enter is pressed in an empty line. In this case PhpStorm seeks backward for the nearest unclosed opening brace { and places the closing one at the corresponding indentation level.
- **Insert documentation comment stub** - this check box defines the behavior on pressing Enter/Enter after opening tag.
 - If this check box is selected, PhpStorm generates a documentation comment stub. For the function comments, this stub contains the required tags (@param tags for each parameter declared in the signature, and @return). Refer to [Creating Documentation Comments](#) for details.
 - If this check box is not selected, only the closing tag is generated.

Reformat on paste Use this drop-down list to specify how to place pasted code blocks. The available options are:

- **None** - The pasted code is inserted at the caret location as plain text without any reformatting or indenting.
- **Indent Block** - The pasted code block is positioned at the proper indentation level, according to the current [Code Style Settings](#), but its inner structure is not changed.
- **Indent Each Line**- Each line of the pasted code block is positioned at the proper indentation level, according to the current [Code Style Settings](#).
- **Reformat Block**- The pasted code block is reformatted according to the current [Code Style Settings](#).

Tip

This feature is applicable to lines that contain the trailing line feed characters.

XML/HTML In this area, define the actions to be invoked automatically when editing XML or HTML code.

- **Automatically insert closing tag**: select this check box to have PhpStorm automatically insert a closing XML or HTML tag upon entering the corresponding opening one.
- **Automatically insert required attributes**: select this check box to have PhpStorm display a template with all mandatory attributes of the typed tag.
- **Automatically start attribute**: select this check box to have PhpStorm display a template with the first mandatory attribute of the typed tag.
- **Enable Zen Coding**: select this check box to use native [Zen selectors](#) in PhpStorm.
- **Expand abbreviation with**: in this drop-down box, specify the default key to expand Zen selectors with.

Note

This key will also by default expand [Zen Coding live templates](#). You can [re-define this default setting](#) for each specific live template.

See Also

Procedures:

- [Cutting, Copying and Pasting](#)
- [Completing Statements](#)
- [Creating Documentation Comments](#)

Reference:

- [Editor. Code Completion](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Editor. Appearance

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

File | Settings | IDE Settings | Editor - Appearance

Ctrl+Alt+S/Command Meta Comma

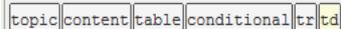


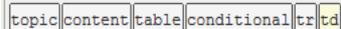
PhpStorm | Preferences | IDE Settings | Editor - Appearance

Ctrl+Alt+S/Command Meta Comma

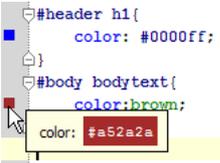


Use this page to customize the appearance of the Editor.

Item	Description
Use antialiased font	Select this check box to enable showing fonts without distortions.
Caret blinking (ms)	Select this check box to make the caret blink with the specified period (in milliseconds).
Use block caret	Select this check box to have the block caret applied in the <i>Insert</i> mode and the usual caret applied in the <i>Overwrite</i> mode. Clear this check box to have the usual caret applied in the <i>Insert</i> mode and the block caret applied in the <i>Overwrite</i> mode.
Show right margin (configured in Code Style options)	Select this check box to have a thin vertical line at the right margin of the editor displayed. Refer to the description of the General page of the <i>Code Style</i> settings.
Show line numbers	Select this check box to have line numbering shown in the left gutter area.
Show method separators	Select this check box to have thin lines displayed in PHP and JavaScript classes to separate functions from field declarations.
Show whitespaces	Select this check box to have PhpStorm display white spaces or tabs (depending on the <i>Code Style</i> settings made in the <i>Tabs and Indents</i> pane of the General page).
Show vertical indent guides	Select this check box to have PhpStorm display vertical lines in the editor to indicate positions of indents and thus facilitate typing, manual formatting, reading, and maintaining code.
Show HTML breadcrumbs (Reopen editor for changes to take effect)	Select this check box to show a breadcrumb trail on top of the editor tab for an HTML file. 
Show breadcrumbs for XML files	Select this check box to show a breadcrumb trail on top of the editor tab for an XML file. Reopen the editor for the changes to take effect. 
Show CSS color preview icon in gutter	Select this check box to show color preview icons for the color values. See Changing Color Values in Style Sheets .



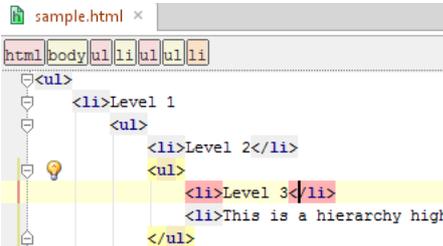
See [Changing Color Values in Style Sheets](#).



Enable HTML/XML tag tree highlighting

Select this check box to show the hierarchy of tags highlighted with different colors. If this option is enabled, you can define the following options:

- Levels to highlight:** specify the depth of hierarchy to be highlighted.
- Opacity:** specify brightness of highlighting



Note

Highlighting is activated when there is more than one tag with the same name in the hierarchy.

Show SASS color preview icon in gutter

Select this check box to show color preview icons for the color values. See [Changing Color Values in Style Sheets](#).

See Also

- Procedures:
- [Advanced Editing Procedures](#)
- Reference:
- [Editor](#)
 - [Accessing the IDE Settings](#)
- Web Resources:
- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>

Editor. Colors and Fonts

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: (Mac)

File | Settings | IDE Settings | Editor - Colors and Fonts

Ctrl+Alt+S Meta Comma



PhpStorm | Preferences | IDE Settings | Editor - Colors and Fonts

Ctrl+Alt+S Meta Comma



In this section:

- [Scheme](#): use this section to select Colors and Fonts scheme. PhpStorm suggests several pre-defined schemes, one of them being the default.
- [Scheme settings](#): use the pages to change font type, colors and highlighting for the supported languages, consoles, Debugger, differences viewer, file statuses, and scopes. Note that besides color settings, highlighting is determined by your [Inspection Profile](#). For example, if you do not want unused symbols to be highlighted, turn off the **Unused Symbol** inspection in the profile being used.

Scheme

Item	Description
Scheme name	From this drop-down list, select the Colors & Fonts scheme to use in your workspace. When a non-default scheme is selected, the node name changes its color to blue: Colors & Fonts.
	Warning The pre-defined schemes are not editable. To change colors and fonts settings in such a scheme, create its copy.
Save As	Click this button to save the currently selected Colors & Fonts settings as a new scheme. After saving, the scheme settings become editable.
Delete	Click this button to delete the current scheme.

Scheme settings

Page	Description
Font	Use this page to define the family of the font to be used in the editor, its size, and line spacing. Click the browse button to select a new font family. Note This button is only available for the editable schemes. The Select Font dialog box displays a list of mono spaced fonts. To choose among all fonts available in the system, clear the Show only mono spaced fonts check box. If an entered character cannot be displayed in the current font, the closest match is used.
General	Use this page to customize the font type and colors for the editor textual components, specified in the list.
Debugger, XML, HTML, SASS, XPath, JSP, CSS, PHP, Smarty, SQL, JavaScript, YAML, Diff	Use these pages to customize font type and colors for the files of the corresponding types and the PhpStorm components.
Custom	Use this page to customize the font type and colors in the user-defined file types, in particular, for custom keywords highlighting .
File Status	Use this page to customize font type and colors to visually denote the current file status with regards to a VCS repository .
Scope Based	Use this page to customize colors and highlighting for class members assigned to various scopes. To edit a scope, click the Edit Scopes button below the pane with a list of existing scopes. The Scopes dialog box opens.

See Also

Concepts:

- [Scope](#)

Procedures:

- [Configuring Colors and Fonts](#)

Reference:

- [Editor](#)
- [Differences Viewer](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Editor. Editor Tabs

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts:

File | Settings | IDE Settings | Editor - Editor tabs

Ctrl+Alt+S Meta Comma



PhpStorm | Preferences | IDE Settings | Editor - Editor tabs

Ctrl+Alt+S Meta Comma



Use this page to configure the appearance of editor tabs and tab headers, specify their positioning on the screen, and define the tab closing policy.

Item	Description
Tab Appearance	
Placement	Use this drop-down list to define the location of the editor tab. The available options are: <ul style="list-style-type: none"> • Top - the default setting. • None - select this option to have single editor without any tabs displayed. • Bottom • Right • Left
Show tabs in single row	Select this check box to have headers of currently opened editor tabs displayed in a single row. As a result, some tab headers may be invisible. If this check box is not selected, headers of all the currently opened tabs are displayed, possibly, in several rows.
Hide file extensions in editor tabs	Select this check box to have only file names displayed in editor tab headers.
Show directory in editor tabs for non-unique file names	If this check box is selected, the editor tabs will show the file name together with the parent directory name; if this check box is not selected, only the file name will be included in the editor tab.
Show "close" button on editor tabs	Select this check box to have the Close Active Editor button displayed in editor tab headers.
Mark modified tabs with asterisk	Select this check box to have tabs with modified but unsaved contents marked with an asterisk in the tab header. By default, this check box is cleared, because PhpStorm saves files transparently and you do not need to do it manually.
Tab Closing Policy	
Tab limit	In this text box, set the maximum number of editor tabs to display.
When number of opened editors exceeds tab limit	In this area, specify which editor tab should be closed when the tab limit is reached and the user attempts to open a new file. The available options are: <ul style="list-style-type: none"> • Close non-modified files first - if this option is selected, PhpStorm examines the tabs in the order they were opened and closes the first tab with content that has not been modified. • Close less frequently used files - if this option is selected, PhpStorm closes the tab with the less frequently modified content.
When closing active editor	In this area, specify which editor tab to activate when closing the currently active tab. The available options are: <ul style="list-style-type: none"> • Activate left neighbouring tab - if this option is selected, PhpStorm activates the closest tab to the left from the tab being closed. • Activate most recently opened tab - if this option is selected, PhpStorm activates the tab with the file which was opened last.

See Also

Procedures:

- [Advanced Editing Procedures](#)

Getting Started:

- [Familiarize Yourself with PhpStorm Editor](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Editor. Code Folding

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts:

File | Settings | IDE Settings | Editor - Code Folding

Ctrl+Alt+S Meta Comma



PhpStorm | Preferences | IDE Settings | Editor - Code Folding

Ctrl+Alt+S Meta Comma



Item	Description
Show code folding outline	Select this check box to have a thin line displayed between the folding toggles for an unfolded block of code. See Code Folding for details.
Collapse by default	In this area, specify which code blocks should be collapsed by default when a file is first opened in the editor. Select the check boxes corresponding to desired code block types.

See Also

Procedures:

- [Advanced Editing Procedures](#)
- [Code Folding](#)

Getting Started:

- [Familiarize Yourself with PhpStorm Editor](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Editor. Code Completion

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

File | Settings | IDE Settings | Editor - Code Completion

Ctrl+Alt+S Meta Comma



PhpStorm | Preferences | IDE Settings | Editor - Code Completion

Ctrl+Alt+S Meta Comma



Use this page to configure the [code completion](#), and [parameter information](#).

Item	Description
Code Completion	
Case sensitive completion	From this drop-down list, select the degree to which you want PhpStorm to take into consideration the case sensitivity when suggesting matches for code completion. The available options are:

- All: The lookup list includes only those items that match the case of all typed letters. This option is most restrictive:

```
var cat = new An
}
  m Anaconda (type) completion.js
  m A Animal (type) completion.js
```

- None: The lookup list includes all matches regardless of their case.

```
var salmon = new A
}
  m Animal (type) completion.js
  m Anaconda (type) completion.js
  m ActiveXObject window(predefines.AJAX.js)
  v ant completion.js
  C Arguments ECMAScript.js2
  D Array ECMAScript.js2
  i Attr predefines.DOMCore.js
```

- First letter: The lookup list includes only the items with the first letter matching.

```
var salmon = new A
}
  m Animal (type) completion.js
  m Anaconda (type) completion.js
  m ActiveXObject window(predefines.AJAX.js)
  C Arguments ECMAScript.js2
  D Array ECMAScript.js2
  i Attr predefines.DOMCore.js
```

Autocomplete when only one choice When the check boxes in this section are selected, PhpStorm doesn't show a lookup list for the corresponding completion type in cases when only one variant of code completion is available, and completes code automatically.

Autocomplete common prefix If this check box is selected, PhpStorm automatically completes the prefix of a name, if all the matches in the lookup list have a common prefix. For example, consider the following entry in the editor:

```
this.getAttr
```

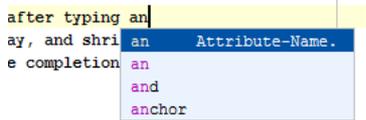
After pressing Ctrl+Space/Command Space, the prefix is automatically completed to the strings that all the methods start with:

```
this.getAttribute
  m getAttribute Element (predefines.DOMCore.js)
  m getAttributeNode Element (predefines.DOMCore.js)
  m getAttributeNodeNS Element (predefines.DOMCore.js)
  m getAttributeNS Element (predefines.DOMCore.js)
```

Sort lookup items lexicographically If this check box is selected, the entries in the suggestion list will be sorted according to their lexical order. If this check box is not selected, the entries in the suggestion list will be sorted by relevance.

Note that the check box defines the default behavior. You can change it any time by clicking the **A** or **π** icons in the suggestion list. Refer to the section [Using Suggestion List](#) for details.

2.0+ Autopop code completion in (ms) Select this check box, if you want suggestion list to appear after typing anything. So doing, the suggestion list appears after the specified delay, and shrinks as you type.



Preselect the first suggestion

Choose the way PhpStorm treats the suggestion list in the code completion autopopup. The following options are available: If the check box is not selected, PhpStorm will not suggest code completion automatically.

- Always. The first element in the suggestion list is always selected.
- 'Smart'. The first element in the suggestion list can be selected depending on the context.
- Never. The first element in the suggestion list is not selected.

If the suggestion list has no focus, you can either place the focus to the desired element using the arrow keys, or select the first element with the `Tab`.

If the suggestion list has the focus, you can select an element as usual, with the `Space`, `Enter`, or `.`

Autopopup documentation (ms)
For explicitly invoked completion

Select this check box to have PhpStorm automatically show a pop-up window with the documentation for the class, method, or field currently highlighted in the lookup list.

In the text field to the right, specify the delay (in milliseconds), after which the pop-up window should appear.

If this check box is not selected, use `Ctrl+Q` to show quick documentation for the element at caret (class, method or field).

Quick documentation window will automatically pop up with the specified delay in those cases only, when code completion has been invoked explicitly. For the automatic code completion list, documentation window will only show up on pressing `Ctrl+Q`

Parameter Info

Autopopup in (ms)

Select this check box to have PhpStorm automatically show a pop-up window with all available method signatures, when an opening bracket is typed in the editor, or a method is selected from the lookup list.

In the text field to the right, specify the delay (in milliseconds) after when the pop-up window should appear.

If this check box is not selected, use `Ctrl+Meta P` to show the parameter info.

Show full signatures

If this check box is selected, the parameter info displays full signatures, including the method name and returned type.

See Also

Procedures:

- [Auto-Completing Code](#)

Reference:

- [Editor](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Editor. Auto Import

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

File | Settings | IDE Settings | Editor - Auto Import

Ctrl+Alt+S Meta Comma



PhpStorm | Preferences | IDE Settings | Editor - Auto Import

Ctrl+Alt+S Meta Comma



- [XML](#)

XML

Item	Description
Show import pop-up	Automatically display an import pop-up dialog box when typing the name of an unbound namespace.

See Also

Procedures:

- [Creating Imports](#)

Getting Started:

- [Familiarize Yourself with PhpStorm Editor](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

External Diff Tools

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

[File](#) | [Settings](#) | [IDE Settings](#) | [External Diff Tools](#)

Ctrl+Alt+S Meta Comma



[PhpStorm](#) | [Preferences](#) | [IDE Settings](#) | [External Diff Tools](#)

Ctrl+Alt+S Meta Comma



In this dialog box, enable PhpStorm to use external tools for comparing files and folders.

Item	Description
Use external tool to compare folders	Select this check box to have PhpStorm use an external tool for comparing folders. In the text box below, specify the path to the executable file of the desired external tool. Use the Browse button  , if necessary.
Use external tool to compare files	Select this check box to have PhpStorm use an external tool for comparing files. In the text box below, specify the path to the executable file of the desired external tool. Use the Browse button  , if necessary.

See Also

Procedures:

- [Configuring IDE Settings](#)
- [Comparing Files](#)

Reference:

- [External Tools](#)
- [Differences Viewer](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

External Tools

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

[File](#) | [Settings](#) | [IDE Settings](#) | [External Tools](#)

Ctrl+Alt+S Meta Comma



[PhpStorm](#) | [Preferences](#) | [IDE Settings](#) | [External Tools](#)

Ctrl+Alt+S Meta Comma



Support for external tools enables integration with the third-party applications without creating plugins. Configuration options for the external tools enable passing contextual information (like the current file name or project source path) to an application via command-line arguments.

The list of external tools can be shared among the development team. Once added, the external tools appear as the new menu commands.

Use this dialog box to manage the list of external tools to be accessible from PhpStorm through the **Tools** menu and context menus.

In this topic:

- [List of External Tools](#)
- [Toolbar](#)

List of external tools

The area displays a tree of available external tools organized by groups they belong to.

- To enable a tool, select the check box next to it.
- To enable an entire group, select the check box next to the group name.

Toolbar

Item	Description
Add	Click this button to open the Edit Tool dialog box and configure access to an external tool from scratch.
Copy	Click this button to open the Edit Tool dialog box and configure access to an external tool based on the copy of the selected configuration.
Edit	Click this button to open the Edit Tool dialog box and change the selected configuration.
Remove	Click this button to delete selected entries from the list of external tools.
Move Up / Move Down	Use these buttons to change the order of entries in the list of external tools. The order of items in the list defines the order in which the tools are presented within a group on the Tools menu.

See Also

Concepts:

- [Project and IDE Settings](#)

Procedures:

- [Configuring Third-Party Tools](#)

Reference:

- [Edit Tool Dialog](#)
- [Output Filters Dialog](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Edit Tool Dialog

Previous | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

File | **Settings** | **IDE Settings** | **External Tools**

Ctrl+Alt+S Meta Comma



PhpStorm | **Preferences** | **IDE Settings** | **External Tools**

Ctrl+Alt+S Meta Comma



The dialog box opens when you click the **Add**, **Copy**, or **Edit** button in the [External Tools](#) dialog box.

Use this dialog box to enable using a specific external tool from PhpStorm, specify the menus to access it from, define where to display the tool's output, and configure the filters to distinguish it from other output. Based on these filters, PhpStorm displays paths to the tool's output files as links. When you click such link, the corresponding file is opened in the editor.

Item	Description
Name	In this text box, specify the name to identify the tool in the PhpStorm menus.
Group	Use this drop-down list to specify the group the current tool belong to. Type a new name or select an existing group from the list.
Description	In this text box, provide an optional description of the tool.
Menu	In this area, specify the menus the tool should be available from. The available options are: <ul style="list-style-type: none"> • Main menu - select this check box to have the tool command added to the Tools menu. • Editor menu - select this check box to have the tool command added to the context menu of the Editor. • Project views - select this check box to have the tool command added to the context menu of the Project tool window. • Search results - select this check box to have the tool command added to the context menu of the Search Results area in the Find tool window.
Output filters	Click this button to open the Output Filters dialog box where you can define the filters to distinguish the output of the external tool from other output. Based on these filters, PhpStorm displays output file paths in output messages as links. When you click such link, the corresponding file is opened in the editor.
Open console	When this check box is selected, PhpStorm opens a special console to display the output of the configured external tool.
Synchronize files after execution	When this check box is selected, PhpStorm detects all externally changed files and reloads them from a disk on completing tool execution.
Program	Use this field to specify the location of the external tool's executable file in one of the following ways: <ul style="list-style-type: none"> • Type the path explicitly in the text box. • Click the Browse button to open the Select Path dialog box and navigate to the desired location. • Click the Insert Macro button to open the Macros dialog box where you can select the desired macro from the list.
Parameters	Use this field to define the parameters of the external tool in one of the following ways: <ul style="list-style-type: none"> • Type the list of parameters in the text box. • Click the Insert Macro button to open the Macros dialog box where you can select the desired macro from the list. <p>When specifying the parameters, follow these rules:</p> <ul style="list-style-type: none"> • Use spaces to separate individual parameters. • If a parameter includes spaces, enclose the spaces or the argument that contains the spaces in double quotes, for example, <code>some "arg" or "some arg"</code>. • If a parameter includes double quotes (e.g. as part of the argument), escape the double quotes by means of the backslashes, for example, <code>- Dmy.prop=\"quoted_value\"</code>.
Working directory	Use this field to define the working directory of the external tool in one of the following ways: <ul style="list-style-type: none"> • Type the path explicitly in the text box. • Click the Browse button to open the Select Path dialog box and navigate to the desired location. • Click the Insert Macro button to open the Macros dialog box, where you can select the desired macro from the list.
Tip	If the field is left blank, the default project directory is used.

Tip

If the specified external tool is a Python script, add `-u` to the Python options. Thus you will be able to get prompt and user input before the script ends.

See Also

Concepts:

- [Plugins](#)

Procedures:

- [Configuring Third-Party Tools](#)

Reference:

- [External Tools](#)
- [Output Filters Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Output Filters Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

[File](#) | [Settings](#) | [IDE Settings](#) | [External Tools](#)

Ctrl+Alt+S Meta Comma



[PhpStorm](#) | [Preferences](#) | [IDE Settings](#) | [External Tools](#)

Ctrl+Alt+S Meta Comma



The dialog box opens when you click the **Output Filters** button in the [Edit Tool](#) dialog box. In this dialog box, manage the list of filters to distinguish the output of a specific external tool from other output. Based on these filters, PhpStorm displays paths to the tool's output files as links. When you click such link, the corresponding file is opened in the editor.

In this section:

- [Output Filters](#)
- [Toolbar](#)

Output filters

The list box displays the filters defined for a specific external tool. Use the toolbar buttons to manage the contents of the list.

Toolbar

Item	Description
Add	Click this button to open the Add Filter dialog box and configure a new filter.
Edit	Click this button to open the Add Filter dialog box and change the configuration of the selected filter.
Remove	Click this button to delete selected entries from the list of filters.
Move Up / Move Down	Use these buttons to change order of entries in the list of filters.

See Also

Concepts:

- [Plugins](#)

Procedures:

- [Configuring Third-Party Tools](#)

Reference:

- [External Tools](#)
- [Edit Tool Dialog](#)
- [Add Filter Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Add Filter Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

[File](#) | [Settings](#) | [IDE Settings](#) | [External Tools](#)

Ctrl+Alt+S Meta Comma



PhpStorm | Preferences | IDE Settings | External Tools

Ctrl+Alt+S Meta Comma



The dialog box opens when you click the **Add** or **Edit** button in the [Output Filters](#) dialog box. Use this dialog box to configure and re-configure filters to distinguish the output of a specific external tool from other output.

Item	Description
Name	In this text box, specify the name to identify the filter.
Description	In this text box, provide a description of the filter.
Regular expression to match output	In this text box, specify a regular expression according to which PhpStorm will distinguish the output of a specific external tool from other output. Type the desired expression manually, or select the relevant macro from the context menu, or assemble the macros into an expression, as necessary. The available macros are: <ul style="list-style-type: none"> • \$FILE_PATH\$ • \$LINES\$ • \$COLUMN\$ <p>PhpStorm applies the specified filtering expressions against the console output of the external tool. As soon as a pattern is hit, its part that matches the \$FILE_PATH\$ will be treated as a path to a specific file and will be displayed as a link. When you click such link, the corresponding file is opened in the editor.</p>

See Also

Concepts:

- [Plugins](#)

Procedures:

- [Configuring Third-Party Tools](#)

Reference:

- [Output Filters Dialog](#)
- [External Tools](#)
- [Edit Tool Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

File Templates

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

File | Settings | IDE Settings | File Templates

Ctrl+Alt+S Meta Comma



PhpStorm | Preferences | IDE Settings | File Templates

Ctrl+Alt+S Meta Comma



Use this page to view, edit, create, and remove [file templates](#).

The toolbar buttons and other controls are common for all the tabs of the page.

- [Toolbar](#)
- [Controls](#)
- [Tabs](#)

Toolbar

Item	Tooltip	Description
	Create Template	Click this button to create a new template in the currently opened tab. The button is available in the Templates and Includes tabs.
	Remove Template	Click this button to delete the selected template. The button is available in the Templates and Includes tabs.
	Copy Template	Click this button to create a copy of the selected template. The button is available in the Templates and Includes tabs.
	Reset to Default	Click this button to abandon all the changes made to the selected template and restore its default state. The button is available only for modified templates.
	Tip	The names of the modified templates are displayed in blue.

Controls

Item	Description
Name	In this text box, specify the name of the selected template.
Extension	In this text box, specify the extension for PhpStorm to detect files, where the selected template should be applied during creation.
Reformat according to style	Select this check box, to have PhpStorm reformat generated stub files according to the style defined in the Code Style dialog box.
Template text	<p>In this text box, specify the body of the template:</p> <ul style="list-style-type: none"> • Plain text. • #parse directives to work with template includes. • Variables to be expanded into corresponding values in the format <code>\${<variable_name>}</code>. <p>The available predefined file template variables are:</p> <ul style="list-style-type: none"> ◦ <code>\$(USER)</code> - login name of the current user. ◦ <code>\$(NAME)</code> - the name of the file that will be created. ◦ <code>\$(DATE)</code> - the current system date. ◦ <code>\$(TIME)</code> - the current system time. ◦ <code>\$(YEAR)</code> - the current year. ◦ <code>\$(MONTH)</code> - the current month. ◦ <code>\$(DAY)</code> - the current day of the month. ◦ <code>\$(HOUR)</code> - the current hour. ◦ <code>\$(MINUTE)</code> - the current minute. <p>Tip</p> <p>To include some Velocity variable in your template that you don't want to be expanded by PhpStorm on template applying, prepend it with the <code>'\'</code> character.</p> <p>For example, to use some version control keywords (such as <code>\$Revision\$</code>, <code>\$Date\$</code>, etc.) in your default class template, you need to prepend them with a <code>'\': \</code> <code>\$Revision\$</code>.</p> <ul style="list-style-type: none"> • Custom variables. Their names will be defined during the file creation.
Description	In this text box, provide some information about the template, its predefined variables, and the way they work.

Tabs

Tab	Description
Templates	<p>The tab displays the available file templates.</p> <p>Note</p> <p>The <code>Class</code> templates cannot be deleted and their names and extensions cannot be edited.</p>
Includes	The tab shows the templates of reusable fragments that may be included in the file templates.
Code	<p>The tab displays built-in snippets, that is, templates for code fragments that PhpStorm may generate in various typical situations.</p> <p>You can edit the existing snippets but you cannot create new ones.</p> <p>For any snippet that you have modified, the <code>Reset</code> button (link) is available in the upper right corner of the page which you can use to restore the initial state of the snippet.</p>

See Also

- Concepts:
- [File Templates](#)
 - [File Template Variables](#)
 - [#Parse Directive](#)
- Procedures:
- [Creating Files from Templates](#)
 - [Creating and Editing File Templates](#)
- Web Resources:
- <http://www.jetbrains.com/devnet/community/wi>
 - <http://youtrack.jetbrains.com/issues/WI>

File Types

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: 

File | Settings | IDE Settings | File Types

Ctrl+Alt+S Meta Comma



PhpStorm | Preferences | IDE Settings | File Types

Ctrl+Alt+S Meta Comma



Use this dialog box to manage the list of file types and extension patterns to be recognized by PhpStorm.

Item	Description
Recognized file types	<p>This list box displays all the default and custom file types currently supported by PhpStorm. Use the Add, Edit, and Remove buttons to manage the contents of the list box.</p> <p>Tip</p> <p>Default types cannot be edited or removed.</p>
Add	Click this button to open the New File Type dialog box and define a new custom file type there.
Edit	Click this button to open the Edit File Type dialog box and edit the selected file type there.
	<p>Note</p> <p>The button is disabled when a default file type is selected.</p>
Remove	Click this button to delete the selected file type from the list.
	<p>Note</p> <p>The button is disabled when a default file type is selected.</p>
Registered Patterns	This list box displays all the registered extensions associated with the file type selected in the Recognized file types list. Use the Add and Remove buttons to manage the contents of the list box.
Add	Click this button to open the Add wildcard dialog box and specify a new pattern using wildcards there.
Remove	Click this button to delete the selected pattern from the list.
Ignore files and folders	<p>In this text box, specify the files and folders, which you want to be ignored by PhpStorm. Such files and folders will be completely excluded from any kind of processing. By default the list includes temporary files, service files related to version control systems, etc.</p> <p>Tip</p> <p>You can specify multiple names or wildcard masks, with semicolons (;) as separators.</p> <p>Below is the default setting, in case you need to restore it:</p> <pre>CVS;SCCS;RCS;rcs;.DS_Store;.svn;.pyc;.pyo;.sbas;.IJI.*;vssver.scc;vssver2.scc;*.pyc;*.pyo;.git;</pre>

See Also

Procedures:

- [Creating and Registering File Types](#)

Reference:

- [New File Type](#)
- [Register New File Type Association Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

New File Type

Previous | Next | [See Also](#) | [Comments](#) | Shortcuts: Mac

[File](#) | [Settings](#) | [IDE Settings](#) | [File Types](#)

Ctrl+Alt+S Meta Comma



[PhpStorm](#) | [Preferences](#) | [IDE Settings](#) | [File Types](#)

Ctrl+Alt+S Meta Comma



The dialog box opens when you click the **Add** button or select a custom file type and click the **Edit** button in the [File Types](#) dialog box.

Use the dialog box to configure and re-configure presentation and highlighting of keywords, comments, numbers etc. in files of a specific custom file type. These settings make the basis for parsing files of this type in the editor.

Item	Description
Name	In this text box, specify the name of the file type.

Description	In this text box, provide an optional description of the file type.
Syntax Highlighting	In this area, specify the character strings to indicate borders of comments, the numeric system used, and configure highlighting for brackets, braces, etc. syntax elements.
Line comment	In this text box, specify the character string to indicate the start of a single-line comment.
Block comment start	In this text box, specify the character string to indicate the start of a block comment.
Block comment end	In this text box, specify the character string to indicate the end of a block comment.
Hex prefix	In this text box, specify the character string to indicate that the subsequent value is a hexadecimal number. For example, 0x.
Number postfixes	In this text box, specify the character string to indicate which numeric system or unit is used. A postfix is a trailing string of characters, for example, e-3, kg etc.
Support paired braces	Select this check box, to have paired braces highlighted.
Support paired brackets	Select this check box, to have paired brackets highlighted.
Support paired parens	Select this check box, to have paired parentheses highlighted.
Support string escapes	Select this check box, to have string escapes highlighted.
Ignore case	Select this check box to have PhpStorm ignore case when processing file type extensions.
Keywords	Use this area to flexibly configure highlighting of keywords by grouping them into sets and associating each set with its own highlighting scheme . The area consists of 4 tabs. In each tab, create a set of keywords using the Add and Remove buttons. To associate a keyword set 1-4 with a highlighting scheme, edit the corresponding <code>Keyword1 - Keyword4</code> property on the Custom page of the Colors and Fonts dialog box.
Add	Click this button to open the Add a new keyword dialog box and define the new keyword there.
Remove	Click this button to delete the selected keyword from the list.

See Also

Procedures:

- [Creating and Registering File Types](#)

Reference:

- [File Types](#)
- [Register New File Type Association Dialog](#)
- [Editor. Colors and Fonts](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

General

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

File | Settings | IDE Settings | General

Ctrl+Alt+S Meta Comma



PhpStorm | Preferences | IDE Settings | General

Ctrl+Alt+S Meta Comma



Use this dialog box to configure general behavior of PhpStorm.

Item	Description
Startup/Shutdown	In this area, configure behavior of PhpStorm on launch and exit.
Reopen last project on startup	Select this check box to have PhpStorm re-open the last opened project on startup.
Confirm application exit	Select this check box to have a warning message displayed when you attempt to close PhpStorm.
Confirm window to open project in	Select this check box to have PhpStorm ask you whether you want to open a new project in the same frame, or in a new one. Refer to the section Opening Multiple Projects . If this check box is not selected, PhpStorm will silently close the currently opened project, and then open a new one in the same window
Synchronization	In this area, configure behavior of PhpStorm on moving the focus to another application.
Synchronize files on frame activation	Select this check box to have all externally changed files detected and reloaded from disk when you switch to PhpStorm from another application.
Save files on frame deactivation	Select this check box to have all modified files saved every time you switch to another application. Clearing the check box may cause a conflict of changes within and outside PhpStorm when it loses the focus. In this case, PhpStorm will prompt you to select the desired revision. See the File Cache Conflict dialog box reference for details.
Save files automatically if application is idle for <i>number</i> sec	Select this check box to have PhpStorm save any modified file every time it is idle for a certain period of time. In the text box, specify the time period in seconds.
Use "safe write" (save changes to a temporary file first)	If this check box is selected, a changed file will be first saved to a temporary file; if the save operation is completed successfully, the original file is deleted, and the temporary file is renamed.

See Also

Procedures:

- [Configuring IDE Settings](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

HTTP Proxy

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

File | Settings | IDE Settings | HTTP Proxy

Ctrl+Alt+S Meta Comma



PhpStorm | Preferences | IDE Settings | HTTP Proxy

Ctrl+Alt+S Meta Comma



In this dialog box, customize the HTTP proxy settings that PhpStorm will use when connecting to the Internet to search for [updates](#) or submit your feedback.

Item	Description
Use HTTP Proxy	When this check box is selected, PhpStorm attempts to connect to the Internet using the specified proxy server. When the check box is cleared, PhpStorm connects to the Internet directly.
Host name	In this text box, specify the name or IP address of the proxy server that will be used to connect to the Internet.
Port number	In this text box, specify the port number the proxy server listens to.
Proxy authentication	Select this check box to enable proxy server authentication.
Login	In this text box, type the user name or login to authenticate at the specified proxy server.
Password	In this text box, type the password that matches the specified login or user name.
Remember password	Select this check box to have PhpStorm save the specified password. Otherwise you will be prompted to type it every time PhpStorm connects to the Internet.

See Also

Procedures:

- [Configuring IDE Settings](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Images

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

File | Settings | IDE Settings | Images

Ctrl+Alt+S Meta Comma



PhpStorm | Preferences | IDE Settings | Images

Ctrl+Alt+S Meta Comma



Use this dialog box to configure how you want images to be displayed in PhpStorm or in an external editor.

Item	Description
Editor	Use this area to specify the settings according to which images should be displayed in PhpStorm.
Show Grid lines by default	Select this check box to have a grid displayed when reviewing image files. Tip When the check box is selected, the area below is enabled where you can define how to display a grid and its elements.
Show Grid lines only when zoom factor equal or more than	Use this spin box to specify the minimum zoom factor to have a grid displayed.
Show Grid line after every (pixels)	Use this spin box to specify the number of pixels between a pair of grid lines.

Grid line color	From this palette, click the colour to display grid lines.
Show transparency chessboard by default	Select this check box to have transparent pieces of images shown as a chessboard.
Chessboard cell size	Use this spin box to specify the chessboard cell size.
Color of 'white' cell	From this palette, click the colour to display chessboard cells located at initially 'white' positions.
Color of 'black' cell	From this palette, click the colour to display chessboard cells located at initially 'black' positions.
Use mouse wheel for image zooming	Select this check box to enable zooming the image through the Ctrl+Mouse wheel combination.
Enable smart zooming for small images	Select this check box to have small images opened with the zooming factor to the size specified below.
Preferred minimum width/height for smart zooming (pixels)	Use this spin box to set the minimum number of pixels to zoom the small images to.
External Editor	In this area, specify the external tool to display and edit images in.
Executable path	In this text box, specify the path to the executable file of the software to be used for editing image files.

See Also

Procedures:

- [Viewing Images](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Intentions

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

File | Settings | IDE Settings | Intentions

Ctrl+Alt+S Meta Comma



PhpStorm | Preferences | IDE Settings | Intentions

Ctrl+Alt+S Meta Comma



Use this dialog box to enable and disable intention actions.

In this topic:

- [Intention List](#)
- [Toolbar and Controls](#)

Intention list

The list shows all intention actions currently available in PhpStorm. The intention actions are grouped according to the areas of their use.

To enable an intention action, select the check box to its left.

Note

The shown intention actions are either hardcoded or related to the *Intention PowerPack* plugin, which is a third-party plugin shipped with PhpStorm.

Toolbar and controls

Item	Tooltip and shortcut	Description
	Ctrl+Add Command Add or Ctrl+Equals Command Equals	Expand all nodes in the intention list.
	Ctrl+Subtract Command Subtract or Ctrl+Minus Command Minus	Collapse all nodes in the intention list.
		Use this text box to search through the list of intention actions. As you type a search string, the intention actions that match the search pattern are displayed. To finalize the search, press Enter . The previously used search patterns are stored in the search history list.
		Click this button to clear the search history list.
Description		This read-only field shows the description of the selected intention action.
Usage examples		This area illustrates the effect of applying the selected intention action through the following fields: <ul style="list-style-type: none"> • Before - this read-only field shows an example of source code before applying the selected intention action. • After - this read-only field shows the result of applying the selected intention action to the above example of source code.

See Also

Concepts:

- [Intention Actions](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Keymap

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

[File](#) | [Settings](#) | [IDE Settings](#) | [Keymap](#)

Ctrl+Alt+S Meta Comma



[PhpStorm](#) | [Preferences](#) | [IDE Settings](#) | [Keymap](#)

Ctrl+Alt+S Meta Comma



Use this dialog box to create, edit, and remove custom keymaps for specific environments.

Note

Default keymaps are protected against editing. To re-configure shortcut associations, create a child keymap based on the desired default one and edit it as required.

Item and shortcut

Description

Keymaps	From this drop-down list, select the desired keymap.
Copy	Click this button to create a child keymap on the basis of the keymap selected in the Keymaps drop-down list.
Reset	Click this button to abandon all the changes made to a custom keymap and restore the configuration of the parent keymap.
Delete	Click this button to remove the selected custom keymap from the list.
Based on keymap	This read-only field shows the name of the parent keymap.
	Click this button to expand all nodes in the All Actions list box.
Ctrl+Add Command Add or Ctrl+Equals Command Equals	
	Click this button to collapse all nodes in the All Actions list box.
Ctrl+Subtract Command Subtract or Ctrl+Minus Command Minus	
	Use this text box to search through the All Actions list. As you type a search string, the actions that match the search pattern are displayed. To finalize the search, press Enter . The previously used search patterns are stored in the search history list.
	Click this button to clear the current search pattern from the text box.
	Click this button to open the Filter Settings dialog box for filtering out the desired actions by keystrokes. The actions with shortcuts that match the specified criteria are shown in the All Actions list box.
	Click this button to restore the initial contents of the All Actions list box.
All Actions	The list box shows all actions currently available in PhpStorm. The actions are grouped below nodes according to the areas of their use. <ul style="list-style-type: none"> • For custom keymaps, use the controls of the dialog box to associate keyboard and mouse shortcuts with the desired actions. • For default keymaps, the controls are disabled and you can only view which shortcuts are associated with an action.
Action Description	This read-only field shows the description of the selected action.
Shortcuts	This read-only field shows the list of shortcuts associated with the selected action in the current keymap. Icons to the left of a shortcut denote its type: <ul style="list-style-type: none"> • - keyboard shortcut • - mouse shortcut
Add Keyboard Shortcut	Click this button to open the Enter Keyboard Shortcut dialog box, where you can specify the combination of keystrokes to be assigned to the selected action in the current keymap.
Add Mouse Shortcut	Click this button to open the Enter Mouse Shortcut dialog box, where you can specify the combination of mouse clicks and buttons to be assigned to the selected action in the current keymap.
Remove	Click this button to delete the selected shortcut from the Shortcuts list.

See Also

Reference:

- [Enter Keyboard Shortcut Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Enter Keyboard Shortcut Dialog

Previous | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

[File](#) | [Settings](#) | [IDE Settings](#) | [Keymap](#)

Ctrl+Alt+S Meta Comma



[PhpStorm](#) | [Preferences](#) | [IDE Settings](#) | [Keymap](#)

Ctrl+Alt+S Meta Comma



The dialog box opens when you select an action and click the **Add Keyboard Shortcut** button. Use this dialog box to bind the selected action with a new keyboard shortcut, which may consist of one or two keystrokes. The resulting keyboard shortcut is marked with the  icon in the **Shortcuts** list.

Warning

Use your mouse pointer to click buttons in the dialog box. Any key stroke is interpreted as a shortcut!

Item	Description
First Stroke	Use this text box to define the primary shortcut by pressing keyboard keys and key combinations.
Enable	Select this check box to allow an optional second shortcut.
Second Stroke	Use this text box to define an optional shortcut by pressing keyboard keys and key combinations.
<p>Note</p> <p>This field is available after the Enable check box is selected.</p>	
Shortcut Preview	In this read-only field, view the newly defined shortcuts.
Conflicts	This read-only field shows messages about conflicts that arise if a suggested keystroke is already in use.
<p>Warning</p> <p>You can ignore a conflict and assign a shortcut to several actions. However it is strongly recommended that you avoid binding two actions with the same shortcut, because the priority of these actions is not defined.</p>	

See Also

Procedures:

- [Configuring Keyboard Shortcuts](#)

Reference:

- [Keymap](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Enter Mouse Shortcut Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

[File](#) | [Settings](#) | [IDE Settings](#) | [Keymap](#)

Ctrl+Alt+S Meta Comma



[PhpStorm](#) | [Preferences](#) | [IDE Settings](#) | [Keymap](#)

Ctrl+Alt+S Meta Comma



The dialog box opens when you select an action and click the **Add Mouse Shortcut** button. Use this dialog box to bind the selected action with a new mouse shortcut, which may be a single or a double clicking one of the mouse buttons or the wheel button.

The resulting mouse shortcut is marked with the  icon in the **Shortcuts** list.

Item	Description
------	-------------

Click Count In this area, specify the type of mouse click to be assigned to the selected action. The available options are:

- **Single Click**
- **Double Click**

Click Pad Click the desired mouse button anywhere in this area.

Tip

The number of clicks in this area does not affect the shortcut configuration. No matter how many times you click a button, the click type chosen in the **Click Count** area will be assigned.

Shortcut Preview This read-only field shows the newly defined shortcut.

Conflicts This read-only field shows messages about conflicts that arise if a suggested mouse shortcut is already in use.

Warning

You can ignore a conflict and assign a shortcut to several actions. However it is strongly recommended that you avoid binding two actions with the same shortcut, because the priority of these actions is not defined.

See Also

Reference:

- [Keymap](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Live Templates

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

[File](#) | [Settings](#) | [IDE Settings](#) | [Live Templates](#)

Ctrl+Alt+S Meta Comma



[PhpStorm](#) | [Preferences](#) | [IDE Settings](#) | [Live Templates](#)

Ctrl+Alt+S Meta Comma



Use this page to [create, manage, and edit live templates](#).

In this topic:

- [List of Available Live Templates](#)
- [Template Text Area](#)

List of available live templates

Item	Tooltip and shortcut	Description
By default expand with		Use this drop-down list to specify the default invocation key for all templates. Individual invocation keys for particular templates are defined in the Edit Template dialog box.
Live Templates		The list shows all currently available template abbreviations supplied with their descriptions. The abbreviations are grouped below nodes and sorted alphabetically within each group. To activate a template or an entire group, select the check box next to it.
		Note
		Note that: <ul style="list-style-type: none"> • Only active templates are displayed upon pressing <code>Ctrl+J</code>/<code>Command J</code> in the editor. • If a template is active, the editor is sensitive to its abbreviation. Otherwise, the abbreviation is considered merely a set of characters.
	Add Alt+Insert Command N	Click this button to have a new template item added to the current node and define the template from scratch in the Template Text area .
	Remove Alt+Delete Meta Delete	Click this button to have the selected live template removed from the list.
	Copy	Click this button to create a new template based on the selected template. A new template item is added to the current node and the fields in the Template Text area show the definition of the selected template.

Template text area

The focus is moved to this area in the following cases:

- When you click the **Add**  or **Copy**  button.
- When you select a live template in the list.
- When you select a fragment of code in the editor and choose [Tools | Save as Live Template](#).

Use controls of this area to create new [live templates](#) and edit settings of the existing ones.

Tip

You can navigate through the Template Text Area using the hot keys that are marked in the field labels.

Item	Description
Abbreviation	In this text box, specify the template abbreviation .
Description	In this text box, provide optional description of a template or an example of its usage.
Template Text	<p>In this text box, type the template body that may contain plain text and variables in the format <code><variable name>\$</code>.</p> <ul style="list-style-type: none"> • If you need a dollar sign (\$) in the template text, escape it by duplicating this character (\$\$). • To change variables in a template, click the Edit Variables button. <p>PhpStorm supports two predefined live template variables: <code>\$END\$</code> and <code>\$SELECTION\$</code>.</p> <ul style="list-style-type: none"> ◦ <code>\$END\$</code> indicates the position of the cursor after the template is expanded. For example, the template <code>return \$END\$</code> will be expanded into <code>return ;</code> with the cursor positioned <i>right before</i> the semicolon. ◦ <code>\$SELECTION\$</code> is used in <i>surround templates</i> and stands for the code fragment to be wrapped. After the template is expanded, the selected text is wrapped as specified in the template. <p>For example, if you select <code>EXAMPLE</code> in your code and invoke the <code>"\$SELECTION\$"</code> template via the assigned abbreviation or by pressing <code>Ctrl+Alt+T</code> (or <code>Command Alt T</code>) and selecting the desired template from the list, PhpStorm will wrap the selection in double quotes as follows: <code>"EXAMPLE"</code>.</p> <p>Note</p> <p>You cannot edit the predefined variables <code>\$END\$</code> and <code>\$SELECTION\$</code>.</p>
Applicable in:	This read-only field shows the languages and/or pieces of code where the editor should be sensitive to the template. Upon pressing <code>Ctrl+J</code> (or <code>Command J</code>) in such context, PhpStorm displays a list of templates that are valid for this context.
Change	Click this link to modify the set of contexts where the current template is enabled. Upon clicking the link, displays a list of supported language contexts is displayed. To make PhpStorm consider a context sensitive to the template, select a check box next to the context name.
Edit Variables	Click this button to open the Edit Template Variables dialog box, where you can define how PhpStorm should process template variables upon template expansion.
	<p>Note</p> <p>The Edit Variables button is enabled only if the template body contains at least one user-defined variable, that is, a variable different from <code>\$END\$</code> or <code>\$SELECTION\$</code>.</p>
Options	<p>In this area, define the behavior of the editor when a template is expanded.</p> <ul style="list-style-type: none"> • Expand with - from this drop-down list, choose the key to invoke the template. • Reformat according to style - select this check box to have PhpStorm automatically reformat the expanded text according to the current style settings, defined in the Code Style dialog box.

See Also

Concepts:

- [Live Templates](#)

Procedures:

- [Creating and Editing Live Templates](#)
- [Zen Coding Support](#)

Reference:

- [Edit Template Variables Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Edit Template Variables Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts:  

File | Settings | IDE Settings | Live Templates - Edit Variables

Ctrl+Alt+S Meta Comma



PhpStorm | Preferences | IDE Settings | Live Templates - Edit Variables

Ctrl+Alt+S Meta Comma



The dialog box opens when you click the Edit Variables button in the [Template Text](#) area on the [Live Templates](#) page of the Settings dialog box.

Note

The Edit Variables button is enabled only if the template body contains at least one user-defined variable, that is, a variable different from \$END\$ or \$SELECTION\$.

Use this dialog box to create and edit expressions within the selected [live template](#).

Controls

Item	Description
Name	In this text box, view or edit the variable name in the format \$<variable_name>\$.
Expression	In this text box, specify the expression to have the value of the corresponding template input field calculated automatically. This expression may contain constructs of the following basic types: <ul style="list-style-type: none"> • Predefined functions with possible arguments. • String constants in double quotes. • The name of another variable defined in a live template. Type an expression manually or select a predefined function from the drop-down list. The list shows also the number and type of parameters, if any, for the selected function. The available functions are listed alphabetically in the Functions table.
Default value	In this text box, specify the default string to be entered in the corresponding input field of the expanded template, if the expression does not give any result after calculation
Skip if defined	Select this check box to have PhpStorm proceed with the next input field, if the value of the current input field is defined.
Move Up / Move Down	Use these buttons to change the order of variables in the list. The order of variables in the table determines the order in which PhpStorm will switch between the corresponding input fields when the template is expanded.

Functions

Item	Description
capitalize(<name>)	Capitalizes the first letter of the name passed as a parameter.
className()	Returns the name of the current class (the class where the template is expanded).
complete()	This expression substitutes for the Code Completion invocation at the variable position.
completeSmart()	This expression substitutes for the Smart Type Completion invocation at the variable position.
date()	Returns the current system date.
decapitalize(<name>)	Replaces the first letter of the name passed as a parameter with the corresponding lowercase letter.
firstWord(String)	Returns the first word of the string passed as a parameter.
lineNumber()	Returns the current line number.
time()	Returns the current system time.
underscoresToSpaces (VAR)	Returns the string passed as a parameter with spaces substituting for underscores.
user()	Returns the name of the current user.
Item	Description
jsClassName()	Returns the name of the current JavaScript class.
jsMethodName()	Returns the name of the current JavaScript method.

See Also

- Concepts:
- [Live Templates](#)
 - [Live Template Variables](#)

- Procedures:
- [Creating and Editing Live Templates](#)

- Reference:
- [Live Templates](#)

- Web Resources:
- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>

Menus and Toolbars

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

[File](#) | [Settings](#) | [IDE Settings](#) | [Menus and Toolbars](#)

Ctrl+Alt+S Meta Comma



[PhpStorm](#) | [Preferences](#) | [IDE Settings](#) | [Menus and Toolbars](#)

Ctrl+Alt+S Meta Comma



Use this dialog box to configure the scheme of [menus and toolbars](#) in your IDE.

In this topic:

- [Menus and Items List](#)
- [Controls](#)

Menus and items list

The list shows all the items that are currently available from the PhpStorm menus and bars. The items are grouped below nodes according to the areas of their use.

To configure an item, expand the relevant node and select the desired item. After that, the buttons in the dialog box are available.

Note

The top level of the tree that defines the set of menus and bars cannot be changed.

Controls

Item	Description
Add After	Click this button to add a new action to the menu after the selected one. In the Choose Actions to Add dialog box that opens choose the desired action and optionally assign an icon to it.
Add Separator	Click this button to have a separator added to the menu after the selected item.
Edit Action Icon	Click this button to associate an icon with the selected menu item. In the Choose Action Icon Path dialog box that opens specify the path to the desired image.
Note	
Only *.png files can be used as icons.	
Remove	Click this button to delete the selected item from the list.
Move Up	Click this button to move the selected item one position up.
Move Down	Click this button to move the selected item one position down.
Restore Default	Click this button to abandon all the changes made to the selected item and return to the default settings.

See Also

Procedures:

- [Configuring IDE Settings](#)

Reference:

- [Choose Actions to Add Dialog](#)
- [Editor. Colors and Fonts](#)
- [Keymap](#)
- [Code Style. HTML](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Choose Actions to Add Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

[File](#) | [Settings](#) | [IDE Settings](#) | [Menus and Toolbars](#)

Ctrl+Alt+S Meta Comma



[PhpStorm](#) | [Preferences](#) | [IDE Settings](#) | [Menus and Toolbars](#)

Ctrl+Alt+S Meta Comma



The dialog box opens when you select an item in the [Menus and Items List](#) and click the **Add After** button.

In the dialog box, choose the desired action to be added to the menu or toolbar and optionally assign an icon to it.

Item	Description
Action List	The list shows all the actions available in PhpStorm. The actions are grouped below nodes according to the areas of their use.
Icon Path	In this text box, specify the location of the file with the icon you want to assign to the selected action. If necessary, use the Browse button  to open the Select Path dialog box.
Tip	The image file should have .png extension.
Set Icon	Click this button to associate the selected action with the icon specified in the Icon Path dialog box.

See Also

Procedures:

- [Configuring IDE Settings](#)

Reference:

- [Menus and Toolbars](#)
- [Editor. Colors and Fonts](#)
- [Keymap](#)
- [Code Style. HTML](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Notifications

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

[File](#) | [Settings](#) | [IDE Settings](#) | [Notifications](#)

Ctrl+Alt+S Meta Comma



[PhpStorm](#) | [Preferences](#) | [IDE Settings](#) | [Notifications](#)

Ctrl+Alt+S Meta Comma



Use this dialog to enable and disable notifications about certain events, change their presentation, and optionally enable their logging.

Item	Description
Display balloon notifications	Select this check box to enable event notifications. (The notifications, generally, are shown in the balloons that appear on the screen when the corresponding events take place.)
Group	This column lists groups of events that you may be notified of and/or that may be logged.
Display	If the Display balloon notification check box is selected, the settings in this column specify how the notifications for the corresponding group of events are shown. The available display options are: <ul style="list-style-type: none"> • Balloon. The balloons with the notification messages appear on the screen for a short period of time and then disappear automatically. The notifications are also shown in the Status bar, and added to the list of notifications. • Sticky balloon. The notification balloons stay on the screen unless you close them. • Tool window balloon. The notification balloons are shown only if an appropriate tool window is open. • No pop-up. The notifications for the corresponding group of events are not shown.
Log	If the check box for a group of events is selected, the corresponding events are logged and can be seen in the Event Log tool window .

See Also

Reference:

- [IDE Settings](#)
- [Event Log](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Plugins

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

[File](#) | [Settings](#) | [IDE Settings](#) | [Plugins](#)

Ctrl+Alt+S Meta Comma



PhpStorm | Preferences | IDE Settings | Plugins

Ctrl+Alt+S Meta Comma



The dialog box shows a list of all currently installed plugins. For each plugin, the dialog box indicates whether it is bundled with PhpStorm or has been downloaded from a repository.

Use this dialog box to enable and disable plugins bundled with PhpStorm, get updates, enable and disable installed plugins.

In this section:

- [Common controls](#)
- [Plugin Information pane](#)
- [Legend of Plugin Status](#)

Item	Tooltip and shortcut	Description
	Reload list of plugins	Click this button to have PhpStorm refresh the list of plugins and make their statuses up-to-date.
	Update Plugin	Click this button to have the selected plugin updated to the newest version. This command is also available on the context menu of the selected plugin.
	Uninstall	Click this button to uninstall the selected plugin. This command is also available on the context menu of the selected plugin.
Show		In this drop-down list, specify the status of plugins to be displayed. The available options are: <ul style="list-style-type: none"> • All plugins • Enabled plugins • Disabled plugins PhpStorm refreshes the contents of the list depending on your choice.
Search		Use this text box to search through the list of plugins. As you type a search string, the plugins that match the search pattern are displayed. To finalize the search, press Enter . The search pattern is added to the search history list.
		Click this button to have the history of the search patterns displayed.
		Click this button to clear the history of the search strings.

Enable/Disable		<ul style="list-style-type: none"> • Select this check box to activate the plugin next to it. • Clear this check to disable the plugin next to it. If you attempt to disable a plugin that is required for other plugins, the dependent plugins are highlighted red.
----------------	--	--

Plugin Information Pane		<ul style="list-style-type: none"> • Description: This read-only field briefly describes the functionality and the purposes of the plugin selected in the list. • Change notes: This read-only field shows the history of changes made to the plugin.. • Plugin home page: This read-only field shows the link to the plugin home page, if specified. • Version: This read-only field shows which version of the selected plugin is currently installed. • Vendor: This read-only field shows the vendor's name, e-mail, and home page, if specified.
-------------------------	--	---

Tip

If you cannot see the **Browse repositories** and **Install plugin from disk**, use the slider that appears to the right of the **Plugin information pane**.

Browse repositories	Click this button to open the Browse Repositories dialog box to monitor the contents of repositories, look for appropriate plugins, download, and install them, if found.
Install plugin from disk	Click this button to install your own plugin. From the Choose Plugin File dialog box, that opens, select a file which implements the required plugin.

Legend of plugin status

Color	Plugin Status
Black	The plugin is installed or available.
Red	The plugin is incompatible with the installed version of PhpStorm or a required plugin is disabled.
Blue	A newer version of the plugin is available.
Green	The plugin has been downloaded and installed, but has not been activated yet. After PhpStorm restarts, the plugin will appear in the Installed tab. The plugin in the Available tab will be highlighted black.
Gray	The plugin has been uninstalled, but the changes have not taken effect yet. After PhpStorm restarts the plugin will be removed from the list in the Installed tab.

See Also

- Procedures:
- [Managing Plugins](#)
- Web Resources:
- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>

Browse Repositories Dialog

Previous | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

[File](#) | [Settings](#) | [IDE Settings](#) | [Plugins - Browse Repositories](#)

Ctrl+Alt+S Meta Comma



[PhpStorm](#) | [Preferences](#) | [IDE Settings](#) | [Plugins - Browse Repositories](#)

Ctrl+Alt+S Meta Comma



The dialog box opens when you click the **Browse Repositories** button in the [Plugins](#) page of the **Settings** dialog box.

The dialog box shows all the plugins, both installed and not installed, that are available from the [selected repository](#). Use the dialog box to monitor, download, and install plugins on your computer.

By default, PhpStorm provides access to the [JetBrains Plugin Repository](#). To monitor and retrieve plugins from [custom \(enterprise\) repositories](#), click the **Manage repositories** button and configure a list of URL addresses to access the relevant repositories in the [Custom Plugin Repositories](#) dialog box.

Item	Tooltip and shortcut	Description
	Reload list of plugins	Click this button to have PhpStorm refresh the list of plugins available from the repository selected in the Repository drop-down list.
	Download and Install	Click this button to have the selected plugin downloaded from the repository and installed. This command is also available on the context menu of the selected plugin.
		<p>Note</p> <p>The plugin will be activated after you restart PhpStorm.</p>
Repository		<p>From this drop-down list, select the enterprise repository to look for plugins in. The contents of the list are composed in the Custom Plugin Repositories dialog box.</p> <p>Note</p> <p>The drop-down is available only if you have configured access to any enterprise repositories.</p>
Category		In this drop-down list, restrict the search for plugins to a specific area of their usage, for example, Build, Framework Integration , etc. Only the plugins for which the selected area is specified will remain in the list.
Sort by		<p>In this drop-down list, choose the sorting order in which you want plugins displayed. The available options are:</p> <ul style="list-style-type: none"> Name: Choose this option to have PhpStorm show plugins sorted alphabetically by their names. Downloads: Choose this option to have PhpStorm show plugins sorted by the number of their downloads in the descending order, that is, most often downloaded plugins at the top of the list. Updated: Choose this option to have most recently updated plugins shown at the top of the list. Status: Choose this option to have installed plugins shown first. Repository: Choose this option to have PhpStorm show plugins sorted alphabetically by the names of the repositories they are available from.
Search		Use this text box to search through the list of plugins. As you type a search string, the plugins that match the search pattern are displayed. To finalize the search, press Enter . The search pattern is added to the search history list.
		Click this button to have the history of the search patterns displayed.
		Click this button to clear the history of the search strings.
List of plugins		<p>This area shows the plugins from the selected or all repositories sorted as specified in the Sort by drop-down list. For each plugin, the following information is provided:</p> <ul style="list-style-type: none"> Name: This read-only field shows the name to identify the plugin. Downloads: This read-only field shows how many times the plugin has been downloaded from the repository. Date: This read-only field shows the date when the plugin version was uploaded to the repository. Category: This read-only field shows the area where the plugin is applicable. Categories are used in search for plugins.
HTTP Proxy Settings		Click this button to customize the HTTP proxy settings that PhpStorm will use when connecting the repositories in the HTTP Proxy Settings dialog box, that opens.
Manage repositories		Click this button to configure a list of available enterprise plugin repositories in the Custom Plugin Repositories dialog box, that opens.

HTTP proxy settings dialog box

The dialog box opens when you click the **HTTP Proxy Settings** button.

Item	Description
Use HTTP proxy	Select this check box to have plugins updated through HTTP proxy.
Host name	In this text box, specify the host name to be used.
Port number	In this text box, specify the port number to listen to. By default, PhpStorm suggests 80.
Proxy Authentication	Select this check box to enable authentication to work with proxy.
Login	In this text box, specify the proxy login to use.
Password	In this text box, specify the proxy password to use.
Remember password	Select this check box to have PhpStorm store your proxy password.

See Also

Procedures:

- [Managing Plugins](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Custom Plugin Repositories

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

File | Settings | IDE Settings | Plugins - Browse Repositories - Manage Repositories

Ctrl+Alt+S Meta Comma



PhpStorm | Preferences | IDE Settings | Plugins - Browse Repositories - Manage Repositories

Ctrl+Alt+S Meta Comma



The dialog box opens when you click the Manage Repositories button in the [Browse Repositories](#) dialog box. In this dialog box, configure a list of [custom \(enterprise\) repositories](#), in addition to the default [JetBrains Plugin Repository](#).

Item	Description
Repositories	This read-only list shows all the custom repositories accessible from PhpStorm.
	Note
	The default JetBrains Plugin Repository is not shown.
Add	Click this button to specify the URL address to access your custom repository in the Add New Plugin Host dialog box, that opens.
Edit	Click this button to update the URL address of the selected repository in the Edit Plugin Host dialog box, that opens.
Remove	Click this button to remove the selected URL address from the list and thus disable access to the corresponding repository.
Apply	Click this button to save the configured list of repositories without leaving the dialog box.

Add new plugin host dialog box

The dialog box opens when you click the Add or Edit button in the **Custom Plugin Repositories** dialog box. In this dialog box, specify or update the URL addresses of enterprise repositories to use plugins from.

Item	Description
Plugin Host URL	In this text box, type the URL address of the required repository.
Check Now	Click this button to make sure that the specified URL address guarantees successful connection to the repository.

See Also

Procedures:

- [Managing Enterprise Repositories](#)
- [Managing Plugins](#)

Reference:

- [Browse Repositories Dialog](#)
- [Plugins](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Passwords

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

File | Settings | IDE Settings | Passwords

Ctrl+Alt+S Meta Comma



PhpStorm | Preferences | IDE Settings | Passwords

Ctrl+Alt+S Meta Comma



In this dialog box, configure how you want PhpStorm to store passwords for accessing remote Git repositories. These settings are valid at the PhpStorm level, that is, they apply to passwords for all remote Git repositories you use from PhpStorm.

Item	Description
Do not remember passwords	When this option is selected, PhpStorm does not store passwords and asks you to enter the password on every attempt to access a remote Git repository.
Remember passwords until closing of the application	When this option is selected, PhpStorm stores a password only during one session. Upon the session end, the password is cleared from the memory.
Remember on disk (protected with master password)	Select this option to have PhpStorm save all passwords to remote repositories in a <code>passwords</code> database protected by single <code>master password</code> . When you attempt to access a remote Git repository, PhpStorm does not request on the corresponding password, but asks you to specify the <code>master password</code> and retrieves the repository password automatically.
Reset Master Password	Click this button to open the Master Password / Reset Master Password dialog box and specify a <code>master password</code> for the first time or reset the existing <code>master password</code> . Warning If you reset a <code>master password</code> , all the previously saved <code>master passwords</code> are cleared.
Change Master Password	Click this button to change the existing <code>master password</code> in the Change Master Password dialog box. Note PhpStorm preserves the history of updates to the <code>master password</code> , until you reset it.

See Also

Procedures:

- [Handling Passwords for Git Remote Repositories](#)

Reference:

- [Git Reference](#)
- [IDE Settings](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Master Password / Reset Master Password Dialog

Previous | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

File | Settings | IDE Settings | Passwords - Reset Master Password

Ctrl+Alt+S Meta Comma



PhpStorm | Preferences | IDE Settings | Passwords - Reset Master Password

Ctrl+Alt+S Meta Comma



The dialog box opens when you click the **Reset Master Password** button on the [Passwords](#) page of the Settings dialog box, or the **Reset Password** button in the [Change Master Password](#) dialog box.

In this dialog box, specify or reset the `master password` for accessing the database where passwords for remote repositories are stored.

Note

When you reset the `master password`, PhpStorm clears the history of all the previously existed `master passwords`.

Item	Description
New Password	In this text box, type the <code>master password</code> to use.
Confirm New Password	In this text box, repeat the <code>master password</code> to use.
Encrypt the master password with user credentials	When this check box is selected, PhpStorm encrypts the specified <code>master password</code> with your system login and password and does not request you to type the <code>master password</code> in the future. Warning Selecting this check box increases security risks. Note This functionality is only available in the Windows operating system.
Set Password	Click this button to have PhpStorm save and use the specified <code>master password</code> .

Note

The button is available in the **Master Password** dialog box, which opens when you set the `master password` for the first time.

Reset Password

Click this button to have PhpStorm save and use the specified `master password` and clear the history of previous `master passwords`.

Note

The button is available in the **Reset Master Password** dialog box.

See Also

Procedures:

- [Handling Passwords for Git Remote Repositories](#)

Reference:

- [Git Reference](#)
- [IDE Settings](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Change Master Password Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

File | Settings | IDE Settings | Passwords - Change Master Password

Ctrl+Alt+S Meta Comma



PhpStorm | Preferences | IDE Settings | Passwords - Change Master Password

Ctrl+Alt+S Meta Comma



The dialog box opens when you click the **Change Master Password** button on the [Passwords](#) page of the **Settings** dialog box. In this dialog box, update the `master password` for accessing the database where passwords for remote repositories are stored.

Note

PhpStorm saves the history of updates of the `master password`, until you reset it.

Item	Description
Old Password	In this text box, type the current <code>master password</code> .
New Password	In this text box, type the <code>master password</code> to use.
Confirm New Password	In this text box, repeat the <code>master password</code> to use.
Encrypt the master password with user credentials	When this check box is selected, PhpStorm encrypts the specified <code>master password</code> with your system login and password and does not request you to type the <code>master password</code> in the future.

Warning

Selecting this check box increases security risks.

Note

This functionality is only available in the Windows operating system.

Change Password

Click this button to have PhpStorm use the specified `master password` and add the previous password to the history.

Reset Password

Click this button to open the [Reset Master Password](#) dialog box, where you can set a new `master password` and clear the history of previous `master passwords`.

See Also

Procedures:

- [Handling Passwords for Git Remote Repositories](#)

Reference:

- [Git Reference](#)

- [IDE Settings](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Quick Lists

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

[File](#) | [Settings](#) | [IDE Settings](#) | [Quick Lists](#)

Ctrl+Alt+S Meta Comma



[PhpStorm](#) | [Preferences](#) | [IDE Settings](#) | [Quick Lists](#)

Ctrl+Alt+S Meta Comma



Use this dialog to configure quick lists. A **Quick List** is a pop-up menu of PhpStorm commands, configured by the user and associated with a keyboard or mouse shortcut. You can create as many quick lists, as necessary. Each command, included in a quick list, is identified by a sequential number. Numbering starts from the numerals (0 to 9), and then proceeds with the letters in alphabetical order.

Item	Shortcut	Description
	InsertInsert	Create a new Quick List.
	DeleteDelete	Delete the selected Quick List.
Display name		Edit the name of the selected Quick List.
Description		Edit optional description of the selected Quick List.
All Actions		View the list of available actions, from which you can select items for a Quick List.
->		Add selected actions to the Quick List.
Add Separator		Inserts a separator between the items in the Quick List. The separators help you organize menu commands into logical groups.
<-		Remove selected actions from the Quick List.
Move Up	Alt+U Alt U	Shift selected action one step up.
Move Down	Alt+D Alt D	Shift selected action one step down.

See Also

Concepts:

- [Project and IDE Settings](#)

Procedures:

- [Configuring Quick Lists](#)
- [Configuring Keyboard Shortcuts](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

TODO

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

[File](#) | [Settings](#) | [IDE Settings](#) | [TODO](#)

Ctrl+Alt+S Meta Comma



[PhpStorm](#) | [Preferences](#) | [IDE Settings](#) | [TODO](#)

Ctrl+Alt+S Meta Comma



In this dialog box, configure filters to maintain your lists of TODO items in the [TODO](#) tool window and define TODO patterns to be used in filters.

Patterns

In this area, create and manage the list of available TODO patterns.

Item	Description
Icon	This read-only field displays the icon that is assigned to the current pattern. This icon is used by the entry in the TODO tool window.
Case sensitive	This read-only check box indicates whether the current pattern is case sensitive or not.
	Note

The status is changed in the [Edit Pattern](#) dialog box.

Pattern	This read-only field displays the regular expression that describes the TODO pattern. PhpStorm recognizes regular expressions in the source code against the specified patterns and displays them in the TODO tool window.
Add	Click this button to open the Add Pattern dialog box, where you can create a new ToDo pattern by specifying a regular expression.
Edit	Click this button to open the Edit Pattern dialog box, where you can modify the selected pattern.
Remove	Click this button to delete the selected pattern from the list.

Filters

In this area, manage the list of available filters.

Item	Description
Name	This read-only field shows a list of filter names.
Patterns	This read-only field shows the names of patterns included in the current filter. A filter can contain several patterns.
Add	Click this button to open the Add Filter dialog box, where you can define a new filter.
Edit	Click this button to open the Edit Filter dialog box and edit the settings of the selected filter.
Remove	Click this button to delete the selected filter.

See Also

Procedures:

- [Configuring IDE Settings](#)
- [Using TODO Lists](#)

Reference:

- [TODO Tool Window](#)
- [Add \ Edit Pattern Dialog](#)
- [Add \ Edit Filter Dialog](#)
- [Regular Expression Syntax Reference](#)
- [Output Filters Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Add \ Edit Pattern Dialog

Previous | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

File | Settings | IDE Settings | TODO - Add/Edit Pattern

Ctrl+Alt+S Meta Comma



PhpStorm | Preferences | IDE Settings | TODO - Add/Edit Pattern

Ctrl+Alt+S Meta Comma



Use this dialog box to define patterns that help track TODO items in your source code. Make sure the TODO items in the source code are inserted inside comments that are valid for the supported file types.

Item	Description
Pattern	In this text box, type the regular expression that describes the desired TODO pattern.
Icon	From this drop-down list, choose an icon for the pattern.
Case sensitive	Select this check-box to make the pattern case-sensitive.
Font type	Select the corresponding check box to have the icon text displayed in bold or italic.
Foreground	Select this check box to enable the palette and choose the foreground color.
Background	Select this check box to enable the palette and choose the background color.
Error Stripe Mark	Select this check box to enable the palette and choose the error stripe color.
Effects	Select this check box to enable effects (underscore, strikethrough, etc.) and choose the color for them from the palette.

See Also

Procedures:

- [Configuring IDE Settings](#)
- [Using TODO Lists](#)

Reference:

- [TODO](#)
- [Add \ Edit Filter Dialog](#)
- [TODO Tool Window](#)

- [Regular Expression Syntax Reference](#)
- [Output Filters Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Add \ Edit Filter Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

File | Settings | IDE Settings | TODO - Add/Edit Filter

Ctrl+Alt+S Meta Comma



PhpStorm | Preferences | IDE Settings | TODO - Add/Edit Filter

Ctrl+Alt+S Meta Comma



Use this dialog box to define filters that help track TODO items in your source code.

Item	Description
Name	In this text box, specify the name of the filter.
Patterns	From the list of available patterns, choose the ones to be included in the filter by selecting the check boxes next to them.

See Also

Procedures:

- [Configuring IDE Settings](#)
- [Using TODO Lists](#)

Reference:

- [TODO](#)
- [Add \ Edit Pattern Dialog](#)
- [TODO Tool Window](#)
- [Regular Expression Syntax Reference](#)
- [Output Filters Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Updates

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

File | Settings | IDE Settings | Updates

Ctrl+Alt+S Meta Comma



PhpStorm | Preferences | IDE Settings | Updates

Ctrl+Alt+S Meta Comma



Use this dialog box to:

- Enable automatic update of PhpStorm and specify to which kind of release you want it updated.
- Obtain information about the current PhpStorm version and availability of a newer version.

Item	Description
Check for updates in channel	Select this check box to enable the automatic update function, and select the desired updates channel (major version, public preview or beta, EAP).
Check Now	Click this button to check for updates immediately.

Tip

For the Windows operating system, you can alternatively choose **Help | Check for Update** on the main menu.

See Also

Procedures:

- [Register PhpStorm](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

2.1+

Usage Statistics

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

File | Settings | IDE Settings | Usage Statistics

Ctrl+Alt+S Meta Comma



PhpStorm | Preferences | IDE Settings | Usage Statistics

Ctrl+Alt+S Meta Comma



Use this dialog box to share the statistics of your usage of the features and plugins with JetBrains.

Item	Description
Allow to send usages statistics to JetBrains <ul style="list-style-type: none"> • Daily • Weekly • Monthly 	By selecting this check box, you allow to send your anonymous statistics to JetBrains. Click one of the radio buttons to define how often your usage statistics will be reported to JetBrains.

See Also

Procedures:

- [Productivity Guide](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Web Browsers

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

File | Settings | IDE Settings | Web Browsers

Ctrl+Alt+S Meta Comma



PhpStorm | Preferences | IDE Settings | Web Browsers

Ctrl+Alt+S Meta Comma



Use this dialog box to specify in which Web browsers PhpStorm will [open HTML files](#) upon request and whether a browser will be launched by running its executable file or through the default system command.

Default web browser

In this section, specify the browser that will be used for rendering external resources and previewing files file with Web contents. Define also the browser's behaviour if the URL address passed to it points at an archive.

Item	Description
Use system default browser	Select this option to accept your operating system default Web browser as default for PhpStorm.
Use	Select this option to specify another Web browser as default for PhpStorm. Type the path to the executable file of the browser or click the Browse button to select the path in the Select Path to Browser dialog box.
Show confirmation before extracting files	Select this check box to have a confirmation dialog box displayed before extracting files, if an archive is referenced.
Clear extracted files	Click this button to remove all the previously extracted files from the browser's cache.

Browsers

In this section, specify which browsers will be available for previewing HTML output. The page displays all the browsers supported in PhpStorm, each browser presented in a separate section. To configure the use of a browser, use the controls in the corresponding section.

Note

You can enable as many browsers as you need.

Item	Description
Active	Select this check box to enable the use of the respective browser through the icon in the editor or through the View Web Preview command. If this check box is not selected, the corresponding browser icon will not appear in the icons toolbar or pop-up menu.

- Default** Click this button to have PhpStorm launch the respective browser using the default system command.
-  Click this button to have PhpStorm launch the respective browser using its executable file. In the **Select Path** dialog box, specify the path to the executable file of the corresponding browser.

Settings

Tip
The button is available only in the **Firefox** and **Chrome** sections.

In the **Firefox Settings** dialog box, specify the [Firefox browser profile](#) to use for previewing output:

- **Path to "profiles.ini"** - in this text box, specify the location of the `profiles.ini` file, which determines the Firefox profile to be used.
- **Profile** - from this drop-down list, select the desired predefined profile to use.

In the **Chrome Settings** dialog box, do the following:

- Select the check box **Use custom profile directory**, and in the text box below, specify the location of the `chrome-user-data` file, which determines the Chrome profile to be used.
- Select **Enable remote debug on port** check box to allow remote debugging, and specify the port number.

See Also

Procedures:

- [Previewing Pages with Web Contents in a Browser](#)

Reference:

- [Run/Debug Configuration: JavaScript Debug](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

XPath Viewer

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

[File](#) | [Settings](#) | [XPath Viewer for Windows and Linux](#)
[PhpStorm](#) | [Preferences](#) | [XPath Viewer for Mac OS](#)

In this dialog box, configure the PhpStorm behaviour during interactive execution of XPath expressions.

Item	Description
Scroll first hit into visible area	Select this check box to have the editor automatically scroll to the first XPath match.
Use node at cursor as context node	Select this check box to have the entered XPath expression use the currently selected node (tag/attribute/pi, etc.) as its context node and evaluate the expression relatively to this node.
Highlight only start tag instead of whole tag content	Do one of the following: <ul style="list-style-type: none"> • Select this check box to have only the name of a matching tag highlighted. • Clear this check box to have the entire content of a matching tag highlighted.
Add error stripe markers for each result	Select this check box to have each match supplied with an error stripe marker which can be quickly navigated to. The tooltip of each marker shows the matched content.
Show actions in Toolbar	Select this check box to have buttons that invoke XPath-related actions displayed on the Main Toolbar .
Show actions in Main Menu	When this check box is selected, XPath-related actions are available from the main menu.
Colors	In this area, configure color indication during execution of XPath expressions. <ul style="list-style-type: none"> • Highlight Color - in this area, select the color to indicate XPath matches in the editor. • Context Node Color - in this area, select the color to indicate the current context node.

See Also

External Links:

- [XPath and XSLT Support](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

XSLT

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

[File](#) | [Settings](#) | [XSLT for Windows and Linux](#)

Use this dialog box to configure [XSLT support](#) at the PhpStorm level.

Item	Description
Show Associated Files in Project View	Select this check box to have XML files associated with XSLT stylesheets displayed in the project view .
	Note
	Associated XML files are displayed below the corresponding stylesheets.

See Also

Reference:

- [XSLT File Associations](#)

External Links:

- [XSLT Support](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Keyboard Shortcuts and Mouse Reference

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Using shortcuts is a major way to maximum efficiency and productivity working with PhpStorm. This part lists keystroke combinations and their functions for the Default keymap defined in [Keymap](#) dialog box.

Warning

The key combinations documented in this part may fail to perform the function described, if you are using a [customized keymap](#).

Tip

To get a printable copy of the default keymap, choose [Help | Default Keymap Reference](#) on the main menu.

In this part:

- [Keyboard Shortcuts by Category](#)
- [Keyboard Shortcuts by Keystroke](#)

See Also

Procedures:

- [Configuring IDE Settings](#)

Reference:

- [Keymap](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

3.0+

Tool Windows Reference

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: 

View | Tool Windows

View | Tool Windows

The Tool Windows Reference contains detailed information about the functionality, controls and menus of the PhpStorm [tool windows](#).

In this section:

- [Project](#) (Alt+1Meta 1)
- [Favorites](#) (Alt+2Meta 2)
- [Find](#) (Alt+3Meta 3)
- [Run](#) (Alt+4Meta 4)
- [Debug](#) (Alt+5Meta 5)
- [TODO](#) (Alt+6Meta 6)
- [Structure](#) (Alt+7Meta 7)
- [Hierarchy](#) (Alt+8Meta 8)

- [Changes](#) (Alt+9Meta 9)
- [Inspection](#)
- [Version Control](#)
- [Data Sources](#)
- [Phing Build](#)

See Also

Reference:

- [Appearance](#)

Getting Started:

- [PhpStorm Tool Windows](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Changes Tool Window

Previous | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

VCS | Show Changes View

View | Tool Windows | Changes

Alt+9Meta 9

VCS | Show Changes View

View | Tool Windows | Changes

Alt+9Meta 9

The Changes tool window enables you to manage changelists and files within them, perform VCS-specific actions, or view modifications introduced by other team members. The actions can be performed using the toolbar buttons, or context menu commands.

Note

Changes tool window is only available when version control [is enabled](#) in your project.

The tool window consists of several tabs. The Local is always present, the set of other tabs depends on the version control system used in your project.

- [Local tab](#).
- [Repository tab](#). This tab is not available for [Git](#) and [Mercurial](#) integration.
- [Incoming tab](#). This tab is not available for [Git](#) and [Mercurial](#) integration.
- [Subversion working copies information tab](#) for [Subversion](#) integration.
- [Log tab](#) for [Git](#) and [Mercurial](#) integration.

See Also

Concepts:

- [Changelist](#)

Procedures:

- [Managing Changelists](#)
- [Viewing Changes Information](#)

Getting Started:

- [PhpStorm Tool Windows](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Local Tab

Previous | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

VCS | Show Changes View - Local

View | Tool Windows | Changes - Local

Alt+9Meta 9

VCS | Show Changes View - Local

View | Tool Windows | Changes - Local

Alt+9Meta 9

Local tab shows the files that have been modified locally and have not been committed to the repository yet.

Use this tab to [commit](#) and [revert](#) changes, [manage changelists](#), [view differences](#), and [clean up](#) locked folders.

Item	Tooltip and Shortcut	Description
	Ctrl+F5Command F5	Click this button to refresh information.
	Commit Changes	Click this button to check in the selected change or changelist. You can also attach and detach Perforce jobs to changelists via Commit Changes dialog.
	Revert	Click this button to roll back the selected changes.
	Alt+InsertCommand N	Click this button to create a new changelist .
	DeleteDelete	Click this button to delete the selected changelist. Note that you cannot delete the default changelist.
	Set Active Changelist	Click this button to make the selected changelist active . The active changelist is highlighted.
	F6F6	Click this button to move the selected item to another changelist.
	Ctrl+DCommand D	Click this button to show differences between your local version and the latest version in the repository.
	Ctrl+Add Command Add Or Ctrl+Equals Command Equals Ctrl+NumPad-Ctrl NumPad-	Click these buttons to expand or collapse all nodes.
	Ctrl+PMeta P	Click this button to display the changed files grouped by directories. If the button is released, the changed files are grouped by changelists.
	Ctrl+C Command C or Ctrl+Insert Command Insert	Click this button to copy the path to the selected file to the Clipboard.
	Configure Ignored Files	Click this button to configure the list of files that should be ignored by your version control system.
	Change Details	Click this toggle button to have PhpStorm open or close the dedicated Change Details pane
	F1F1	Click this button to show context reference page.

Change details pane

The pane opens when you click the **Change Details** button on the toolbar. In this pane, examine the changes made to the selected file compared to the base revision.

The pane consists of two areas:

- The affected code as it was in the base revision.
- The affected code as it is after the change is introduced.

The pane can be [split horizontally or vertically](#).

In each area, PhpStorm numbers both changes and the lines involved in them.

To close the pane, click the **Change Details** button once more.

Item	Tooltip and Shortcut	Description
	More/Less Lines	Click this button to open a slider and specify the number of lines to be shown above and below the updated code fragment at the caret.
	Next Change	Click this button to move to the next updated piece of code.
	Previous Change	Click this button to return to the previous updated code fragment.
	Settings	Click this button to show the list of options that define the appearance of the pane: <ul style="list-style-type: none"> • Top/Bottom: Select this option to have the pane split horizontally, so the base revision is shown in the upper part and the locally updated version is shown in the bottom part of the pane. • Left/Right: Select this option to have the pane split vertically, so so the base revision is shown in the left-hand part and the locally updated version is shown in the right-hand part of the pane. • Use soft wraps: Select this option to have the <code>soft wraps</code> (or <code>word wraps</code>) used.

See Also

Concepts:

- [Local, Committed and Incoming Changes](#)

Procedures:

- [Viewing Changes on UML Class Diagram](#)

Version Control:

- [Viewing Changes Information](#)
- [Viewing Details of Changes](#)
- [Attaching and Detaching Perforce Jobs to Changelists](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Repository and Incoming Tabs

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts:

VCS | Show Changes View - Repository/Incoming
 View | Tool Windows | Changes - Repository/Incoming
 Alt+9Meta 9

The **Repository** tab shows the changes committed to the repository under the VCS roots within the current project. The **Incoming** tab shows the changes committed to the repository by other team members, and not yet checked out locally. Both tabs display information stored in the history cache. The number of changelists displayed depends on the [cache scope](#).

Each tab contains the following panes:

- The [Changelists](#) pane shows changelists.
- The [Changed Files](#) pane shows the list of files that were modified and committed within the selected changelist.

Note

If you are using SVN 1.5 or higher both on the server and in the local working copies, the **Repository** tab also features a [Merge Info](#) pane that configures the view in the other two panes and provides control over integration between branches.

Changelists pane

The pane shows the changelists committed and stored in the history cache. When you click a changelist, the files affected by the selected commit are displayed in the [Changed Files](#) pane.

Note

Committed changelists often correspond to issues in tracking systems. You can have such issues opened in the browser right from the **Changelists** pane. This functionality has the following prerequisites:

1. The [pattern](#) of the bug tracking system is specified in the **Issue Navigation** page of the **Settings** dialog box.
2. The corresponding issue number is mentioned in the commit message.

After the issue navigation is configured, issue numbers in commit messages are rendered as links. Clicking such link brings you to the corresponding page of your issue tracker.

Item	Tooltip and Shortcut	Description	Available In
	Refresh Ctrl+F5 Command F5	Click this button to refresh information in the view.	Both tabs
	Show Details Ctrl+Q Command J	Click this button to show the following information on the selected changelist: <ul style="list-style-type: none"> • Changelist number • User and client name • Date and time of commit 	Both tabs
	Create Patch	Click this button to create a patch based on the selected changelist.	Repository tab
	Revert Changes	Click this button to create a <i>reverse patch</i> for the selected changelist and roll back the previously made changes. The Select Target Changelist dialog box opens.	Repository tab
		Warning	
		If the <i>reverse patch</i> applies to a version committed earlier, this roll back attempt may fail because of the conflicts with the later changes.	
		Note	
		You can use this action to revert changes committed by any user.	
	Update Project Ctrl+T Command T	Click this button to update the project to the latest version.	Incoming tab
	Expand All Ctrl+Add Command Add or Ctrl+Equals Command Equals	Click this button to have all nodes expanded.	Both tabs
	Collapse All Ctrl+Subtract Command Subtract or Ctrl+Minus Command Minus	Click this button to have all nodes collapsed.	Both tabs
	Copy Ctrl+C Command C or Ctrl+Insert Command Insert	Click this button to copy the commit message of the selected changelist to the Clipboard.	Both tabs
	Help F1 F1	Click this button to show the corresponding help topic.	Both tabs
	Highlight Integrated	Click this button to have the Merge Info pane displayed.	

Note

The button is enabled only when both the server side and the client side use Subversion 1.5.

Filter by Use this drop-down list to hide the changelists that are of no interest and view only those that meet a certain criterion. Both tabs

The available options are:

- User: select the user whose changelists you need to view.
- Structure: select the path to the module or folder the changelists committed to which you need to view.

Note

Select **None** to turn off filtering.

Group by In this drop-down list, select whether you need to group changelists by users who committed them or by date. Both tabs

Search field Use this text box to specify a search pattern to find the change list(s) with specific commit messages. As you type, the list dynamically reduces to show the changelists with the commit messages that match the entered pattern. Repository tab

To save the pattern, press **Enter**.

To view the list of recent search patterns, click the  button.

To clear the list of search patterns, click the  button.

Changed files pane

Item	Tooltip and Shortcut	Description
	Show Diff Ctrl+DCommand D	Click this button to show the differences between the current and previous revisions of the selected file.
	Show Diff with Local	Click this button to show the differences between the selected revision of the selected file and its current local copy.
	Edit Source F4F4	Click this button to open the local copy of the selected file for editing.
	Open Repository Version	Click this button to open the repository version of the selected file.
	Integrate to Branch	Click this button to integrate the changes from the selected file to the target branch.
	Properties Diff	Click this button to view the differences in file properties between the current version and the previous revision.
	Show History	Click this button to open the History view of the selected file in the Version Control tool window.
	Group by Directory Ctrl+PMeta P	Click this button to transform a flat list of files into a tree of packages with files.
	Expand All Ctrl+Add Command Add or Ctrl+Equals Command Equals	Click this button to expand all nodes. Note The button is available only when the files in the pane are displayed grouped by directories.
	Collapse All Ctrl+Subtract Command Subtract or Ctrl+Minus Command Minus	Click this button to collapse all nodes. Note The button is available only when the files in the pane are displayed grouped by directories.
	Select All Ctrl+ACommand A	Click this button to select all files in the Changed Files pane.

Merge info pane

Tip

The pane is available only if you are using SVN 1.5 or higher both on the server and in the local working copies.

In this pane, specify a pair of branches integration between which you need to monitor. The **Changelists** pane will show changelists related to the specified branches and provide information on the [integration status](#) of each changelist.

Note

You can specify several pairs of branches if several projects or roots are involved.

Item	Tooltip and Shortcut	Description
From		Specify the URL address of the source branch.
		Note PhpStorm suggests the URL address selected in the Checkout from Subversion dialog box.

- To Do the following:
- Specify the path to the target branch.
Click  or press `Shift+Enter` or `Shift Enter` to open the **Select Branch** dialog box.
 - Specify the path to the local working copy to which you will apply patches created based on selected changelists.
Click  to open the **Configure Working Copy Paths** dialog box and select a working copy.
-  **Highlight Integrated** Click this button to have each changelist in the **Changelists** pane supplied with an indication of whether it is integrated or not.
-  **Integrate To Branch** Click this button to integrate the selected changelist into the working copy.
The **Integrate To Branch** dialog box opens.
-  **Undo Integrate To Branch** Click this button to revert the last integration of the selected changelist into the working copy.
-  **Mark As Merged** Click this button to indicate that the selected changelist is integrated into the working copy without actually integrating the changelist.
The action affects the administrative information in the `.svn` folder.
The icon next to the selected changelist changes from  to .
-  **Mark As Not Merged** Click this button to indicate that the selected changelist is not integrated into the working copy without actually reverting integration.
Update the administrative information in the `.svn` folder.
The icon next to the selected changelist changes from  to .
-  **Filter Out Integrated** Click this button to display only changelists that have not been integrated into the working copy.
-  **Filter Out Not Integrated** Click this button to display only changelists that have been integrated into the working copy.
-  **Filter Out Others** Click this button to hide extraneous changelists in the **Changelists** pane.
- Note
Extraneous changelists are changelists that are [managed in another VCS](#) or are located under another root.
-  **Show Working Copies** Click this button to open the [Subversion Working Copies Information](#) dialog box.
-  **Refresh** Click this button to refresh the information in the **Changelists** pane.

See Also

Concepts:

- [Version Control with PhpStorm](#)
- [Local, Committed and Incoming Changes](#)

Procedures:

- [Viewing Changes Information](#)
- [Viewing and Fast Processing of Changelists](#)

Reference:

- [Version Control](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Log Tab

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts:  

Warning

The tab is available for [Git](#) and [Mercurial](#) integrations only.

VCS | [Show Changes View - Log](#)

[View](#) | [Tool Windows](#) | [Changes - Log](#)

Alt+9Meta 9

VCS | [Show Changes View - Log](#)

[View](#) | [Tool Windows](#) | [Changes - Log](#)

Alt+9Meta 9

The tab shows changes committed in all the branches of the local and remote repositories or in a specific local or remote branch.

The tab contains the following panes:

- The [Commits](#) pane shows the commits in the selected branch from the local and remote repositories.
- The [Changed Files](#) pane shows the list of files that were modified and committed within the selected commit.

Commits pane

The pane consists of three areas:

1. [Commits](#)
2. [Commit Details](#)
3. [Toolbar](#)

The screenshot shows the 'Commits' pane in PhpStorm. At the top, there are tabs for 'Changes', 'Local', and 'Log'. Below the tabs is a search bar and a toolbar with icons for search, refresh, and other actions. The main area displays a list of commits for the 'branch_2' branch. The commits are listed with their messages, authors, and dates. The selected commit is highlighted in blue. Below the list, the 'Commit Details' section shows the hash, author, committer, and description of the selected commit.

Item **Description**

Commits The area shows a list of all the commits performed in the selected branch or in all branches. For each commit, the list shows the commit message, the author, and the commit date. If you are viewing commits from all branches, both local and remote, each commit is supplied with a color label with the name of the branch or tag in which the commit was performed. The current branch is marked with an asterisk *. The color of the label depends on the type of the branch:

- Green for local branches.
- Violet for remote branches.
- Yellow for [tags](#).

Note

Committed changelists often correspond to issues in tracking systems. You can have such issues opened in the browser right from the **Commits** pane. This functionality has the following prerequisites:

1. The [pattern](#) of the bug tracking system is specified in the **Issue Navigation** page of the **Settings** dialog box.
2. The corresponding issue number is mentioned in the commit message.

After the issue navigation is configured, issue numbers in commit messages are rendered as links. Clicking such link brings you to the corresponding page of your issue tracker.

Commit Details This area provides details on the commit selected in the **Commits** list. All the branches in which the commit is made are also shown and their type - local or remote - is indicated.

Toolbar

Item	Tooltip and Shortcut	Description
	Find	Use this text box to search through the list of commits for entire or parts of commit messages or names of commit authors. As you type a search string, the commits with messages or authors that match the search pattern are displayed, with the matching character strings highlighted. To finalize the search, press EnterEnter . The previously used search patterns are stored in the search history list.
		Click this button to clear the search history list.
Branch		In this drop-down list, specify the branch to show commits in. To have commits from all local and remote branches shown, choose All . The current branch is indicated with an asterisk *.
User		In this drop-down list, specify the name of the author whose commits you need listed. To view commits from all users, choose All .
Load More		Click this button if you need to have older commits listed. The text on the button changes to Loading . When PhpStorm is through with loading a new portion of commits, the original text is restored.

Note

The button is only available when the **Commits** list does not contain the entire history.

-  **Cherry-pick** Click this button to have [the changes from the selected commit applied to the current branch](#).
-  **Create Patch** Click this button to [create a patch](#) based on the selected commit.
-  **Show graph** Click this toggle button to have the [graph of revisions](#) shown or hidden.
-  **Go to commit** Click this button to find a specific commit in the shown part of the graph. In the **Go to** pop-up tool-window that opens, specify the commit hash, or description fragment, or branch or tag name. PhpStorm navigates to the commit that meets the specified requirements and its visual presentation in the graph.
-  **Refresh** Click this button to refresh the information in the **Commits** pane.

Changed files pane

The pane shows the list of files that were modified and committed within the selected commit.

Item	Tooltip and Shortcut	Description
	Show Diff Ctrl+DCommand D	Click this button to show the differences between the current and previous revisions of the selected file.
	Show Diff with Local	Click this button to show the differences between the selected revision of the selected file and its current local copy.
	Edit Source F4F4	Click this button to open the local copy of the selected file for editing.
	Open Repository Version	Click this button to open the repository version of the selected file.
	Revert Selected Changes	Click this button to have the changes in the selected file abandoned.
	Properties Diff	Click this button to view the differences in file properties between the current version and the previous revision.
	Group by Directory Ctrl+PCommand P	Click this button to transform a flat list of files into a tree of packages with files.
	Expand All Ctrl+Add Command Add OR Ctrl+Equals Command Equals	Click this button to expand all nodes. Note The button is available only when the files in the pane are displayed grouped by directories.
	Collapse All Ctrl+Subtract Command Subtract OR Ctrl+Minus Command Minus	Click this button to collapse all nodes. Note The button is available only when the files in the pane are displayed grouped by directories.
	Select All Ctrl+ACommand A	Click this button to have all files in the Changed Files pane selected.

See Also

Procedures:

- [Committing Changes to a Local Git Repository](#)
- [Applying Changes from a Specific Commit to Other Branches \(Cherry Picking\)](#)
- [Version Control with PhpStorm](#)
- [Using Git Integration](#)

Reference:

- [Commit Changes Dialog](#)
- [Changes Tool Window](#)
- [Git Reference](#)
- [Version Control Reference](#)

External Links:

- <http://www.kernel.org/pub/software/scm/git/docs/git-cherry-pick.html>

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Command Line Tools Console Tool Window

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Tools | Run Command

Ctrl+Shift+XCtrl Shift X

Use the tabs of this tool window to [type and run PHP-specific commands](#) in the command line and to [validate tool definition files](#) for structure consistence.

The tool window consists of the following tabs and areas:

- The **Input** text box is displayed in the bottom of the tool window if you have chosen **Tool window** in the **Show console** in area of the [Command Line Tool Support](#) page. Otherwise, the **Command Line Tools Input** pane opens in a separate pop-up window.

In this text box, type the desired command in the format <tool alias> <command>

- The **output** tab shows the results of executing commands. The tab is named after the last invoked command.
- The **Framework definition file errors** tab is not shown by default. The tab is accessible only if any structure discrepancies are detected in the tool definition file.

Every time you invoke a command, PhpStorm performs [full validation](#) of the tool definition file. If the validation fails, PhpStorm displays a **Command Line Tool** pop-up window with a notification on validation failure. Upon clicking the **More** link, the **Framework definition file errors** tab opens showing messages on detected inconsistencies. Each message contains information on the file and the line number where the problem was found, as well as a brief description of the error.

You can close the tab by clicking the cross on its header. To re-open the tab, again click **More** in the **Command Line Tool** notification pop-up window, which remains on the screen until you close it manually.

Note

Upon validation failure, the tool is marked with the **Invalid description** icon  in the [Command Line Tool Support](#) page.

Toolbar options

Item	Tooltip and Shortcut	Description
	Stop Ctrl+F2 Command F2	Click this button to cancel execution of a command without closing the tool window.
	Up the Stack Trace Ctrl+Alt+Up Ctrl Alt Up	Click this button to navigate up in the stack trace.
	Down the Stack Trace Ctrl+Alt+Down Ctrl Alt Down	Click this button to navigate down in the stack trace.
	Use Soft Wraps	Click this button to toggle the soft wrap mode of the output.
	Scroll to the end	Click this button to navigate to the bottom of the output tab named after the last invoked command.
	Export to Text Alt+O Command O	Click this button to have the results of executing commands saved in a text file. In the Export Preview dialog box, that opens, specify the target file, and click Save .
	Ctrl+Shift+F4 Command Shift F4	Click this button to close the tool window. If one or more commands are still running, the Command Line Tool dialog box opens. Specify whether you want to stop them or leave running in the background by clicking one of the following buttons: <ul style="list-style-type: none"> • Terminate and close • Close without terminating

See Also

Procedures:

- [Running Command Line Tool Commands](#)
- [Updating a Command Line Tool](#)
- [PHP-Specific Guidelines](#)

Reference:

- [Command Line Tools Input Pane](#)
- [Command Line Tool Support](#)
- [PHP](#)

Getting Started:

- [PhpStorm Tool Windows](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Command Line Tools Input Pane

Previous | Next | [See Also](#) | [Comments](#) | Shortcuts: 

Tools | Run Command

Ctrl+Shift+XCtrl **Shift** **X**

In this input pane, type the desired command in the format <tool alias> <command>

The input pane is available in two modes:

- As a separate pop-up window, if you have chosen **Tool window** in the **Show console** in area of the [Command Line Tool Support](#) page.
- As the **Input** text box at the bottom of the **Command Line Tools** tool window.

View the results of command execution in the **output** tab of the dedicated [Command Line Tools Console](#) tool window. The tab is named after the last invoked command.

See Also

Procedures:

- [Running Command Line Tool Commands](#)
- [Updating a Command Line Tool](#)

- [PHP-Specific Guidelines](#)

Reference:

- [Command Line Tools Console Tool Window](#)
- [Command Line Tool Support](#)
- [PHP](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Database Console Tool Window

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

[Data Sources tool window](#) | < data source > |

The **Database Console** tool window opens when you select a data source in the [Data Sources](#) tool window and click the **Run Database Console** button on the toolbar. For each data source, a separate dedicated console is opened.

Use the **Database** console to [view, query, and modify](#) the information in the data sources [configured](#) in your workspace.

The **Database** console consists of the following panes:

- The [Input](#) pane is intended for entering commands in one of the supported languages.

Tip

The **Input** pane opens as a new editor tab as soon as the **Database** console is invoked.

- The [Result](#) pane shows results of executing SQL queries in a table form. The results of running each command are shown in a separate tab.
- The [Output](#) pane displays time stamps, execution time and error messages.
- The [Parameters](#) pane, located in the upper-right part of the console, is used to specify the values of parameters, encountered in the commands.

Tip

The usual editor techniques are available in the **Database** console.

Input pane

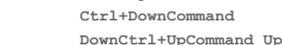
The **Input** pane opens as an editor tab. Use this pane to type, edit, and run SQL commands on data sources. For each data source PhpStorm opens a separate **Input** pane. For your convenience, the [common console toolbar icons](#) are also available from the **Input** pane.

Item	Shortcut	Description
	Ctrl+EnterCommand Enter	Click this button to run the command at the caret or all the selected consecutive commands.
	Browse history Ctrl+Alt+ECommand Alt E	Click this button to show the list of commands, executed in the console. You can select the desired commands, and place them to the Input pane.
	Properties Ctrl+Alt+EnterCommand Alt Enter	Click this button to open the Database Console Properties dialog box, where you can specify the SQL dialect to use and customize the console behaviour.
	Close Ctrl+Shift+F4Command Shift F4	Click this button to close the current instance of the Database console.
	Layout	Click this button to choose the desired layout of the toolbars from the pop-up menu. The available options are: <ul style="list-style-type: none"> • Horizontal Toolbars • Restore Layout

Result pane

Use this pane to view the result of executing queries as well as to add, remove, and update records in the queried table.

Item	Shortcut	Description
	Ctrl+EnterCommand Enter	Click this button to run the command at the caret or all the selected consecutive commands.
	Browse history Ctrl+Alt+ECommand Alt E	Click this button to show the list of commands, executed in the console. You can select the desired commands, and place them to the Input pane.
	Properties Ctrl+Alt+EnterCommand Alt Enter	Click this button to open the Database Console Properties dialog box, where you can specify the SQL dialect to use and customize the console behaviour.
	Close Ctrl+Shift+F4Command Shift F4	Click this button to close the current instance of the Database console.
	First Page	Click this button to switch to the page with the first/last set of results.

-  Last Page
-  Previous Page
-  Next Page
- 
 - Ctrl+DownCommand
 - DownCtrl+UpCommand Up

Note

The number of items to be shown in a result set is configured in the [Database Console Properties](#) dialog.

-  Reload Page
 - Ctrl+RCommand R
-  Add New Row
 - Alt+InsertCommand N
-  Delete Selected Rows
 - Ctrl+YCommand Y
-  Copy Query
 - Ctrl+Alt+Shift+CCommand
 - Alt Shift C
-  Copy
 - Ctrl+C Command C or
 - Ctrl+Insert Command
 - Insert

Output pane

This pane displays information on SQL command execution. This information includes:

- Time stamp of the executed command.
- Number of affected rows.
- Duration in milliseconds. If the command execution involves processing the data received from the data source, the execution time is shown as x/y ms, where x is the time taken to receive the data, and y is the total time of generating the query result which includes both receiving and processing the data.
- Error messages.

Item	Description
Clear All	Choose this option to have all the messages removed from the pane.
Copy Content	Choose this option to have PhpStorm copy all the displayed messages to the clipboard.
Copy Selected Content	Choose this option to have PhpStorm copy the selected messages to the clipboard.
Fold lines like this	Choose this option to open the Console Folding dialog box and add the current line to the list of patterns that determine lines to be folded in console output.
Compare with Clipboard	Choose this option to open the Differences Viewer an compare the contents of the clipboard with the entire contents of the Output pane or with a specific selected message.

Item Shortcut	Description
 Ctrl+EnterCommand Enter	Click this button to run the command at the caret or all the selected consecutive commands.
 Browse history Ctrl+Alt+ECommand Alt E	Click this button to show the list of commands, executed in the console. You can select the desired commands, and place them to the Input pane.
 Properties Ctrl+Alt+EnterCommand Alt Enter	Click this button to open the Database Console Properties dialog box, where you can specify the SQL dialect to use and customize the console behaviour.
 Close Ctrl+Shift+F4Command Shift F4	Click this button to close the current instance of the Database console.

Parameters pane

Item	Description
Parameter	This read-only field displays the names of parameters detected in the current SQL statement.
Value	In this text box, type the value to substitute for the parameter when the statement will be executed.

Context menu of a tab

Context menu command	Description	Available in
Detach	Choose this command to detach the selected view. This enables you to move the detached pane to any desired location.	All panes
Note		
The Output and the Result panes can be detached only together.		
Close	Choose this command to close the current tab.	Result pane
Focus on Startup	If this option is selected, the current pane gets the focus on launching the Database console.	All panes
Select Next Tab/	Use these commands to switch the focus between tabs within the Result pane or between the Output and the Result panes.	Output pane

Select Previous Tab

Result pane

See Also

Concepts:

- [Data Sources](#)

Procedures:

- [Creating and Importing Data Sources](#)
- [Accessing Data Sources Via the Database Console](#)

Reference:

- [Regular Expression Syntax Reference](#)
- [Advanced Editing](#)
- [Basic Editing](#)

Getting Started:

- [PhpStorm Tool Windows](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Database Console Properties Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

The dialog box opens when you click the **Properties** toolbar button  in the [Database Console](#) tool window.

In this dialog box, specify the SQL dialect to use and define how parameters should be parsed. Here you can also configure displaying query results, and define layout to export query execution results.

The dialog box consists of the following tabs:

- [General](#)
- [Data Export](#)
- [Text Mode](#)

General tab

In this tab configure the common behaviour of the Database console.

Item	Description
SQL Dialect	From this drop-down list, select the SQL dialect to use. The selection determines the coding assistance provided. The available options are: <ul style="list-style-type: none"> • Derby • H2 • HSQLDB • MySQL • Oracle • PostgreSQL • PostgresPLSQL • SQL Server • SQL92 • SQLite • Sybase • keywords only. The coding assistance will be limited to highlighting the SQL keywords. • Plain text. The corresponding content will be treated as plain text.
Connection in auto-commit mode	Select this check box to have SQL statements executed and committed as individual transactions.
Result Set Page Size	In this text box, type the maximum number of retrieved records to display per page. The zero value stands for unlimited page size.
Show query results in new tab	Select this check box to have a separate tab created for each query execution.

Data export tab

In this tab, define how query execution results will be presented if you need to export them.

Item	Description
String Quotation	In this pair of text boxes, type the symbols to enclose exported String data.
Values Separator	In this text box, type the symbol to use as separator between exported values.
Include Table Header	Select this check box to have the headers of tables exported.
Include Row Number	Select this check box to have id numbers of exported records displayed.

Text mode tab

In this tab, configure parsing parameters inside SQL commands.

Item	Description
Parameter Pattern (Java RegExp)	Select this check box to have parameters within commands parsed using a regular expression and specify the desired regular expression in th text box.

See Also

Procedures:

- [Accessing Data Sources Via the Database Console](#)

Reference:

- [Database Console Tool Window](#)
- [Editor](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Data Sources Tool Window

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

[View](#) | [Tool Windows](#) | [Data Sources](#)

[Tools](#) | [Data Sources](#)

[View](#) | [Tool Windows](#) | [Data Sources](#)

[Tools](#) | [Data Sources](#)

Use the **Data Sources** tool window to:

- Manage available data sources. (These are shown as a tree of data sources, tables and table columns.)
- View open database connections and close the unnecessary ones.
- View the structure of tables within data sources in a UML class diagram.
- View table information in a quick documentation pop-up window.

In this section:

- [Toolbar buttons](#)
- [Context menu commands and shortcuts](#)
- [Tree View Nodes](#)

Toolbar

Icon	Shortcut	Description
	Add Data Source Alt+InsertCommand N	Click this button to create a new data source. Select the data source type or the import option: <ul style="list-style-type: none"> • DB Data Source. Select this option to create a new DB data source. Specify the properties of the new data source in the DB Data Source Properties dialog. • DDL Data Source. Select this option to create a new DDL data source. Specify the properties of the new data source in the DDL Data Source Properties dialog. • Import from Project Files. Select this option to create one or more data sources by importing the corresponding configuration files. Select the configuration files of interest in the Import Data Sources dialog.
	Remove DeleteDelete	Click this button to remove the selected data source entry from the list.
	Close Database Connection DeleteDelete	Click this button to close the selected active database connection. Note The button is available only when a connection is selected.
	Copy Data Source Ctrl+DCtrl D	Click this button to have a copy of the selected data source created with the incremental name.
	Data Source Properties EnterEnter	Click this button to open the Data Source Properties dialog, where you can view or edit the properties of the selected data source. Depending on the data source type, one of the following dialogs will open: <ul style="list-style-type: none"> • DB Data Source Properties. • DDL Data Source Properties.
	Refresh Tables Ctrl+Alt+YCommand Alt Y	Click this button to have the local data source updated and the tables loaded.
	Run Database Console Ctrl+Shift+F10Command Shift F10	Click this button to run the database console.
	Open Table Editor F4F4	Click this button to open the selected database table in the Table Editor . Note The Table Editor is not available for the data sources of the DDL type .

See also, [Manipulating Table Data in the Table Editor](#).

 **Generate and Copy DDL**
 Ctrl+C Command C or
 Ctrl+Insert Command Insert

Click this button to have data structure definition statements generated based on the selected node.

Note
 PhpStorm saves the generated statements in the clipboard.

 **Compare**

Click this button to [view differences in schemas](#) between two selected DB objects.

Note
 This button is only enabled when two data sources, schemas, tables or columns are selected.

 **Collapse All**
 Ctrl+Subtract Command
 Subtract or Ctrl+Minus Command
 Minus

Click this button to have all data source nodes folded.



Click this button to view or change the options that define how the data source items are arranged in the tree view:

- **Flatten Schemas.** Select this option to make the tree flatter: schemas are shown as the top hierarchical elements, at the same level as the data sources.
- **Group Tables by Type.** Select this option to have schema objects ordered by their types (sequences, views, tables, etc.). (When this option is off, the objects are ordered by their names alphabetically.)

Context menu

Item	Icon / Shortcut	Description	Available from
Add Data Source	 Alt+InsertCommand N	Choose this option to create a new data source. Select the data source type or the import option: <ul style="list-style-type: none"> •  DB Data Source. Select this option to create a new DB data source. Specify the properties of the new data source in the DB Data Source Properties dialog. •  DDL Data Source. Select this option to create a new DDL data source. Specify the properties of the new data source in the DDL Data Source Properties dialog. •  Import from Project Files. Select this option to create one or more data sources by importing the corresponding configuration files. Select the configuration files of interest in the Import Data Sources dialog. 	All contexts
Remove	 DeleteDelete	Choose this option to remove the selected data source entry from the list.	Data source node
Close Database Connection	 DeleteDelete	Choose this option to close the selected active database connection. Note that PhpStorm opens connections when necessary without any steps from your side, and displays them beneath the corresponding data source nodes.	Active connection
Copy Data Source	 Ctrl+D Ctrl D	Choose this option to have a copy of the current data source created with the incremental name.	Data source node
Copy Reference	Ctrl+Alt+Shift+C Command Alt Shift C Ctrl+C Command C or Ctrl+Insert Command Insert	Choose this option to copy the fully qualified name of the selected item to the clipboard. Press this shortcut to copy the name of the selected object to the clipboard. As an alternative, drag-and-drop an object from the Data Sources tool window to the editor.	All contexts Data source, table, and column nodes.
Data Source Properties	 EnterEnter	Choose this option to open the Data Source Properties dialog, where you can view or edit the properties of the selected data source. Depending on the data source type, one of the following dialogs will open: <ul style="list-style-type: none"> • DB Data Source Properties. • DDL Data Source Properties. 	All contexts
Refresh Tables	 Ctrl+Alt+Y Command Alt Y	Choose this option to have the current data source updated and the database tables loaded.	All contexts
Run Database Console	 Ctrl+Shift+F10 Command Shift F10	Choose this option to run the Database console.	All contexts
Open Table Editor	 F4F4	Choose this option to open the selected database table in the Table Editor . Note The Table Editor is not available for the data sources of the DDL type . See also, Manipulating Table Data in the Table Editor .	DB table node
Generate and Copy DDL	 Ctrl+C Command C or	Choose this option to have data structure definition statements generated based on the selected node.	All contexts

	Ctrl+Insert Command Insert	Note The generated statements are stored in the clipboard.	
Diagrams	 Ctrl+Shift+Alt+UCommand Shift Alt U Ctrl+Alt+UCommand Alt U	Select this option, and then select one of the possible ways to view database structure: <ul style="list-style-type: none"> • Show Visualization. A UML class diagram is shown on a separate editor tab. • Show Visualization Popup. A UML class diagram is shown in a pop-up window. 	Data source node
View table data	Ctrl+QCommand J	Select the desired table and press Ctrl+QCommand J to view the create table query for the table and its first 10 rows.	This action has no item in the context menu.
Find Usages	Alt+F7Alt F7	Select the desired node (data source, schema, table, column) and press Alt+F7Alt F7. The usages of tables and columns include database schema usages, i.e. references in indices, primary and foreign keys.	All contexts
Compare		Select two nodes, and choose this command on the context menu. The differences are shown in Differences Viewer for Folders and DB Objects.	All contexts

Tree view nodes

Icon Description

-  DB data source
-  DDL data source
-  Schema
-  Table
-  Column
-  Column with a primary key
-  Column with a foreign key
-  Column with an index
-  Primary key
-  Foreign key
-  Index
-  [Database connection](#) node for a data source.

PhpStorm establishes a database connection automatically first time it needs to execute a query, and closes it automatically upon the session end. If necessary, a connection can be terminated manually, using the toolbar button .

See Also

Concepts:

- [Data Sources](#)

Procedures:

- [Creating and Importing Data Sources](#)
- [Accessing Data Sources Via the Database Console](#)

Reference:

- [Database Console Tool Window](#)
- [Viewing Local History of a File or Folder](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

DB Data Source Properties

Previous | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

To access this dialog:

- When creating a new data source:
Tools | Data Sources |  or Alt+InsertCommand N | DB Data Source
View | Tool Windows | Data Sources |  or Alt+InsertCommand N | DB Data Source
- For an existing data source:
Tools | Data Sources | select the data source |  or EnterEnter
View | Tool Windows | Data Sources | select the data source |  or EnterEnter

Use this dialog box to configure JDBC data sources and behaviour of the [Database Console](#) tool window.

The dialog box consists of the following tabs:

- [Database](#)
- [Schemas & Tables](#)

- [Console](#)
- [Advanced](#)

Common options

The following controls are available in all of the tabs.

Item	Description
Data Source Name	In this text box, specify the data source name.
Data Source Level	Use this drop-down list to specify scope in which the data source will be available. The available options are: <ul style="list-style-type: none"> • Project - select this option to enable access to the data source only from the current project. • Global - select this option to enable access to the data source from any project in your workspace.
Test connection	Click this button to check that the specified settings ensure successful connection to the target database.
Refresh tables	Click this button to download the data from the target database for a newly configured data source or to synchronize to the target database for a previously configured data source.

Database tab

In this tab, configure the JDBC data sources to use.

Item	Description
JDBC Driver Files	In this drop-down list, specify the library of archive to search for the class that implements the JDBC driver of required type. Choose one of the suggested JDBC driver sources from the list (if necessary, the selected driver files will be downloaded automatically), or click the Browse button and select the relevant library or archive in the Choose JDBC Driver Files dialog box that opens.
The following files will be used:	This read-only list box shows all the relevant archives detected at the location specified in the JDBC Driver Files field.
JDBC Driver Class	From this drop-down list, select the name of the class that will implement the driver of the desired type. The list shows all the driver classes detected in the specified archives and libraries.
Database URL	In this field, specify the URL of the target database.
User Name	In this text box, type the user name to access the target database with.
Password	In this text box, type the password to access the target database with.

Schemas & tables tab

In this tab, specify the [schemas](#) or table name patterns to restrict the set of accessible tables in the target databases.

Tip

Specifying a schema makes sense if the target database supports schemas.

Item	Description
Schemas	This read-only field lists all the schemas that are currently available in the system.
Scan for Tables	Select the check boxes next to the schemes which you want to use for retrieving tables. If you select the check box next to *, tables will be scanned in all schemes.
Make Default	If you select this check box next to a schema, the names of the tables accessed through this schema will be resolved by PhpStorm so you do not need to write their fully qualified names.
Table Names Pattern	In this text box, type a regular expression which determines the subset of accessible tables. PhpStorm will access the tables with names that match the specified pattern.

Tip

An asterisk * means that all available tables will be loaded.

Console tab

In this tab, configure the general behaviour of the [Database Console](#) tool window.

Item	Description
Default SQL Dialect	From this drop-down list, select the SQL dialect to be used by default. The available options are: <ul style="list-style-type: none"> • Derby • H2 • HSQldb • MySQL • Oracle • PostgreSQL • PostgresPLSQL • SQL Server • SQL92 • SQLite • Sybase • keywords only. The coding assistance will be limited to highlighting the SQL keywords. • Plain text. The corresponding content will be treated as plain text.

Default Run Configuration In this drop-down list, specify the [run configuration](#) to launch the [Database Console](#) tool window.

Advanced tab

In this tab, configure the connection properties manually. This is necessary if your JDBC driver (for example, DB2) does not return the list of supported properties and therefore connection cannot be established successfully. To edit a parameter, click the Value field next to the desired parameter and update the field contents as necessary.

See Also

Procedures:

- [Creating and Importing Data Sources](#)
- [Configuring a DB Data Source](#)

Reference:

- [Data Sources Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

DDL Data Source Properties

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

To access this dialog:

- When creating a new data source:
Tools | Data Sources |  or Alt+InsertCommand N | DDL Data Source
View | Tool Windows | Data Sources |  or Alt+InsertCommand N | DDL Data Source
- For an existing data source:
Tools | Data Sources | select the data source |  or EnterEnter
View | Tool Windows | Data Sources | select the data source |  or EnterEnter

Use this dialog box to define DDL data source properties based on the existing DDL file.

Item	Shortcut	Description
Data Source Name		In this text box, specify the data source name.
	InsertInsert	Click this button to add an SQL file that contains the DDL statements to be used for creating a DDL data source.
	DeleteDelete	Click this button to remove the selected entry from the list.
		Use these buttons to change the order of DDL files in the list.
Parent Data Source		Use this field to select the desired parent data source from the list of already configured ones.

See Also

Concepts:

- [Data Sources](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Import Data Sources Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Tools | Data Sources |  or Alt+InsertCommand N | Import from Project Files
View | Tool Windows | Data Sources |  or Alt+InsertCommand N | Import from Project Files

Use this dialog to create data sources by importing the configuration files that contain the necessary data definitions.

You can import the data definitions from Spring, Hibernate, JPA and Tomcat (context.xml) configuration files.

Note
 If the corresponding configuration files have been developed externally, you must add them to your project prior to performing the import.

The **Import Data Sources** dialog contains a table that shows a tree view of parameters and their values for the configuration files found in the project. The bottom section of the dialog shows the source code of the selected configuration file where the parameters are taken from.

Item	Shortcut	Description
	Data Source Properties	Click this button to open the Data Source Properties dialog box.
	Select All	Click this button to select all locally encountered data sources.
	Unselect All	Click this button to deselect all locally encountered data sources.

- 
Expand All
Click this button to have all nodes expanded.
Ctrl+Add
Command Add or
Ctrl+Equals
Command Equals
- 
Collapse All
Click this button to have all nodes collapsed.
Ctrl+Subtract
Command
Subtract or
Ctrl+Minus
Command Minus

Test Connection Click this button to check whether connection to the selected data source can be established successfully.

See Also

Language and Framework-Specific Guidelines:

- [Data Sources](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Debug Tool Window

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

[View](#) | [Tool Windows](#) | [Debug](#)

Alt+5Meta 5

[View](#) | [Tool Windows](#) | [Debug](#)

Alt+5Meta 5

The **Debug** tool window becomes available, when you start [debugging](#).

The **Debug** tool window is divided into several areas:

- [Console](#)
- [Frames](#)
- [Variables](#)
- [Watches](#)

Each area can be shown as a **pane**, or as a **tab**, detached or attached, shown or hidden. The debug toolbar to the left and the stepping toolbar on top of the Debug tool window help you control the debugging session.

This topic provides reference information about the [toolbars of the panes](#), [debug toolbar](#) and the [stepping toolbar](#).

Pane toolbars

Item	Tooltip	Description
	Move to tab / Move to grid	Click these buttons to toggle between showing an area as a tab, or as a pane (a cell of a grid).
	Detach/Attach	Click these buttons to toggle between attached and detached modes of a pane or tab. When a pane or tab is detached, you can move it to any location.
	Hide	Click this button to hide a pane. To show it again, click  on the debug toolbar and choose Restore layout on the submenu, or click the pane's icon  in the upper-right corner of the tool window. This button is only available when an area is shown as a pane.
	Close	Click this button to close a tab or a pane.

Debug toolbar

Item	Tooltip and Shortcut	Description
	Rerun Ctrl+F5Command F5	When the application is stopped, click this button to run it again.
	Resume Program F9F9	When the application is paused, click this button to resume the program execution.
	Pause Program Ctrl+PauseCtrl Pause	Click this button to pause the program execution.
	Stop Ctrl+F2Command F2	Click this button to terminate the current process externally by means of the standard <code>shutdown</code> script.
	View Breakpoints Ctrl+Shift+F8Command Shift F8	Click this button to have the Breakpoints dialog box displayed where you can set the behavior of your breakpoints.
	Mute Breakpoints	Use this button to toggle the status of the breakpoints (enabled/disabled). You can temporarily disable all breakpoints in the project and thus have the program executed without stopping at breakpoints.
	Layout	Click this button to select the suitable layout from a list of available layout options, that appears: <ul style="list-style-type: none"> • Restore Layout - choose this option to have the changes to the current layout abandoned and return to the default state.

- **Horizontal Toolbars** - choose this option to change positioning of toolbars on the panes **Frames**, **Console**, **Watches**, or **Variables**. By default the toolbar on each pane is located horizontally, next to the pane's name.

	Pin	When this button is pressed, the current tab will not be overwritten; instead, the results of the next command will be displayed in a new tab.
	Close Ctrl+Shift+F4Command Shift F4	Click this button to disconnect from the process.
	Help F1F1	Click this button to have the reference topic displayed.

Stepping toolbar

Item	Tooltip and Shortcut	Description
	Show Execution Point Alt+F10Alt F10	Click this button to have the current execution point highlighted in the editor and have the corresponding stack frame shown in the Frames pane.
	Step Over F8F8	Click this button to have execution run until the next line in the current method or file, skipping the methods referenced at the current execution point (if any). If the current line is the last one in the method, execution steps to the line executed right after this method.
	Step Into F7F7	Click this button to have the debugger step into the method called at the current execution point.
	Force Step Into Alt+Shift+F7Alt Shift F7	Click this button to have the debugger step into the method called in the current execution point even if this method is to be skipped.
	Step Out Shift+F8Shift F8	Click this button to have the debugger step out of the current method, to the line executed right after it.
	Run to Cursor Alt+F9Alt F9	Click this button to resume the program execution and pause until the execution point reaches the line at the current cursor location in the editor. No breakpoint is required. Actually there is a temporary breakpoint set for the current line at the caret, which is removed once your program execution is paused. Thus, if the caret is positioned at the line which has already been executed, the program will be just resumed for further execution, because there is no way to roll back to the previous breakpoints. This action is especially useful when you have stepped deep into the methods sequence and need to step out of several methods at once.
	Note	If there are breakpoints set for the lines that should be executed before bringing you to the specified line, the debugger will pause at the first encountered breakpoint.
	Tip	Use this action when you need a kind of a temporary breakpoint at a specific line, where the program execution should not be interrupted.

See Also

Concepts:

- [Running and Debugging](#)

Procedures:

- [Debugging](#)

Reference:

- [Debug Tool Window. Frames](#)
- [Debug Tool Window. Console](#)
- [Debug Tool Window. Watches](#)
- [Debug Tool Window. Variables](#)
- [Debugger](#)

Getting Started:

- [PhpStorm Tool Windows](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Debug Tool Window. Console

Previous | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

This pane enables you to view the output and error stream messages, and optionally use the interactive command line console.

Toolbar buttons

Item	Tooltip and shortcut	Description
	Up the Stack Trace Ctrl+Alt+UpCtrl Alt Up	Click this button to navigate up in the stack trace and have the cursor jump to the corresponding location in the source code.

-  **Down the Stack Trace** Click this button to navigate down in the stack trace and have the cursor jump to the corresponding location in the source code.
Ctrl+Alt+DownCtrl Alt Down
-  **Use Soft Wraps** Click this button to toggle the soft wrap mode of the output.
-  **Scroll to the end** If this button is pressed, the caret is always kept at the last line of the console output.

Context menu options

Item	Description
Clear All	Clears the output window.
Copy Content	Copies the output to the Clipboard.
Compare with Clipboard	Opens the Clipboard vs Editor dialog box that allows you to view the differences between the selection from the editor and current Clipboard content. This dialog is a regular comparing tool that enables you to copy line at caret to the Clipboard, find text, navigate between differences and manage white spaces.

See Also

- Concepts:
- [Running and Debugging](#)
- Procedures:
- [Debugging](#)
- Reference:
- [Debug Tool Window. Frames](#)
 - [Debugger](#)
 - [Differences Viewer](#)
- Web Resources:
- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>

Debug Tool Window. Frames

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

To examine the values stored in a frame, use the [Variables pane](#) of the Debug tool window.

Toolbar

Item	Shortcut	Description
	Ctrl+Alt+UpCommand	Use the arrow buttons to navigate through the frame stack.
	Alt Up	
	Ctrl+Alt+DownCommand	
	Alt Down	

See Also

- Concepts:
- [Running and Debugging](#)
- Procedures:
- [Debugging](#)
- Reference:
- [Debug Tool Window. Console](#)
 - [Debug Tool Window. Variables](#)
 - [Debugger](#)
 - [Export Threads](#)
- Web Resources:
- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>

Debug Tool Window. Variables

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

The **Variables** pane enables you to examine the values stored in the objects of your application. When a stack frame is selected in the [Frames pane](#), the **Variables** pane displays all the data within it scope (method parameters, local and instance variables).

In this pane you can set labels for the objects, inspect objects, evaluate expressions, add variables to watches and more.

- In this topic:
- [Context Menu Options](#)
 - [Types of Variables](#)

Context menu options

Item	Shortcut	Description
Inspect		This command is available for fields, local variables and reference expressions, and opens a non-modal Inspection window, where you can concentrate on a particular reference. You can open as many Inspection windows as required. The view in the Inspection window is the same as in the Watches pane, but requires less screen space.
Set Value	F2F2	This command enables you to change the runtime value of a field or variable.
Jump to Source	F4F4	This command opens the source code of the selected variable or field in the editor and places the caret on a proper line.
Add to Watches		This command is available for all nodes except static. Use this command to create an expression that references the node and add this expression to the Watches pane.
Copy Value		Use this command to copy the value of a node to the Clipboard.

Tip

If a string value is too long to fit in the stack frame view, it is truncated. You can use this command to copy the value to the Clipboard, and then paste it to the editor, where you can examine the contents. Alternatively, hover your mouse cursor over the value and view the contents at the tooltip.

Types of variables

Icon	Description
	Field
	Array
	Primitive type

See Also

- Concepts:
- [Running and Debugging](#)
- Procedures:
- [Debugging](#)
- Reference:
- [Debug Tool Window. Frames](#)
 - [Debug Tool Window. Console](#)
 - [Debug Tool Window. Variables](#)
- Web Resources:
- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>

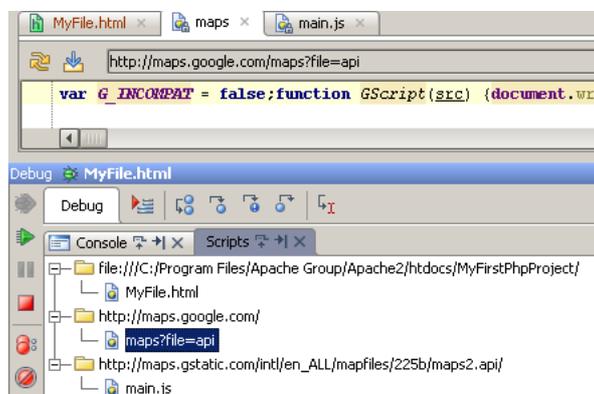
Debug Tool Window: Scripts

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Warning

The tab is only available during a JavaScript debugging session.

The tab shows all the external resources that are automatically downloaded during the JavaScript code execution. Use this tab to monitor downloading the external resources. To open the code of a resource in the editor, double-click the item in question.



See Also

- Reference:
- [Debug Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Debug Tool Window. Watches

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

In the **Watches** pane you can evaluate any number of variables or expressions in the context of the current stack frame. The values are updated with each step through the application, and become visible every time the application is suspended.

Watches pane shows multiple expressions that persist from one debug session to another, until you remove them. .

Note

Watch expressions are always evaluated in the context of a stack frame that is currently inspected in the [Frames](#) pane. If an expression cannot be evaluated, it is displayed with a question mark.

In this topic:

- [Toolbar](#)
- [Context Menu Options](#)

Toolbar

Item	Shortcut	Description
	InsertInsert	Click this button to create a new watch.
	DeleteDelete	Click this button to remove the selected watch from the list.

Context menu options

Item	Shortcut	Description
New Watch	InsertInsert	Choose this command to create a new watch. A text field opens, where you can enter new watch expression.
Remove Watch	DeleteDelete	Choose this command to delete the currently selected watch expression from the list.
Edit	F2F2	Choose this command to change the selected watch expression.
Remove All Watches		Choose this command to delete all watch expressions from the list.
Inspect		Available for fields, local variables and reference expressions. Choose this command to open the Inspect window for the node, which allows you to perform the same operations as those available in the stack frame, with the only difference that the root node is the one you have selected. You can recursively call the new Inspect windows from within each other. Each window is not modal and immediately reflects all changes in its subtree.
Jump to Source	F4F4	This command opens the source code of the selected variable or field in the editor and places the caret on a proper line.
Copy Value	Ctrl+C Command C or Ctrl+Insert Command Insert	Copies the selected node value to the clipboard.
Copy Name		Copies the selected node name to the clipboard.

See Also

Concepts:

- [Running and Debugging](#)

Procedures:

- [Debugging](#)

Reference:

- [Debug Tool Window. Frames](#)
- [Debug Tool Window. Console](#)
- [Debug Tool Window. Variables](#)
- [Debugger](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Duplicates Tool Window

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

[View](#) | [Tool Windows](#) | [Duplicates](#)

[View](#) | [Tool Windows](#) | [Duplicates](#)

The **Duplicates** tool window displays results of the search for duplicates.

In this section:

- [Panels of the Duplicates Tool Window](#)
- [Toolbar Buttons](#)
- [Context Menu Commands](#)

Panels of the duplicates tool window

The window consists of the following panes:

- The **left pane** displays the tree view of the duplicate fragments of source code. Each node shows the following information:
 - The number of duplicated code fragments found in scope.
 - The 'cost' of the duplicate (which is an arbitrary unit calculated using an additive algorithm on the base of the code block size; generally, the larger is the code fragment, the higher is its cost).
 - The containing class where the duplicates are located.
- The **right pane** shows the differences between the duplicated fragments of source code, selected in the left pane.

Toolbar buttons

Item	Shortcut	Description
	Rerun	Click this button to rerun the duplicates analysis in the active tab.
	Close Active Tab Ctrl+Shift+F4Command Shift F4	Click this button to close the active tab.
	Autoscroll to Source	If the button is pressed, selecting an entry in the left pane opens the respective file in the editor.
		Click this button to have the selected item shown as the left diff version.
		Click this button to have the selected item shown as the right diff version.
	F1F1	Click this button to show reference.
	F7F7 / Shift+F7Shift F7	Move to the Next/Previous Difference
Ignore whitespace		Use this drop-down list to define how the differences viewer should treat white spaces in the text. <ul style="list-style-type: none"> • Do not ignore: white spaces are important, and the differences should be highlighted. • Leading and Trailing: Ignore the differences, if they appear in the end and in the beginning of a line. • All: white spaces are not important, regardless of their location in the source code.
Legend		Shows summary information about the encountered differences: number of differences found, and color map. <p>Tip</p> Color map for the Differences viewer is configurable in the Colors and Fonts dialog box.

Context menu commands

Item	Keyboard Shortcut	Description
Jump to Source	F4F4	Open in the editor the file that contains the selected duplicate, and place the caret at the beginning of the duplicate. The fragment of code is highlighted.
Show Source	Ctrl+EnterCommand Enter	Open in the editor the file that contains the selected duplicate, and highlight the fragment of code.
Send to left/Send to right		Use these commands, or the arrow icons and to place the selected duplicate to the left or right pane of the differences viewer.

See Also

Procedures:

- [Analyzing Duplicates](#)
- [Extract Method](#)

Reference:

- [Differences Viewer](#)

Getting Started:

- [PhpStorm Tool Windows](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

```
ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js'; var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s); })();
```



PhpStorm 3.0.0 Web Help

File Transfer Tool Window

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

View | Tool Windows | File Transfer

Use this tool window to view logs while uploading files or folders to a remote server via the FTP or SFTP protocol.



Click thumbnail to view larger image.

See Also

Reference:

- [Deployment](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

JSTestDriver Server Tool Window

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

View | Tool Windows | JSTestDriver Server

View | Tool Windows | JSTestDriver Server

Note

To get access to the tool window, [download](#), [install](#), and [enable](#) the **JSTestDriver** [plugin from the JetBrains default repository](#). Then restart PhpStorm for the changes to take effect.

In this tool window, start and stop the JSTestDriver Server to run unit tests against and capture the browser to execute unit tests in.

Item	Tooltip and shortcut	Description
	Run a local server	Click this button to have PhpStorm launch the default JSTestDriver server.
	Stop the local server	Click this button to have PhpStorm stop the currently running JSTestDriver server.
Capture a browser using the URL		This read-only field shows the URL address to access the Remote Console of the JSTestDriver . Copy the URL address and open it in the browser of your choice.
		The icons indicate available browsers. The icon that corresponds to the browser you just opened, is active. Click the icon to get ready for executing tests.

See Also

Procedures:

- [Configuring JavaScript Libraries](#)

Getting Started:

- [PhpStorm Tool Windows](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Testing Support:

- [Unit Testing JavaScript](#)
- [Running JavaScript Unit Tests in Browser](#)
- [Running and Debugging](#)
- [Testing](#)

Project Tool Window

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

View | Tool Windows | Project

Alt+IMeta 1

View | Tool Windows | Project

Alt+IMeta 1

The **Project** tool window enables you to observe your project structure from the various viewpoints.

Item	Description
View as	Use this drop-down list to show your project as a directory tree, or a set of scopes .
	Click this button to navigate from a file in the Editor that gets the focus, to the corresponding node in the Project tool window.
	Click this button to collapse all nodes (Ctrl+Subtract Command Subtract or Ctrl+Minus Command Minus)



Click this button to open the [Scopes](#) dialog box where you can create and modify scopes. This button is available in the Scopes View.



Click this button to show the list of options that define how the project tree view will be rendered:

- **Show Members:** if this option is selected, all fields and methods will be displayed.
- **Autoscroll to Source:** if this option is selected, PhpStorm navigates from the selected file or field in the Project tool window to the corresponding source file in the editor (opening it, if necessary).
- **Autoscroll from Source:** if this option is selected, PhpStorm navigates from a file in the Editor that gets the focus, to the corresponding node in the Project tool window.
- **Sort by type:** if this option is selected, the node elements will be sorted by type rather than in alphabetical order.



Below this node, PhpStorm displays all the available libraries that are stored outside the [project content roots](#).

External
Libraries

See Also

Concepts:

- [Project](#)
- [Scope](#)

Procedures:

- [Creating and Managing Projects](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Favorites View

Previous | Next | [See Also](#) | [Comments](#) | Shortcuts: Mac

[Project Tool Window](#) | [View as](#) | [Favorites](#)

Favorites view helps you group the most needed items in the Favorite lists.

This section describes how to:

- [Add to Favorites](#)
- [Remove from Favorites list](#)
- [Delete Favorites list](#)
- [Rename Favorites list](#)
- [Move Favorite items between lists](#)

To add items to favorites

1. Select one or more items in the **Project** tool window or open a file in the Editor, and do one of the following:
 - Choose **Add to Favorites** on the main View menu, or on the context menu of the selection.
 - Press **Alt+VAlt V**, and then press **AA**.
2. On the submenu, select an existing Favorites list, or choose **Add to New Favorites List**.
3. In the Add New Favorites List dialog enter the desired group name, or accept default.

To remove items from favorites

1. Select one or more items in the Favorites view.
2. Do one of the following:
 - Choose **Remove from Favorites** on the context menu.
 - Press **Ctrl+DeleteAlt Delete**.

To delete favorites lists

1. Show Favorites view.
2. Right-click Favorites list and choose one of the following commands:
 - **Delete Favorites List <name>**, to delete the current list.
 - **Delete All Favorites Lists Except <name>**, to delete all lists except the current one.

To rename a favorites list

1. Show Favorites view.
2. Right-click Favorites list and choose **Rename Favorites list** on the context menu.
3. In the dialog **New Name for Favorites List**, enter the new name and click **OK**.

To move favorites items between the favorite lists

1. In the Favorites view, right-click the necessary item.
2. Choose **Send to Favorites** on the context menu.
3. On the submenu, select the target list, or choose **Send to New Favorites List**, to create a new one.

See Also

Concepts:

- [Project](#)
- [Scope](#)

Procedures:

- [Creating and Managing Projects](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Find Tool Window

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

View | Tool Windows | Find

Alt+3Meta 3

View | Tool Windows | Find

Alt+3Meta 3

Find tool window displays results of the following searches:

- [Find/Replace in Path](#)
- [Find Usages](#)
- [Structural Search and Replace](#)
- [Refactoring Preview](#)
- [Find Usages](#) of a data source, table or column.

The results of each search display in a separate tab, or replace the contents of the current tab, depending on the **Open in new tab** dialog setting. By default the window appears at the bottom of the screen. It has a toolbar with a set of buttons, a pane of results, and additional buttons for [Replace in Path](#), [Structural Replace](#), and [Refactoring Preview](#) operations.

In this section:

- [Toolbar Buttons](#)
- [Context Menu](#)

Toolbar buttons

Item	Tooltip and shortcut	Description
	Rerun Ctrl+F5Command F5	Rerun the last search.
	Pin	When this button is pressed, the current tab will not be overwritten; instead, the results of the next command will be displayed in a new tab.
	Close Ctrl+Shift+F4Command Shift F4	Close the current tab. This button is not available in Replace in Path and Refactoring Preview dialogs.
	Recent find usages Ctrl+ECommand E	Show the list of recent find usages. Clicking the desired entry opens results of the selected search for usages.
	Expand all Ctrl+Add Command Add or Ctrl+Equals Command Equals Collapse all Ctrl+Subtract Command Subtract or Ctrl+Minus Command Minus	Click this button to have all nodes expanded or collapsed.
	Previous occurrence Ctrl+Alt+UpCommand Alt Up	Navigate to the previous element in the tab of results.
	Next occurrence Ctrl+Alt+DownCommand Alt Down	Navigate to the next element in the tab of results.
	Autoscroll to source	Toggle the Autoscroll to source mode. When the button is pressed, every time the node is focused, the corresponding line of source code is highlighted in the editor.
	Export to Text File Alt+OCommand O	Save the contents of the current result tab. In the Export preview dialog, specify the target file or copy information to the Clipboard. Before saving, you can also modify the information to be saved.
	F1F1	Click this button to show reference.
	Group by usage type Ctrl+TCtrl T	If this button is pressed, the search results are grouped by the following categories: <ul style="list-style-type: none"> • comments • unclassified usage not related to any of the categories
	Group by module	If this button is pressed, the found usages show under the corresponding module or library node.

Ctrl+DCommand D

Tip

This type of grouping is helpful, when a package is split between several modules.

-  **Group by package**
Ctrl+PCommand P
If this button is pressed, all the usages found are displayed under their respective packages.
-  **Group by file structure**
Ctrl+M
If this button is pressed, the found usages show under the corresponding method nodes.
-  **Merge usages from the same line**
Ctrl+F Command F or
Alt+F3 Alt F3
If this button is pressed, the duplicate lines are merged.
-  **Show read access**
Ctrl+RCommand R
If this button is pressed, the search results include references to the read access methods.
-  **Show write access**
Ctrl+WCommand W
If this button is pressed, the search results include references to the write access methods.
-  **Show import statements**
Ctrl+ICommand I
If this button is pressed, the search results include the usages in the import statements.

Tip

This button is available for the Search for Usages only.

-  **Preview usages**
If this button is pressed, the preview of the selected usage appears in the Preview pane of the tool window. If the button is not pressed, the Preview pane is hidden.
-  **Sort members alphabetically**
Click this button to have all members sorted alphabetically. Otherwise, members are sorted in the order they are declared.

Context menu commands

Item	Shortcut	Description
Jump to Source	F4F4	Navigate to the selected usage in the source code.
Exclude	DeleteDelete	Exclude the selected item from the list of results, with all nested subelements. The excluded nodes are marked with the strikethrough. If you perform Do Replace All or Do Refactor All actions, the excluded elements are skipped.
Include	InsertInsert	Include previously excluded items to the list of results. All nested subelements are included too.
Add to Favorites		Add selected node to the Favorites list.
Recent Find Usages	Ctrl+ECommand E	Show the list of recent find usages. Clicking the desired entry opens results of the selected search for usages.
Local History		Show Local History submenu for the selected search result. Refer to the Local Version Control procedures for details.

See Also

Procedures:

- [Finding Usages in Project](#)
- [Finding and Replacing Text in Project](#)
- [Viewing Usages of a Symbol](#)
- [Refactoring Source Code](#)
- [Structural Search and Replace](#)

Getting Started:

- [PhpStorm Tool Windows](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Run Tool Window

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Run | Run

View | Tool Windows | Run

Alt+4Meta 4

Run | Run

View | Tool Windows | Run

Alt+4Meta 4

The Run tool window displays output generated by your application. If you are running multiple applications, each one displays its console output and logs in a tab named after the [run/debug](#)

[configuration](#) applied.

Note

The appearance of each tab depends on the type of the application being run and can include additional toolboxes and panes.

- [Deployment console](#).
- [Test Runner](#).

The main toolbar of the Run tool window lets you rerun, stop, pause, or terminate an application. The following table contains descriptions of the buttons that are common for most applications.

Run toolbar

Item	Tooltip and shortcut	Description
	Rerun Ctrl+F5Command F5	Click this button to rerun the current process. Note The process reruns always in the same console regardless of whether this console is pinned or not.
	Pause Output	Click this button to have the process output paused.
	Dump Threads Ctrl+BreakCtrl Break	Click this button to dump all threads of the current process showing their status in the Sun format.
	Exit	Click this button to terminate the current process gracefully using in-process internal mechanisms. Note This button is available for GWT applications only.
	Stop Ctrl+F2Command F2	Click this button to terminate the current process externally by means of the standard <code>shutdown</code> script.
	Layout	Click this button to choose a layout of the toolbars by selecting the desired option from the pop-up menu.
	Pin	When this button is pressed, the current tab will not be overwritten; instead, the results of the next command will be displayed in a new tab.
	Ctrl+Shift+F4Command Shift F4	Click this button to close the selected tab of the Run tool window and terminate the current process.
	F1F1	Click this button to show reference.
	Up the Stack Trace Ctrl+Alt+UpCtrl Alt Up	Click this button to navigate up in the stack trace and have the cursor jump to the corresponding location in the source code.
	Down the Stack Trace Ctrl+Alt+DownCtrl Alt Down	Click this button to navigate down in the stack trace and have the cursor jump to the corresponding location in the source code.
	Use Soft Wraps	Click this button to toggle the soft wrap mode of the output.
	Scroll to the end	Click this button to navigate to the bottom of the stack trace and have the cursor jump to the corresponding location in the source code.

See Also

Concepts:

- [Running and Debugging](#)

Procedures:

- [Running Applications](#)
- [Testing](#)

Reference:

- [Manipulating the Tool Windows](#)

Getting Started:

- [PhpStorm Tool Windows](#)
- [Viewing Modes](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

```
ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js'; var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s);})();
```



Test Runner Tab

Previous | Next | [See Also](#) | [Comments](#) | Shortcuts: Mac

The **Test Runner** tab opens in the [Run](#) tool window when a testing session begins, and features the same [toolbar buttons](#).



[Click thumbnail to view larger image.](#)

1. The **progress bar** shows the percentage of tests executed so far.
2. The color of the **status bar** indicates whether the tests have passed successfully. If at least one of the tests fails, the status bar turns red.
3. The **left-hand pane** shows the tree view of all tests within the current run/debug configuration.
 - The root node represents the test case selected to run.
 - The nested nodes represent the hierarchy of test suites and test cases.
 - The leaf nodes represent the individual tests.

The status of each test is indicated by an [icon](#). Double-click a node to open the respective test class or test method in the editor.

4. The [toolbar](#) provides controls that enable you to monitor the tests and analyze results. Some of the commands are duplicated on the [context menus](#) of the test tree nodes and the entries in the [Statistics](#) pane.
5. The [Output](#) pane shows the output of the current test suit.
6. The [Statistics](#) pane shows the list of executed tests with the time used for executing each test and the test result.

Run toolbar

Item	Tooltip and Shortcut	Description
	Rerun Ctrl+F5Command F5	Click this button to rerun the current process. Note The process reruns always in the same console regardless of whether this console is pinned or not.
	Pause Output	Click this button to have the process output paused.
	Dump Threads Ctrl+BreakCtrl Break	Click this button to dump all threads of the current process showing their status in the Sun format.
	Exit	Click this button to terminate the current process gracefully using in-process internal mechanisms. Note This button is available for GWT applications only.
	Stop Ctrl+F2Command F2	Click this button to terminate the current process externally by means of the standard mechanisms.
	Layout	Click this button to choose a layout of the toolbars by selecting the desired option from the pop-up menu.
	Pin	When this button is pressed, the current tab will not be overwritten; instead, the results of the next command will be displayed in a new tab.
	Ctrl+Shift+F4Command Shift F4	Click this button to close the selected tab of the Run tool window and terminate the current process.
	F1F1	Click this button to show reference.

Testing toolbar

Item	Tooltip and Shortcut	Description
	Hide Passed	Press this button, to have PhpStorm show only the failed tests or the tests that encountered problems in the left-hand pane. If this button is not pressed, all tests are displayed in the tree view.
	Expand All/Collapse All Ctrl+Add Command Add or Ctrl+Equals Command EqualsCtrl+Subtract Command Subtract or Ctrl+Minus Command Minus	Click these buttons to have all nodes in the tree view of tests expanded. Note These buttons are only available if the tested application contains more than test case.
	Previous Failed Test Ctrl+Alt+UpCtrl Alt Up	Click this button to navigate to the previous failed test.
	Next Failed Test Ctrl+Alt+DownCtrl Alt Down	Click this button to navigate to the next failed test.
	Export Test Results	Click this button to have the results of the selected test saved in a file.
	Click this cog button to access the context menu with the following options:	<ul style="list-style-type: none"> • Track Running Test - select this option to monitor execution of the current test. If a test suite contains multiple tests, the tree view of tests expands to show sequential test methods, as they are executed. • Select First Failed Test When Finished - select this option to have the first failed test automatically selected in the tree view upon completing the tests. • Scroll to Stacktrace - select this option to have the console scroll to the beginning of the trace of the last failed test. If the option is not selected and you click the root node (the test package) in the tree view, the console will show the very beginning of the test.

Tip

This option is helpful when a test package contains multiple test classes and test methods. If some of the tests fail, you can scroll in the console to the beginning of a stack trace of an exception or assertion.

- **Autoscroll to Source** - select this option to have the currently selected test in the tree view synchronized with the editor automatically.
- **Open Source at Exception** - use this option to explore the results of a test that fails as an error, throwing an uncaught exception. If this option is selected and you double-click the failed test class or method in the tree view, the respective test class or method will open in the editor, with the caret placed at the line that caused the problem.
- **Show Statistics** - select this option to have the [Statistics](#) pane opened, displaying a list of executed tests with the time used for executing each test and the test result.

Test status icons

Icon	Description
	Test error.
	Test failed.
	Test ignored.
	Test in progress.
	Test passed successfully.
	Test paused.
	Test terminated.
	Test not run.

Output pane

This pane shows output of each test, generated at runtime, including all the messages sent to the output stream, and the error messages. The following table shows the toolbar buttons and context menu commands available for the Output pane.

Item	Keyboard Shortcut	Description
	Up the Stack Trace Ctrl+Alt+UpCtrl Alt Up	Click this button to navigate up in the stack trace and have the cursor jump to the corresponding location in the source code.
	Down the Stack Trace Ctrl+Alt+DownCtrl Alt Down	Click this button to navigate down in the stack trace and have the cursor jump to the corresponding location in the source code.
	Use Soft Wraps	Click this button to toggle the soft wrap mode of the output.
	Scroll to the end	Click this button to navigate to the bottom of the stack trace and have the cursor jump to the corresponding location in the source code.
Clear All		Choose this item on the context menu to have all messages for the selected test deleted.
Copy Content	Ctrl+C Command C or Ctrl+Insert Command Insert	Choose this item on the context menu to have the current contents of the Output pane placed to the Clipboard.
Compare with Clipboard		Choose this item on the context menu to invoke the Differences Viewer which shows the current contents of the Clipboard in the left-hand pane and the contents of the Output pane for the selected test in the right-hand pane. pane.

Statistics pane

This pane shows how long it took to execute a test suite and the number of passed and failed tests within it. You can have information in columns shown in the ascending or descending order by clicking the column headers.

To view statistics on each test separately, double click the test suite which contains the desired test. The pane shows a list of all the tests in the selected suit.

The [Context menu](#) commands are the same as for the tree view of tests.

Column name	Description
Test	Name of the test class or test method.
Time elapsed	The number of seconds spent to to execute this test.
Results	This column shows summary information about the test execution results. For individual tests, the column shows the <code>pass</code> or <code>fail</code> status. For test case or test suites, the column shows the number of failed and passed tests within it.

Context menu commands

Command	Keyboard shortcut	Description
Jump to Source	F4F4	Choose this command to move the focus to the editor, to the definition of a test class, or test method.

See Also

Procedures:

- [Manipulating the Tool Windows](#)
- [Testing](#)
- [Performing Tests](#)
- [Rerunning Tests](#)

Reference:

- [Run Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Hierarchy Tool Window

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

View | Tool Windows | Hierarchy

Alt+8Meta 8

View | Tool Windows | Hierarchy

Alt+8Meta 8

Prerequisite

Warning

This tool window becomes available only on [building a hierarchy](#). Unless a hierarchy is created, the tool window is not available. Once created, the Hierarchy tool window helps you analyze and navigate through the [hierarchies](#) of classes .

This tool window is not automatically updated as you navigate through the source code or change editor tab. The tool window shows the results of the latest hierarchy command and is updated when you run the next hierarchy command, unless a [tab is pinned](#).

Toolbar buttons

Item	Description	Available In
	When this button is pressed, the hierarchical tree shows both the parent and child classes of the selected class, which is marked with an arrow in the tree of results.	Class hierarchies
	Depending on the hierarchy type: <ul style="list-style-type: none"> • For class hierarchies - when this button is pressed, the tool window shows all classes that extend the selected class. • For call hierarchies - when this button is pressed, the tool window shows the callees of the selected method. 	Class hierarchies Call hierarchies
	Depending on the hierarchy type: <ul style="list-style-type: none"> • For class hierarchies - when this button is pressed, the tool window shows the hierarchy of each supertype of the current class. • For call hierarchies - when this button is pressed, the tool window shows the callers of the selected method. 	Class hierarchies Call hierarchies
	When this button is pressed, the elements within a tree are sorted alphabetically.	All hierarchies
Scope	Use this drop-down list to limit the scope of the current hierarchy: <ul style="list-style-type: none"> • Project - PhpStorm traces usages of the method across the project. • Test - PhpStorm traces usages of the method across the test classes. • All - PhpStorm traces usages of the method across the project and the libraries. • This class - the scope is limited to the current class. 	Call hierarchies

Note

In a method hierarchy, [the tree-views](#) of the following classes are available:

- - where this method is defined.
- - where this method is not defined, but defined in the superclass.
- - where this method should be defined, because the class is not abstract.

	If you made any changes of the classes or the class structure, they become visible in the Hierarchy tool window only after you press this button.	All hierarchies
	Toggle the Autoscroll to source mode. When the button is pressed, every time the node is focused, the corresponding line of source code is highlighted in the editor.	All hierarchies
	When this button is pressed, the current tab will not be overwritten; instead, the results of the next command will be displayed in a new tab.	All hierarchies
	Click this button to close the selected tab of results.	All hierarchies
	Click this button to show reference page.	All hierarchies

See Also

Procedures:

- [Viewing Structure and Hierarchy of the Source Code](#)

Getting Started:

- [PhpStorm Tool Windows](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Inspection Tool Window

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

[View](#) | [Tool Windows](#) | [Inspection](#)

[View](#) | [Tool Windows](#) | [Inspection](#)

The Inspection tool window displays inspection results in separate tabs. The left pane of each tab shows a tree view of inspections, for which the problems are reported; the right pane shows summary information on each inspection.

In this topic:

- [Toolbar Buttons](#)
- [Context Menu Commands](#)
- [Inspection Results Report](#)

Toolbar buttons

Item	Shortcut	Description
	Ctrl+F5Command F5	Click this button to execute inspection and show results in the same tab.
	Ctrl+Shift+F4Command Shift F4	Click this button to close active tab.
	Ctrl+Add Command Add or Ctrl+Equals Command Equals	Expand all nodes
	Ctrl+Subtract Command Subtract or Ctrl+Minus Command Minus	Collapse all nodes
	Ctrl+Alt+DownCommand Alt Down	Navigate to the next item.
	Ctrl+Alt+UpCommand Alt Up	Navigate to the previous item.
		Toggle the Autoscroll to source mode. When the button is pressed, every time the node is focused, the corresponding line of source code is highlighted in the editor.
		Click this button to export inspection results in XML or HTML format .
	F1F1	Show reference.
		If this button is pressed, the problems reported by the code inspection are grouped into Errors and Warnings. Otherwise, the problems are grouped by inspections.
		If this button is pressed, the items show under the corresponding directory nodes.
		When this button is pressed, the resolved and excluded items to not display in the tool window.
		If you run inspections several times in the same tab, press this button to filter differences.
		If you run inspections several times in the same tab, press this button to hide items that did not change since the previous inspection.
		Click this button to edit the current inspection settings .
	Alt+EnterAlt Enter	Click this button to resolve the problem for the selected inspection item, by choosing one of the available quick fixes from the list.
		Click this button to disable, suppress or run the selected inspection.

Context menu commands

Item	Shortcut	Description
Jump to Source	F4F4	Open in the editor the file that contains the selected problem, and place the caret at the beginning of the respective code fragment.
Find Usages	Alt+F7Alt F7	Search for usages of the selected file.
Exclude	DeleteDelete	Exclude the selected items from further examination. Excluded nodes are marked with a strikethrough. If the filter button is pressed, the excluded nodes are hidden.
Include	InsertInsert	Include previously excluded items to the list of results. All nested subelements are included too.
	Alt+EnterAlt Enter	Accept one of the suggested solutions to fix the problem.
Edit Settings		Change settings for the selected inspection, or group of inspections, in the Errors dialog.
Disable inspection		Disable alerts for the selected inspection in the active tab of results. If the filter button is pressed, the nodes for disabled inspections are hidden. You can also disable an inspection in general.
Run inspection on		Rerun the selected inspection and display results in a new tab.
Suppress problem/		Suppress inspection for the selected problem in general, or for the selected class.

Suppress problem for class

Local history [Perform local history actions.](#)

Inspection results report

The inspection results report appears in the right pane of the results tab, when an inspection node is selected in the left pane. The report includes the following sections:

- **Name:** this is the name of the class or member where the problem has been detected.
- **Location:** this is a hyperlink to the class, where the problem has been detected.
- **Problem synopsis:** shows a brief description of the problem.
- **Problem resolution:** this optional field provides a list of possible solutions. Clicking a hyperlink invokes the corresponding action. If no hyperlinks are present, it means that you have to fix the problem yourself.

See Also

Concepts:

- [Code Inspection](#)

Reference:

- [Inspections](#)

Getting Started:

- [PhpStorm Tool Windows](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Messages Tool Window

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

View | Tool Windows | Messages

Alt+0Meta 0

View | Tool Windows | Messages

Alt+0Meta 0

PhpStorm parses the output and displays it in a convenient format in the Messages window, letting you navigate to the source of the problem whenever possible.

Toolbar buttons

Item	Tooltip and shortcut	Description
	Close	Click this button to close the messages console.
	Previous/Next Message Ctrl+Alt+UpCommand Alt Up Ctrl+Alt+DownCommand Alt Down	Navigate to the previous/next message.
	Export to Text File Alt+OCommand O	Save the current console contents. In the Export dialog, specify the target file or copy information to the Clipboard. Before saving, you can also edit the information to be saved.
	Expand all Ctrl+Add Command Add or Ctrl+Equals Command Equals Collapse all Ctrl+Subtract Command Subtract or Ctrl+Minus Command Minus	Click this button to have all nodes expanded or collapsed.
	Hide warnings	When this button is pressed, the tree of messages shows error messages only.
	Autoscroll to source	If this button is pressed, the file that contains the selected error automatically opens in the editor, with the caret at the appropriate line.

Results context menu

Item	Shortcut	Description
Jump to source	F4F4	Navigate to the selected item in the editor.
Copy	Ctrl+C Command C or Ctrl+Insert Command Insert	Take the line at caret to the clipboard.

See Also

Concepts:

- [Compilation Types](#)

Procedures:

- [Configuring Compiler Settings](#)
- [Reviewing Compilation and Build Results](#)

Reference:

- [Compiler](#)

Getting Started:

- [PhpStorm Tool Windows](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Structure Tool Window

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

View | Tool Windows | Structure

Alt+7Meta 7

View | Tool Windows | Structure

Alt+7Meta 7

The **Structure** tool window displays the structure of a file currently opened in the editor, or selected in the Project tool window. For diagrams, this tool window shows the diagram preview.

The set of toolbar buttons depends on the file type.

In this section:

- [Toolbar](#)
- [Tree View Members](#)

Toolbar

Item	Tooltip and shortcut	Description	Available for
	PHP	Use this view to have the hierarchy of the PHP elements displayed.	PHP files and classes
	HTML View	Use this view to have the hierarchy of the PHP elements displayed.	HTML files PHP classes and files with HTML code blocks
	Sort by Visibility	Click this button to have the items sorted by their visibility in the following order: public - protected - package local - private.	Java classes, PHP classes
	Sort Alphabetically	Click this button to have the elements within a class sorted alphabetically.	All file types
	Show Inherited	Click this button to display all the methods and fields inherited by the current class and accessible from it. The inherited members are displayed gray to distinguish them from the members defined in the current class.	JavaScript, CoffeeScript
	Show Includes	Click this button to have all files included through <code>include</code> or <code>require</code> statements shown in the tree.	PHP classes
	Show Constants	Click this button to have constants shown in the tree.	PHP
	Expand All Ctrl+Add Command Add or Ctrl+Equals Command Equals	Expand all nodes.	All file types
	Collapse All Ctrl+Subtract Command Subtract or Ctrl+Minus Command Minus	Collapse all nodes.	All file types
	Autoscroll to Source	Toggle this button to enable automatic navigation to the line of source code that corresponds to the selected node when the focus switches to the editor.	All file types
	Autoscroll from Source	Toggle this button to have PhpStorm automatically move the focus in the Structure tool window to the node that corresponds to the code where the cursor is currently positioned in the editor.	All file types
	Help	Click this button to show reference.	All file types

Tree view members

Tip

Refer to the section [Symbols](#) for the information about the member icons.

See Also

Procedures:

- [Viewing Structure of a Source File](#)
- [Navigating with Structure Views](#)

Reference:

- [Symbols](#)

Getting Started:

- [PhpStorm Tool Windows](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

TODO Tool Window

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

View | Tool Windows | TODO

Alt+6Meta 6

View | Tool Windows | TODO

Alt+6Meta 6

PhpStorm scans your project for comments in the source code that match the TODO patterns defined in the [TODO dialog](#) and displays results in the TODO tool window.

The TODO tool window consists of three tabs that show the TODO items for the whole project, the current file, or the default changelist. This tool window helps you view, sort and group the TODO items in a convenient way, navigate to the source code and keep fixes under the version control.

In this section:

- [Toolbar Buttons](#)
- [Context Menu](#)

Toolbar buttons

Item	Shortcut	Description
	Ctrl+Alt+UpCommand Alt Up	Navigate to the previous TODO item.
	Ctrl+Alt+DownCommand Alt Down	Navigate to the next TODO item.
	F1F1	Click this button to show reference.
	Expand all Ctrl+Add Command Add or Ctrl+Equals Command Equals Collapse all Ctrl+Subtract Command Subtract or Ctrl+Minus Command Minus	Click this button to have all nodes expanded or collapsed.
		Toggle the Autoscroll to source mode. When the button is pressed, every time the node is focused, the corresponding line of source code is highlighted in the editor.
		Click this button to select the desired filter from the list, or invoke the TODO dialog and edit the list of filters as required.

Tip

The following buttons are available in the **Project tab** only.

	Ctrl+DCommand D	If this button is pressed, the TODO items show under the corresponding module or library node.
	Ctrl+PMeta P	If this button is pressed, the TODO items show under the corresponding packages.
	Ctrl+F Command F or Alt+F3 Alt F3	If this button is pressed, the TODO items show as a flat list. Thus, if a package is somewhere deep within your project, you do not need to dig deep into the hierarchy.

Context menu commands

Item	Keyboard Shortcut	Description
Jump to Source	F4F4	Navigate to the selected usage in the source code.
Local History		Show Local History submenu for the selected search result. Refer to the Local Version Control procedures for details.
<VCS>		Show menu of the VCS, associated with the directory. See Version Control Procedures and Reference for details.

See Also

Procedures:

- [Using TODO Lists](#)

Reference:

- [TODO](#)
- [Regular Expression Syntax Reference](#)

Getting Started:

- [PhpStorm Tool Windows](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Version Control Tool Window

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

[View](#) | [Tool Windows](#) | [Version Control](#)

[View](#) | [Tool Windows](#) | [Version Control](#)

The tool window accommodates various views that display VCS-related information, such as:

- History of files.
- Local information update.
- VCS-specific commands and messages with the results of executing them.

In this part:

- [History Tab](#)
- [Update Info Tab](#)
- [Integrate to Branch Info View](#)
- [Console Tab](#)

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Checking in Files](#)
- [Using Subversion Integration](#)

Reference:

- [Version Control Reference](#)

Getting Started:

- [PhpStorm Tool Windows](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

History Tab

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

History tab is added to the Version Control tool window on invoking the **Show History** command of your particular VCS. So doing, a new tab is created for each file or directory, with the name `File <name> History`. The set of toolbar buttons is slightly different for the different version control systems.

Item	Description
	Click this button to compare the selected revision of a file with its local copy in the Differences viewer.
	Click this button to create a patch from the selected revision.
	Click this button to retrieve the selected revision. If the local copy has already been modified, PhpStorm prompts to overwrite the local version, or cancel operation.
	Click this button to open the selected revision of a file in the editor with annotations.
	Click this button to show Perforce branches.
	Click this button to show CVS branches.
	Click this button to display the list of all revisions committed in the same changelist as the selected revision of a file.
	Click this button to refresh the current information.
	Click this button to show reference information.

Note

The revisions of a file are shown in the regular font. The revision that currently exists in your working copy, is shown in **bold font**.

See Also

Concepts:

- [Changelist](#)

Procedures:

- [Viewing Changes History for a File or Selection](#)

Reference:

- [Version Control Reference](#)
- [Changes Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Update Info Tab

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

[View](#) | [Tool Windows](#) | [Version Control](#)

This tab is available when [local information is synchronized](#) to the server.

Item	Shortcut	Description
		When this button is pressed, the update information within nodes is grouped by packages.
		When this button is pressed, the update information within nodes is grouped by changelists, and by the day the changelist has been committed. The Update Info tab is divided into two panes: the left pane shows the changelists, grouped by the check-in date, and the right pane shows the list of changed files .
	Expand all Ctrl+Add Command Add or Ctrl+Equals Command Equals	Click this button to have all nodes expanded or collapsed.
	Collapse all Ctrl+Subtract Command Subtract or Ctrl+Minus Command Minus	
	Ctrl+DCommand D	Click this button to open the Differences Viewer , where you can compare the local copies of all the project files one after another with their updates from the server. Use the buttons Compare Next File and Compare Previous File to scroll through the list of updated files.
		Click this button to close the tab.
	F1F1	Click this button to show reference page.
The controls of the Changed files pane appear when the button is pressed.		
	Ctrl+DCommand D	Click this button to show differences between the selected and the previous revision for the selected file in the Changes Files pane.
		Click this button to show differences between the selected revision of a file, and its current local copy.
	F4F4	Click this button to open local copy of the selected file for editing.
		Click this button to open repository version of the selected file.
		Click this button to revert selected changes.
		This button is only available for he files under Subversion, and enables viewing differences in properties of the selected file.
		Click this button to open the History view of the selected file in the History Tab of the Version Control tool window.
	Ctrl+PMeta P	When the button is not pressed, the pane shows a flat list of files. When the button is pressed, the pane shows files in their respective packages. In the latter case, expand and collapse buttons appear in the toolbar.
	Ctrl+ACommand A	Click this button to select all files in the Changed Files pane.

See Also

Concepts:

- [Changelist](#)

Procedures:

- [Updating Local Information](#)

Reference:

- [Version Control Reference](#)
- [Changes Tool Window](#)
- [Differences Viewer](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Integrate to Branch Info View

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

[View](#) | [Tool Windows](#) | [Version Control](#)

This view is available after running integration with the **Run status after update** setting specified. The view displays a list of files or packages affected by the latest integration. The items are displayed under the following nodes:

- Modified
- Merged
- Not in repository
- Locally added

Item	Tooltip and shortcut	Description
		Click this button, to group information within nodes by packages. If the button is released, files are presented in plain lists.
	Ctrl+Add Command Add or Ctrl+Equals Command Equals	Click this button to expand all nodes.
	Ctrl+Subtract Command Subtract or Ctrl+Minus Command Minus	Click this button to collapse all nodes.
		Click this button to close the view.
	F1F1	Click this button to show the corresponding reference page.

See Also

Concepts:

- [Supported Version Control Systems](#)

Procedures:

- [Using Subversion Integration](#)
- [Integrating SVN Projects or Directories](#)

Reference:

- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Console Tab

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

[View](#) | [Tool Windows](#) | [Version Control](#)

The tab displays:

- Version control related commands generated based on the settings you specify through the PhpStorm interface.
- Results of executing version control related commands.



See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Using Git Integration](#)

Reference:

- [Version Control Reference](#)
- [Version Control Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Remote Host Tool Window

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Some features described here are available in PHP Developer edition only.

[Tools](#) | [Deployment](#) | [Browse Remote Host](#)

[View](#) | [Tool Windows](#) | [Remote Host](#)

[Tools](#) | [Deployment](#) | [Browse Remote Host](#)

[View](#) | [Tool Windows](#) | [Remote Host](#)

Use this tool window to view the folder structure of the target FTP/SFTP servers and the data uploaded to them.

Note

- [View](#) | [Tool Windows](#) | [Remote Host](#) - the tool window can be accessed this way after you have opened it using the [Tools](#) | [Deployment](#) | [Browse Remote Host](#) command.
- The tool window is available only when the **Remote Hosts Access** bundled plugin is enabled. The plugin is active by default. If not, activate it in the [Plugins](#) page of the [Settings](#) dialog box.

Item Tooltip and Shortcut Description

Remote host		From this drop-down list, select the desired remote host configuration.
	Shift+Enter Enter	Click the browse button to add a new server .
	Collapse All Ctrl+Subtract Command Subtract or Ctrl+Minus Command Minus	Click this button to have all nodes in the view collapsed.
	Close Ctrl+Shift+F4 Command Shift F4	Click this button to close the tool window.
		Click this button to configure the layout of the server view by selecting the corresponding options in the list that opens: <ul style="list-style-type: none"> • Show Size - select this option to have the size indicated next to each item. • Show Date - select this option to have the deployment date indicated next to each item. • Show Permissions - select this option to have the permissions for each item indicated. • Highlight Symlinks - select this option to have symbolic links highlighted.

See Also

Language and Framework-Specific Guidelines:

- [Remote Hosts](#)

Reference:

- [Deployment](#)
- [Run/Debug Configurations](#)

Getting Started:

- [PhpStorm Tool Windows](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Phing Build Tool Window

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

[PhpStorm editor](#) | [Context menu of a Phing build file](#) | [Add as Phing build file](#)

[Project Tool Window](#) | [Context menu of a Phing build file](#) | [Add as Phing build file](#)

[View](#) | [Tool Windows](#) | [Phing Build](#)

[PhpStorm editor](#) | [Context menu of a Phing build file](#) | [Add as Phing build file](#)

Project Tool Window | Context menu of a Phing build file | Add as Phing build file

View | Tool Windows | Phing Build

Use this tool window to:

- Specify the location of the `phing.bat` file.
- Configure a list of Phing build files for packaging, deploying, or testing PHP applications.
- Appoint targets to be executed before running or debugging according to specific configurations.
- Run entire build files and specific build targets.

The functionality of the Phing tool window is available through the [toolbar buttons](#) and the [context menu](#) of a build file or target.

Note

- **View | Tool Windows | Phing Build** - the tool window can be accessed this way after you have opened it through the context menu of a Phing build file in the editor or in the Project tool window.
- The tool window is available only when the **Phing Support** bundled plugin is enabled. The plugin is active by default. If not, activate it in the [Plugins](#) page of the [Settings](#) dialog box.

Toolbar

Item	Tooltip and shortcut	Description
	Add	Click this button add a new script to the list. In the Select Phing Build Script dialog box, that opens, choose the desired file.
	Remove	Click this button to remove the selected build file from the list.
	Run	Click this button to run the selected build file or target.
	Expand all Ctrl+Add Command Add or Ctrl+Equals Command Equals	Click this button to have PhpStorm show a tree of targets defined in the selected build file.
	Collapse all Ctrl+Subtract Command Subtract or Ctrl+Minus Command Minus	Click this button to have PhpStorm collapse all the targets in the selected build file.
	Properties	Click this button to have the Phing Properties dialog box opened. In this dialog box, specify the location of the <code>phing.bat</code> file. If necessary, configure the build procedure and the format of the output by specifying command line arguments in the Command Line Options text box. Tip If the set of arguments is too large to fit in the text box, click the  button or press <code>Shift+EnterShift Enter</code> and type the desired arguments in the Command Line Options dialog box, that opens. Note The button is only available if the list of build files is not empty.

Context menu

Item	Description	Available for
Run Build 	Choose this option to have the selected build file executed.	Build files
Run Target 	Choose this option to have the selected build target executed.	Build targets
Jump to Source	Choose this option to navigate to the source code of the selected build file or to the definition of the selected target.	Build files Build targets
Remove	Choose this option to remove the selected build file from the list.	Build files
Before Run/Debug	Choose this option to have the selected build target always executed before running or debugging the project according to specific run/debug configurations. In the Execute Target Before Run/Debug dialog box that opens, select the configurations before which you want the target executed.	Build targets
Properties 	Click this button to have the Phing Properties dialog box opened. In this dialog box, specify the location of the <code>phing.bat</code> file. If necessary, configure the build procedure and the format of the output by specifying command line arguments in the Command Line Options text box. Tip If the set of arguments is too large to fit in the text box, click the  button or press <code>Shift+EnterShift Enter</code> and type the desired arguments in the Command Line Options dialog box, that opens. Note The button is only available if the list of build files is not empty.	Build files build targets

See Also

Procedures:

- [Using Phing](#)
- [PHP-Specific Guidelines](#)

Getting Started:

- [PhpStorm Tool Windows](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
 - <http://youtrack.jetbrains.net/issues/WI> 
-

Version Control Reference

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

This part provides miscellaneous information, related to common version control operations, and to VCS integrations:

- [CVS Reference](#)
- [Git Reference](#)
- [Mercurial Reference](#)
- [Subversion Reference](#)
- [Perforce Reference](#)
- [Apply Patch Dialog](#)
- [Enable Version Control Integration Dialog](#)
- [Commit Changes Dialog](#)
- [Configure Ignored Files Dialog](#)
- [Create Patch Dialog](#)
- [File Status Highlights](#)
- [New Changelist Dialog](#)
- [Revert Changes Dialog](#)
- [Select Target Changelist Dialog](#)
- [Shelve Changes Dialog](#)
- [Show History for File / Selection Dialog](#)
- [Show History for Folder Dialog](#)
- [Unshelve Changes Dialog](#)
- [Visual SourceSafe Options Dialog](#)

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Version Control with PhpStorm](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
 - <http://youtrack.jetbrains.net/issues/WI> 
-

CVS Reference

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

In this part:

- [CVS Global Settings Dialog](#)
- [CVS Options Dialog](#)
- [CVS Root Dialog](#)
- [CVS Tool Window](#)

See Also

Concepts:

- [Supported Version Control Systems](#)

Procedures:

- [Using CVS Integration](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
 - <http://youtrack.jetbrains.net/issues/WI> 
-

CVS Global Settings Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

VCS | CVS | Global Settings

File | Settings | Version Control | VCSs | CVS - Global Settings for Windows and Linux
 PhpStorm | Preferences | Version Control | VCSs | CVS - Global Settings for Mac OS

Use this dialog box to set up CVS options at the global level. The dialog is available for files and directories that are under CVS control.

Item	Description
Charset	From this drop-down list, select the character set to be used.
Use gzip compression	Select this check box to apply <code>gzip</code> compression.
Password file	In this text box, specify the fully qualified path to the <code>.cvspass</code> file. Click the  button to open the Select Path dialog where you can navigate to the desired location.
Connection timeout	In this text box, type the timeout value in seconds.
Send environment variable to server	Select this check box to have CVS-related environment variables sent to the server.
Log CVS client/server output to <code>cv.s.log</code> file	Select this check box to enable logging and have the <code>cv.s.log</code> log file stored in the <code>log</code> directory of your PhpStorm installation.

See Also

Procedures:

- [Configuring Global CVS Settings](#)

Reference:

- [CVS Roots Dialog](#)
- [CVS](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

CVS Roots Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

VCS | CVS | Configure CVS Roots

Use this dialog box to set up CVS roots. The dialog box is available for files and directories that are under CVS version control.

Common options

Item	Description
	Click this button to configure a new CVS root.
	Click this button to remove the selected CVS root configuration from the list.
	Click this button to create a copy of the selected CVS root.
Global Settings	Click this button to open the Global CVS Settings dialog box where you can set up CVS options at the global level.
CVS root	In this text box, specify the CVS repository string according to the following syntax: <code>[[:method:]] [[:user]] [[:password]]@hostname[:[:port]]/path/to/repository.</code>
Tip	Obtain the valid string from your system administrator or click the Edit by Field button to open the Configure CVS Root Field by Field dialog box where you can specify the mandatory connection parameters and have PhpStorm assemble them into a correct repository string.
Edit by Field	Click this button to open the Configure CVS Root Field by Field dialog box where you can specify the mandatory connection parameters and have them assembled into a CVS root string automatically.
Use version	Use this section to specify the revision you want to synchronize your local working copy with. The available options are: <ul style="list-style-type: none"> • HEAD revision: this option is suggested by default. • By tag: select this option to access the revision with a specific tag. Type the desired tag in the text box or click the Browse button  and select the desired tag from the list. The list shows all the tags available on the CVS server according to the specified CVS root. • By date: select this option to access the revision with a specific date and time stamp. Type the end date and time in the format <code>dd:mm:yy hh:mm:ss</code> or click the Browse button  and select the desired date from the calendar.
Note	This date and time are passed to the server with the GMT parameter.

Tip

The controls in the area are available only after the **CVS root** text box is filled in with valid data.

Warning

If you perform update or checkout from the CVS repository with the **By tag** or **By date** option selected, the resulting working copy will be permanently restricted to the specified tag or date, until you force the update operation to reset this [sticky data](#).

Test connection

Click this button to check that the specified settings ensure successful connection to the CVS server.

Additional connection settings

In this area, specify additional settings to flexibly configure connection to the CVS server. The contents of the area depends on the [connection method](#) set in the **CVS root** text box.

- [pserver](#)
- [ext](#)
- [ssh](#)
- [local](#)

Pserver

Warning

The settings specified in this area affect all CVS roots that use the **pserver** connection method.

Item	Description
Password	In this text box, type the fully qualified path to the <code>.cvspass</code> file. Click the Browse button  to open the Select Path dialog box and navigate to the desired location.
Connection timeout	In this text box, type the connection timeout in seconds.
Proxy Settings	See the Proxy Settings section below.

Ext

Item	Description
Use internal <code>ssh</code> implementation	Select this check box to access the ssh area , where you can specify the SSH version to use, the port to listen to, and configure your private key and password. Clear this check box to access the Ext Protocol Settings area with the following controls available: <ul style="list-style-type: none"> • Path to external <code>rsh</code>: in this text box, specify the location of the external <code>rsh</code>. If necessary, click the Browse button  to open the Select Path dialog box. • Path to private key file: in this text box, specify the location of the file with your private <code>ssh</code> key. If necessary, click the Browse button  to open the Select Path dialog box. • Additional parameters: in this text box, specify additional connection parameters.

Ssh

Note

This area is also available when you have specified the [ext](#) connection method and selected the **Use internal ssh implementation** check box.

Item	Description
SSH version	In this area, specify the SSH version to use. The available options are: <ul style="list-style-type: none"> • Allow both • Force SSH1 • Force SSH2
Port	In this text box, specify the <code>ssh</code> port to listen to.
Use Private key file	Select this check box, if you want to pass server authentication using a private <code>ssh</code> key. In the text box, specify the location of the file with your private <code>ssh</code> key. If necessary, click the Browse button  to open the Select Path dialog box.
Change password	Click this button to open the SSH PAssword dialog box, where you can specify the password for the current CVS root.
Proxy Settings	See Proxy Settings section below.

Local

Tip

PhpStorm does not provide the server functionality. If you want to use a local CVS client, you need to install CVS on your local host computer and configure it to work as a server.

Item	Description
Path to CVS client	In this text box, type the path to CVS client installed on the host computer and configured to work as a server. If necessary, click the Browse button  to open the Select Path dialog box.
Server command	In this text box, specify the server command.

Proxy settings

Item	Description
Use proxy	Select this check box to enable using the Proxy server and access the Login , Password , Proxy host , and Proxy port text boxes below. Tip This check box is available only in two cases: <ul style="list-style-type: none"> • The connection method is <code>pserver</code> or <code>ssh</code>. • The connection method is <code>ext</code> and the Use internal ssh implementation check box is selected.
Protocol	Select the protocol to use. The available options are: <ul style="list-style-type: none"> • HTTP • Socks4 • Socks5
Login	In this text box, specify your user name.
Password	In this text box, specify the user password.
Proxy host	In this text box, specify the Proxy host name.
Proxy port	In this text box, specify the Proxy port number.

See Also

Procedures:

- [Configuring CVS Roots](#)
- [Updating Local Information in CVS](#)
- [Configuring Global CVS Settings](#)

Reference:

- [CVS Global Settings Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

CVS Tool Window

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

VCS | [Browse CVS Repository](#)
View | [Tool Windows](#) | [CVS](#)

This tool window opens, when you browse a CVS repository, and enables you to view contents of the repository, check out files, browse changes, view annotations and navigate to source code.

Item	Description
	Click this button to close the current tab.
	Click this button to open the selected file in the editor.
	Click this button to obtain a local copy of the selected file or directory.
	Click this button to open selected file in the editor with the annotations turned on. See Viewing Annotations .
	Click this button to see the changes that affect the selected file or directory, and that have been committed to the repository by a certain user, or during the specified period. The filtering information is entered in the Search Criteria dialog. Search results show in a dedicated tab of the Changes tool window .
	Click this button to show reference page.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Browsing CVS Repository](#)

Reference:

- [CVS Roots Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Import Into CVS

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

VCS | Import into CVS

Use this dialog to import a directory into the specified CVS repository.

Select CVS configuration

Use this page to select the target CVS root and [change its configuration](#), if necessary.

Select directory to import to

Use this page to select the target directory.

Select import directory

Use this page to select the directory to be imported. If you are importing an PhpStorm project, make sure that the project file is located under that directory. Multiple selection is not available.

Customize keyboard substitution

Use this page to specify the [keyword substitution rule](#) for the files imported into the repository.

Import settings

Item	Description
Name in repository	In this field, specify the name that corresponds to the <code>module</code> argument. Tip For import, <code>module</code> refers to the absolute location in the repository, not to a module name defined in the modules file.
Vendor	In this field, specify the name that corresponds to the <code>vendor-tag</code> argument. This tag is used as a branch tag. No checkouts will ever be done explicitly on it. Type a name that is relevant to the project, or just <code>VENDOR</code> .
Release tag	In this text box, specify the string that corresponds to the <code>release-tag</code> argument. The tag should refer to a version or a release number. Tip A tag name cannot contain punctuation marks. For example: <code>-release-2.2</code> is wrong <code>release-2-2</code> is correct.
Log message	In this field, specify the string that corresponds to the <code>-m</code> command-line argument. By default, the field shows the previous log message; you can accept default, or type a new comment.
Checkout after import	Select this option to have CVS checkout run after completing the import operation.
Make checked out files read-only	Select this option to mark the checked out files as read-only after import. This option is disabled if the Checkout after import option is cleared.

See Also

Procedures:

- [Importing a Local Directory to CVS Repository](#)
- [Configuring CVS Roots](#)
- [Configuring Global CVS Settings](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Check Out from CVS Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac 

VCS | Checkout From Version Control | CVS

The dialog consists of the following pages:

- [Select CS Configuration](#)
- [Select CVS Element to Check Out](#)
- [Select Checkout Location](#)
- [Check out to](#)

Select CVS configuration

Item	Description
List of available CVS configurations	Use this list to select the desired CVS configuration.

Configure Click this button to define a new CVS configuration, or modify an existing one, in the [CVS Roots](#) dialog box.

Select CVS element to check out

Use this page to select elements of the repository to check out. Next button is only available when an element is selected.

Select checkout location

Use this page to specify the target location for the artifacts to check out. All actions can be performed using the toolbar buttons, or context menu.

Item	Shortcut	Description
	Ctrl+1Command 1	Navigate to the user home.
	Ctrl+2Command 2	Navigate to the project root directory.
	Alt+InsertCommand N	Create a new directory, where the files will be checked out to.
	DeleteDelete	Delete selected directory.
	Ctrl+Alt+YCommand Alt Y	Synchronize with external changes.

Check out to

Use this page to define CVS-specific checkout options.

Item	Description
List of local paths	Select the local path to which the module name should be added.
Make new files read-only	Check this option to set read-only attribute for the files that did not exist locally but were checked out from the repository.
Prune empty directories	Check this option to delete empty directories from the repository.
Change keyword substitution to	Check this option to enable keyword substitution, and select the desired substitution mode from the drop-down list.

See Also

Procedures:

- [Checking Out Files from CVS Repository](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Configure CVS Root Field by Field Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

VCS | CVS | **Configure CVS Roots**

This dialog box opens when you click the Edit by Field button in the [CVS Roots](#) dialog box. Use this dialog box to specify the parameters for connecting to the CVS server and have PhpStorm assemble them into a correct repository string according to the CVS root string syntax.

Item	Description
Method	From this drop-down list, select the desired connection method. The available options are: <ul style="list-style-type: none"> • pserver • ext • ssh (internal implementation) • local
User	In this text box, type your login to the CVS server.
Port	In this text box, specify the port to listen to on the CVS server host.
Host	In this text box, type the name of the host where the desired CVS server is located.
Repository	In this text box, type the path to the CVS repository relative to the host name.

See Also

Procedures:

- [Configuring CVS Roots](#)

Reference:

- [CVS Roots Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Rollback Actions with Regards to File Status

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

In this section:

- [Menu commands according to file status.](#)
- [Effect of rolling back local changes.](#)

Menu commands according to file status

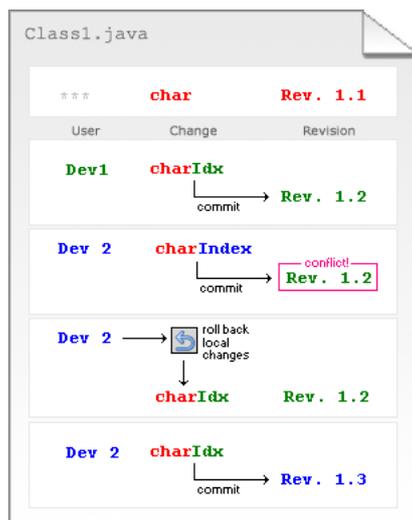
Depending on the file status, the Rollback Changes command will be *aliased* as shown in the following table:

File status	Rollback Command	Result
modified	Rollback Local Changes	all changes made in the file will be reverted, and the file will acquire the <i>up to date</i> status
deleted	Rollback Deletion	file will be restored both on the disk and in CVS with the <i>up to date</i> status
externally deleted	Rollback Deletion	file will be restored on the disk and assigned the <i>up to date</i> status
added	Rollback Creation	file will be deleted from disk
merged	Rollback Local Changes	all local changes will be dropped, changes from the repository will be accepted, and the file will be assigned the <i>up to date</i> status
merged with conflicts	Rollback Local Changes	all local changes will be dropped, changes from the repository will be accepted, and the file will be assigned the <i>up to date</i> status
unknown	Rollback Creation	file will be deleted from the disk

Effect of rolling back local changes

The effect of Rollback Local Changes may not be what you intuitively expect in terms of the revision you have locally after running the command.

The image below represents a file in CVS and a sequence of actions by two developers. It shows a simple example of what happens in terms of the local copy's CVS revision after rolling back conflicting local changes.



Here's what happens:

- The developer `Dev1` takes Revision 1.1 from the repository, modifies it, and commits changes to CVS.
- The developer `Dev2` doesn't know about `Dev1`'s changes and modifies the same code in the local copy of Revision 1.1. When `Dev2` commits these changes, he gets a message from CVS that the repository has changed. So he runs Update to synchronize, and his local copy is then updated to Revision 1.2, and CVS sets the *Merged with Conflicts* status on the file.
- `Dev2` decides to roll back local changes. He is left with a local copy of Revision 1.2.
- When `Dev2` commits the file to CVS, it becomes Revision 1.3 even though the content is identical to Revision 1.2.

See Also

Procedures:

- [Reverting to a Previous Version](#)
- [Reverting Local Changes](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Update Directory / Update File Dialog (CVS)

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

VCS | CVS | Update File

VCS | CVS | Update Directory Editor | context menu of a file | CVS | Update File

- Project Tool Window | context menu of a file | CVS | Update File
- Project Tool Window | context menu of a selection | CVS | Update Files
- Project Tool Window | context menu of a folder | CVS | Update Directory
- Project Tool Window | context menu of a selection | CVS | Update Directories

In these dialog boxes, configure synchronization of local files or folders with the repository.

Item	Description
Branch Merging	<p>In this area, specify the repository branch to synchronize with.</p> <ul style="list-style-type: none"> • Don't merge - select this option to synchronize with the counterpart of the current branch in the repository. This option is selected by default. • Merge with branch - select this option to have the local copy synchronized with a repository branch different from the counterpart of the current branch. In the first text box below, specify the branch to synchronize with. Type the branch name manually or click the Browse button  and select the desired branch from the list in the Select Tag dialog box that opens. <p>Note</p> <p>This option is equivalent to the <code>-j</code> command-line option of the <code>update</code> command.</p> <ul style="list-style-type: none"> • Merge two branches - select this option to have the local copy synchronized with the result of merging two repository branches different from the counterpart of the current branch. In the text boxes below, specify the branches to synchronize with. <p>Note</p> <p>This option is equivalent to the <code>-j -j</code> command-line option of the <code>update</code> command.</p>
Use Version	<p>In this area, specify the version of repository file(s) or folder(s) to synchronize with.</p> <ul style="list-style-type: none"> • Default - select this option to have the local copy synchronized with the latest repository version. • By tag - select this option to have the local copy synchronized with a particular repository version. In the text box, specify the desired version number or tag. Type the version number or tag manually or click the Browse button  and select the desired branch from the list in the Select Revision or Tag dialog box that opens. <ul style="list-style-type: none"> ◦ When updating a single file, you can specify its repository counterpart either through a version number or a tag. ◦ When updating an entire folder, its repository counterpart can be specified only through a tag. <p>Note</p> <p>This option is equivalent to the <code>-r</code> command-line option of the <code>update</code> command.</p> <ul style="list-style-type: none"> • By date - select this option to have the local copy synchronized with a repository version that was submitted on a particular date. Type the date manually or click the Calendar button  and select the desired date from the calendar pop-up window that opens with the current date selected by default. <p>Note</p> <p>This option is equivalent to the <code>-D</code> command-line option of the <code>update</code> command.</p>
Reset sticky data	<p>Select this check box to remove the date or tag restriction from the local file(s) or folder(s) to be updated. Such restrictions are set on files and folders checked out or previously updated with the By tag or By date options.</p> <p>Note</p> <p>This option is equivalent to the <code>-A</code> command-line option of the <code>update</code> command.</p>
Prune empty directories	<p>Select this option while updating a directory to have PhpStorm remove the subfolders whose repository counterparts are empty.</p> <p>Note</p> <p>This option is equivalent to the <code>-P</code> command-line option of the <code>update</code> command.</p>
Change keyword substitution to	<p>Select this check box to have the default keyword expansion mode changed. From the drop-down list, choose the relevant substitution.</p> <p>Note</p> <p>This option is equivalent to the <code>-k</code> command-line option of the <code>update</code> command.</p>
Create new directories	<p>Select this option while updating a directory to have PhpStorm create new local subfolders when any new subfolders have been created in the repository counterpart of the directory to be updated.</p> <p>Note</p> <ol style="list-style-type: none"> 1. This option is equivalent to the <code>-k</code> command-line option of the <code>update</code> command. 2. By default, all the new subfolders will be picked including empty ones. To avoid creating empty subfolders, it is recommended that you select the Prune empty directories check box as well.

Clean copy Select this check box to have PhpStorm backup the local changes and replace the changed file(s) with their counterparts from the repository.

Note

This option is equivalent to the -C command-line option of the update command.

Do not show this dialog in the future Select this option to have the update operation performed [silently](#) in the future.

See Also

Procedures:

- [Updating Local Information in CVS](#)
- [Updating Local Information](#)

External Links:

- <http://savannah.nongnu.org/userguide/>

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Git Reference

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

In this part:

- [Push Changes Dialog](#)
- [Push Active Branches](#)
- [Pull Changes Dialog](#)
- [Merge Branches Dialog](#)
- [Checkout Dialog](#)
- [Tag Dialog](#)
- [Stash Dialog](#)
- [Unstash Changes Dialog](#)
- [Clone Repository Dialog](#)
- [Rebase Branches Dialog](#)
- [Reset Head Dialog](#)
- [Current Branch Dialog](#)
- [GitHub Integration Reference](#)

See Also

Procedures:

- [Using Git Integration](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Push Changes Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

VCS | Git | Push Changes

Use this dialog box to specify arguments for uploading changes to a remote repository from the branches in one local repository.

Item	Description
Git Root	From this drop-down list, select the path to the required local repository.
Current Branch	This read-only field shows the name of the branch which is currently checked out in the selected local repository.
Remote	From this drop-down list, select the alias of the target remote repository.

Tip

To check that the selected alias corresponds to the correct URL address, expand the list - the URL addresses will be displayed explicitly.

Push Use this drop-down list to specify the push strategy. The following options are available:

- **Selected Branches** - select this option to push changes only from specific branches. When this option is selected, the **Branches** list box is enabled and displays all the branches in the selected local repository.
- **Default** - select this option to push changes according to the default Git settings of the remote repository.
- **All** - select this option to push changes from all branches but exclude changes from tags. When this option is selected, the **Branches** list box is empty and the **Show**

Tags and Push Tags check boxes are disabled.

- **Mirror** - select this option to push changes from all branches and tags. When this option is selected, the **Branches** list box is empty, the **Show Tags** check box is disabled, the **Push Tags** and **Force Update** check boxes are selected and disabled.

Branches Use this list box to select the branches and/or tags from which you want to push changes.

Tip

By default, the list box shows only branches. To have the names of tags displayed too, select the **Show Tags** check box.

Show Tags Select this check box to have the names of tags displayed in the **Branches** list box.

Push Tags Select this check box to push changes from all the tags in the local repository.

Use Thin Pack Select this check box to minimize the amount of data transferred. Minimization is ensured through running additional CPU cycles.

Tip

This option is recommended for slow connections.

Force Update Select this check box to overwrite remote references even if they are not ancestors of the corresponding local references.

Warning

Use this option with care because it may cause the remote repository to lose previously pushed changes.

Push Click this button to initiate uploading changes from the local repository to the specified upstream according to the defined settings.

See Also

Procedures:

- [Uploading a Local Git Repository \(Push\)](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Push Active Branches

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

VCS | Git | Push Active Branches

Use this dialog box to specify options for uploading changes to a remote repository from the active branches in multiple local repositories (Git roots).

Item	Description
Commits	<p>This read-only list box shows a list of active branches in all your local repositories (roots). All the commits performed in the active branch of a root since the last push are shown below the relevant root node. The latest commit is shown at the top of each list. To have a commit pushed, select the check box next to it.</p> <p>Note</p> <p>If you skip a commit but include commits performed later, PhpStorm informs you that you need to apply the Rebase operation.</p>
View	Click this button to open the Paths affected in commit dialog box, where you can view a list of files that contain changes included in the selected commit.
Fetch	click this button to have the changes from a remote repository downloaded without applying them locally.
Rebase	<p>Click this button to initiate rebasing if you skip a commit but include commits that were performed later.</p> <ul style="list-style-type: none"> • If no conflicts are detected, rebasing is performed silently and the commit to be skipped is moved to the top of the list. • If any conflicts arise, a Merge tool appears, where you can resolve them.
Push	Click this button to have the selected commits pushed .
Clean working tree before rebase	<p>In this area, specify the technique to use for cleaning the working tree which involves removing files that are not under Git control. The available options are:</p> <ul style="list-style-type: none"> • Using Stash - choose this option to have a patch with stashed changes generated. • Using Shelve - choose this option to have a patch with shelved changes generated. <p>Note</p> <ul style="list-style-type: none"> • Patches with stashed changes are generated by Git itself. To apply them later, you do not need PhpStorm. • Patches with shelved changes are generated by PhpStorm. Normally, they are also applied through the IDE. Applying shelved changes outside PhpStorm is also possible but requires additional steps.
Rebase and Push	Click this button to initiate rebasing and pushing in the background.

Warning

It is strongly recommended that you do not perform any editing of the affected files until the process is completed successfully.

See Also

Procedures:

- [Uploading a Local Git Repository \(Push\)](#)

Reference:

- [Push Changes Dialog](#)
- [Rebasing Branches](#)

External Links:

- <http://www.kernel.org/pub/software/scm/git/docs/git-push.html>
- <http://www.kernel.org/pub/software/scm/git/docs/git-rebase.html>
- <http://www.kernel.org/pub/software/scm/git/docs/git-clean.html>

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Pull Changes Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

VCS | Git | Pull Changes

Use this dialog box to specify parameters for fetching changes from a remote repository and applying them to a local repository.

Item	Description
Git Root	From this drop-down list, select the path to the local repository which you want to refresh.
Current Branch	This read-only field shows the name of the branch which is currently checked out in the selected local repository. The changes retrieved from the source remote repository will be applied to the displayed branch.
	<p>Note</p> <p>The contents of the field depend on the selection in the Git Root drop-down list.</p>
Remote	From this drop-down list, select the alias of the source remote repository.
	<p>Tip</p> <p>To check that the selected alias corresponds to the correct URL address, expand the list - the URL addresses will be displayed explicitly.</p>
Get Branches	Click this button to have the Branches to Merge list box display the actual list of branches in the source remote repository.
Branches to Merge	Use this list box to specify the branches to which you want to apply the fetched changes.
Strategy	From this drop-down list, select the merge strategy . The available options are: <ul style="list-style-type: none"> • Default • Resolve - select this option if you need to resolve two HEADs, one of which is the current branch and the other HEAD is the branch from which you pulled changes. When this option is selected, the 3-way merge algorithm is applied. • Recursive - the default merge strategy for pulling one branch. Select this option if you need to resolve two HEADs by applying the 3-way merge algorithm and there are more than one common ancestor that can be used for 3-way merge. • Octopus - the default merge strategy for pulling more than one branch.
	<p>Note</p> <p>Merges that require resolving conflicts manually are not performed.</p> <ul style="list-style-type: none"> • Ours - select this option if you need to supersede old development history of side branches. By applying this strategy any number of HEADs can be resolved but the result of the merge is always the HEAD of the current branch. • Subtree - a modified recursive strategy.
No Commit	Select this check box if you need to inspect and, if necessary, adjust the result of merging the fetched data before committing the result. The merge is performed but is not committed automatically, as if it failed.
No Fast Forward	Select this check box to generate a merge commit even if the merge resolved as a fast-forward .
Squash Commit	Select this check box to create a single commit on top of the current branch instead of merging one or more other branches. The working tree and index state are produced as if a real merge happened, but commit is not performed and the HEAD is not moved.
Add Log Information	Select this check box to have PhpStorm populate, in addition to branch names, a log message with one-line descriptions from the actual commits that are being

merged.

Pull Click this button to initiate fetching changes from the specified remote repository and applying them locally according to the defined settings.

See Also

Procedures:

- [Updating a Local Git Repository \(Pull\)](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Merge Branches Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

VCS | Git | Merge Changes

Use this dialog box to specify arguments for merging branches in a local Git repository.

Item	Description
Git Root	From this drop-down list, select the path to the local repository in which you want to merge branches.
Current Branch	This read-only field shows the name of the branch which is currently checked out in the selected local repository. This is the target branch, the changes from the selected source branches will be applied to it. Note The contents of the field depend on the selection in the Git Root drop-down list.
Branches to Merge	Use this list box to specify the source branches from which changes will be applied to the target branch. Note The list shows only those branches that contain applicable commits. Applicable commits are commits made after a branch separated from the target branch.
Strategy	From this drop-down list, select the merge strategy . The available options are: <ul style="list-style-type: none"> • Default • Resolve - select this option if you need to resolve two HEADs, one of which is the current branch and the other HEAD is the branch which you selected in the Branches to Merge list. When this option is selected, the 3-way merge algorithm is applied. • Recursive - the default merge strategy for merging the current branch with one branch. Select this option if you need to resolve two HEADs by applying the 3-way merge algorithm and there are more than one common ancestor that can be used for 3-way merge. • Octopus - the default merge strategy for merging the current branch with more than one branch. Note Merges that require resolving conflicts manually are not performed. <ul style="list-style-type: none"> • Ours - select this option if you need to resolve several HEADs. The result of the merge is always the HEAD of the current branch. • Subtree - a modified recursive strategy. Note When two or more source branches are selected in the Branches to Merge list box, only the Octopus and Ours options are available.
No Commit	Select this check box if you need to inspect and, if necessary, adjust the result of merging before committing the result. The merge is performed but is not committed automatically, as if it failed.
No Fast Forward	Select this check box to generate a merge commit even if the merge resolved as a fast-forward .
Squash Commit	Select this check box to create a single commit on top of the current branch instead of merging one or more other branches. The working tree and index state are produced as if a real merge happened, but commit is not performed and the HEAD is not moved.
Add Log Information	Select this check box to have PhpStorm populate, in addition to branch names, a log message with one-line descriptions from the actual commits that are being merged.
Commit Message	In this text box, provide a description for the commit. Note The text box is available only if the No Commit check box is not selected.
Merge	Click this button to initiate merging the specified branches in the local repository according to the defined settings.

See Also

Procedures:

- [Merging Branches](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Checkout Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

VCS | [Git](#) | [Checkout Branch](#)

Use this dialog box to switch between existing branches and create new branches in a local repository.

Item	Description
Git Root	From this drop-down list, select the path to the local repository in which you want to switch between existing branches or create a new branch.
Current Branch	This read-only field shows the name of the working branch in the selected local repository.
Checkout	From this drop-down list, select the branch or tag to which you want to switch or which you want to copy to a new branch. To specify a particular commit, type its commit hash or use an expression, for example, of the following structure: <branch><number of commits backwards between the latest commit (HEAD) and the required commit>. Refer to the Git commit naming conventions for details.
Validate	Click this button to check that the commit specified in the Checkout field exists and view which files were affected in it. The Validate button next to an editable field indicates that regular expressions are allowed in the field.
Include Tags	Select this check box to have tags also shown in the Checkout drop-down list.
As New Branch	In this text box, type the name of the branch to be created. You can also type the name of an existing branch and select the Override check box.
Create Ref Log	Select this check box to have a reference log for the new branch created where all changes made to the branch references will be recorded.
Track Branch	Select this check box to track the parent of the new branch.
Override	Select this check box if you want to checkout a branch to an existing branch. Use this option with care because the history of the target branch will be lost.
Checkout	Click this button to switch to the specified existing branch or to create a new branch in the local repository.

See Also

Procedures:

- [Switching Between Branches \(Checkout\)](#)
- [Creating a New Branch](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Tag Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

VCS | [Git](#) | [Tag Files](#)

Use this dialog box to assign tags to commits and commit objects.

Item	Description
Git Root	In this drop-down list, select the path to the local repository in which you want to tag a commit or a commit object.
Current Branch	This read-only field shows the name of the branch which is currently checked out in the selected local repository.
Tag Name	In this text box, type the name of the new tag.
Force	Select this check box to assign an existing tag to another commit or object.
Note	This check box is enabled only if you type the name of an existing tag in the Tag Name text box whereupon PhpStorm displays an error message.
Commit	In this text box, specify the commit or object which you want to tag. Type the desired commit hash or use an expression, for example, of the following structure: <branch><number of commits backwards between the latest commit (HEAD) and the required commit>. Refer to the Git commit naming conventions for details.
Validate	Click this button to check that the commit specified in the Commit field exists and view which files were affected in it.

Message In the text box, provide a description of the commit to be tagged.

Tip

Fill in this text box to create an annotated tag.

Create Tag Click this button to have the specified commit or commit object tagged.

See Also

Procedures:

- [Adding Tags](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Stash Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

VCS | Git | Stash Changes

Use this dialog box to specify parameters for saving local modifications in a new stash.

Item	Description
Git Root	From this drop-down list, select the path to the local repository in which you want to stash modifications.
Current Branch	This read-only field shows the name of the branch which is currently checked out in the selected local repository.
Message	In this text box, provide a description of the new stash. This description will be displayed in the Unstash Changes dialog box to help you select the required stash.
Keep Index	Select this check box to have PhpStorm bring the changes staged in the index to your working tree for examination and testing.
Create Stash	Click this button to have the specified local modification saved in a new stash.

See Also

Concepts:

- [Shelved Changes](#)

Procedures:

- [Stashing and Unstashing Changes](#)
- [Shelving and Unshelving Changes](#)

Reference:

- [Unstash Changes Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Unstash Changes Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

VCS | Git | Unstash Changes

Use this dialog box to specify parameters for applying stashed modifications to your working copy.

Item	Description
Git Root	From this drop-down list, select the path to the local repository in which you want to apply the stashed modifications.
Current Branch	This read-only field shows the name of the branch which is currently checked out in the selected local repository.
Stashes	From this list, select the stash that you want to apply. Each item is supplied with a number, an indication of the branch in which it was created, and the description provided during the stash creation.
View	Click this button to open the Paths affected in commit dialog box where you can learn which files are affected in the selected stash.
Drop	Click this button to remove the selected stash from the Stashes list.
Clear	Click this button to remove all the stashes from the Stashes list.
Pop Stash	Select this check box to have the selected stash removed from the list after it is applied.
Reinstate Index	Select this check box to have the stashed index modifications applied as well.
As New Branch	If you want to create a new branch on the basis of the selected stash, type the name of the new branch in this list box.

Note

When the list box is filled in, the **Pop Stash** and **Reinstate Index** check boxes are selected and disabled.

Apply Stash Click this button to have the specified modifications applied to the working copy.

Pop Stash Click this button to have the specified modifications applied to the working copy and then removed from the list.

Note

The button is available only if the **Pop Stash** check box is selected.

See Also

Concepts:

- [Shelved Changes](#)

Procedures:

- [Stashing and Unstashing Changes](#)
- [Shelving and Unshelving Changes](#)

Reference:

- [Stash Dialog](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Clone Repository Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

[Checkout from Version Control](#) | [Git](#)

Use the dialog box to set up a local repository by downloading the data from a remote repository.

Item	Description
Git Repository URL	In this text box, type the URL of the remote repository which you want to clone.
Test	Click this button to check that connection to the remote repository has been established successfully.
Parent Directory	In this text box, specify the directory where you want PhpStorm to create a folder for your local Git repository. Type the path manually or click the Browse button  and choose the desired directory in the Select Path dialog box that opens.
Directory Name	In this text box, type the name of the new folder into which the repository will be cloned.
Warning	
	The parent directory must not contain a folder with the specified name.
Clone	Click this button to start cloning the specified repository.

See Also

Procedures:

- [Setting up a Local Git Repository](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Rebase Branches Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

[VCS](#) | [Git](#) | [Rebase](#)

Use this dialog box to specify the branch to rebase, the new base, the rebasing mode, and configure the rebasing procedure.

Item	Description
Git Root	From this drop-down list, select the path to the local repository in which you want to rebase a branch.
Branch	From this drop-down list, select the branch to rebase. By default, the current working branch is selected. If you specify another branch, it will be automatically checked out first.

Interactive	Select this check box to view and possibly edit a list of commits to be rebased.
Preserve Merges	Select this check box to have the possibility to recreate merges instead of ignoring them. The check box is available only when the Interactive check box is selected.
	Warning
	When this check box is selected, Git does not support squashing commits.
Onto	Use this drop-down list to specify the new base for the selected branch. To specify the required commit, type its commit hash or use an expression, for example, of the following structure: <branch>~<number of commits backwards between the latest commit (HEAD) and the required commit>. Refer to the Git commit naming conventions for details. If no commit is specified, the HEAD of the selected branch is used as the new base.
Validate	Click this button to check that the commit specified in the Onto field exists and view which files were affected in it.
From	Use this drop-down list to specify the commit starting from which you want to apply the branch to the new base. Type the required commit hash or use an expression, for example, of the following structure: <branch>~<number of commits backwards between the latest commit (HEAD) and the required commit>. Refer to the Git commit naming conventions for details. To apply the entire branch, leave the field empty.
Validate	Click this button to check that the commit specified in the From field exists and view which files were affected in it.
Show Tags	Select this check box to have tagged commits included in the Onto and From drop-down lists.
Show Remote Branches	Select this check box to have branches in the remote repository included in the Onto drop-down list.
Merge Strategy	From this drop-down list, select the merge strategy . The available options are: <ul style="list-style-type: none"> • Default • Resolve - select this option if you need to resolve two HEADs, one of which is the current branch and the other HEAD is the branch from which you pulled changes. When this option is selected, the 3-way merge algorithm is applied. • Recursive - the default merge strategy for pulling one branch. Select this option if you need to resolve two HEADs by applying the 3-way merge algorithm and there are more than one common ancestor that can be used for 3-way merge. • Octopus - the default merge strategy for pulling more than one branch. <p style="text-align: center;">Note</p> <p style="text-align: center;">Merges that require resolving conflicts manually are not performed.</p> <ul style="list-style-type: none"> • Ours - select this option if you need to supersede old development history of side branches. By applying this strategy any number of HEADs can be resolved but the result of the merge is always the HEAD of the current branch. • Subtree - a modified recursive strategy.
Do not use merge strategies	When this check box is selected, no merge strategy is applied during rebase.
Rebase	Click this button to initiate rebasing according to the defined settings.

See Also

- Concepts:
- [Version Control with PhpStorm](#)

- Procedures:
- [Rebasing Branches](#)

- Web Resources:
- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>

Rebasing Commits Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

VCS | Git | Rebase

The dialog box opens when you click OK in the [Rebase branches](#) dialog box, with the **Interactive** check box selected.
Use this dialog box to define the order of applying commits, squash or edit commits before applying, and skip commits that contain extraneous changes.

Item	Description
Action	Use this drop-down list to define the action to apply to the selected commit. The available options are: <ul style="list-style-type: none"> • Pick - select this option to apply the commit as is. • Edit - select this option to update the commit before applying it. • Skip - select this option to ignore the commit. • Squash - select this option to combine the commit with the previous commit.

Commit	This read-only field displays the hash of the selected commit.
Comment	This read-only field shows the comment supplied for the selected commit.
View	Click this button to view the files affected in the selected commit.
Move Up/Move Down	Use these buttons to change the order in which commits should be applied.

See Also

- Concepts:
- [Version Control with PhpStorm](#)
- Procedures:
- [Interactive Rebase](#)
- Reference:
- [Rebase Branches Dialog](#)
- Web Resources:
- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>

Reset Head Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

VCS | Git | Reset Head

Use this dialog box to specify the commit you want to reset the current HEAD to and define the reset method.

Item	Description
Git Root	In this drop-down list, select the path to the local repository in which you want to reset the HEAD of a branch.
Current Branch	This read-only field shows the name of the branch which is currently checked out in the selected local repository.
	Note
	The contents of the field depend on the selection in the Git Root drop-down list.
Reset Type	Use this drop-down list to define the reset method to use. The available options are: <ul style="list-style-type: none"> • Mixed - the default strategy. When this option is selected, the index is reset while the working tree is not, which means that changed files are preserved but not marked for commit. You are presented with a report of what has not been updated. • Soft - when this option is selected, the index and the working tree are not affected, only the HEAD pointer is moved to the specified commit. Your current state with any changes remains different from the commit you are switching to. All the changes are "staged" for committing. • Hard - when this option is selected, both the working directory and the index are changed to the specified commit.
To Commit	In this text box, specify the commit you want to reset the current HEAD to. Type the desired commit hash or use an expression, for example, of the following structure: <code><branch>><number of commits backwards between the current HEAD and the required commit> .</code> Refer to the Git commit naming conventions for details.
Validate	Click this button to check that the commit specified in the To Commit field exists and view which files were affected in it.
Reset	Click this button to initiate resetting the HEAD to the specified commit according to the defined method.

See Also

- Procedures:
- [Resetting Head Commit](#)
- Web Resources:
- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>

Current Branch Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

VCS | Git | Current Branch

Use this dialog box to find out which branch you are [currently working with](#) and which branch is [currently tracked](#). In this dialog box, you can also [switch tracking](#) to another branch in any repository.

Item	Description
Git Root	From this drop-down list, select the path to the local repository in which you want to detect which branch you are currently working with.
Current Branch	This read-only field shows the name of the working branch in the selected local repository.

- Repository** From this drop-down list, select the path to the local repository in which you want to detect which branch is currently tracked or change the tracking, if necessary.
- Branch** From this drop-down list, select the branch which you want to be tracked. By default, the field shows the currently tracked branch.
- Change Tracked Branch** Click this button to have the selected branch tracked.

See Also

Procedures:

- [Managing Branch Tracking](#)
- [Switching Between Branches \(Checkout\)](#)
- [Creating a New Branch](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

GitHub Integration Reference

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

In this part:

- [Select Repository to Clone Dialog](#)
- [Share Project on GitHub Dialog](#)
- [Login to GitHub Dialog](#)
- [Create Gist Dialog](#)

See Also

Procedures:

- [Using GitHub Integration](#)
- [Using Git Integration](#)
- [Version Control with PhpStorm](#)

Reference:

- [Git Reference](#)
- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Select Repository to Clone Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

VCS | [Checkout from Version Control](#) | [GitHub](#)

Use this dialog box to choose the source remote repository, the folder to clone the source to, and the name of the project to be created around the cloned data.

Item	Description
Repository	From this drop-down list, select the source repository to clone the data from.
Parent Directory	In this text box, specify the directory where the local repository for cloned sources will be created. Type the path to the directory manually or click the Browse button  and choose the desired directory in the Select project destination folder dialog box that opens.
Directory Name	In this text box, specify the name of the folder where the repository will be created based on the cloned sources.
Clone	Click this button to start cloning the sources from the specified remote repository.

See Also

Procedures:

- [Publishing a Project on GitHub](#)
- [Using GitHub Integration](#)
- [Using Git Integration](#)
- [Version Control with PhpStorm](#)

Reference:

- [Share Project on GitHub Dialog](#)
- [GitHub](#)
- [GitHub Integration Reference](#)
- [Git Reference](#)
- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>

- <http://youtrack.jetbrains.net/issues/WI>

Share Project on GitHub Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

VCS | [Import into Version Control](#) | [Share Project on GitHub](#)

Use this dialog box to create a repository on the GitHub remote storage and publish your project sources there.

Item	Description
New repository name	In this text box, type the name of the repository to be created. By default, PhpStorm suggests the name of the current project.
Description	In this text box, describe the main functionality implemented in the project.
Private	Select this check box to suppress access to the repository for other users.
	Warning
	The check box is disabled for free accounts.
Share	Click this button to have PhpStorm initiate repository creation and afterwards upload the project sources to the new repository.

See Also

Procedures:

- [Publishing a Project on GitHub](#)
- [Using GitHub Integration](#)
- [Using Git Integration](#)
- [Version Control with PhpStorm](#)

Reference:

- [GitHub](#)
- [GitHub Integration Reference](#)
- [Git Reference](#)
- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Login to GitHub Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

VCS | [Import into Version Control](#) | [Share Project on GitHub](#)

Use this dialog box to log on the [GitHub](#) remote storage using your account credentials or to create a GitHub account if you do not have it yet.

Item	Description
Login	In this text box, type your GitHub logon name.
Password	In this text box, type your GitHub account password.
Test	Click this button to check whether the specified credentials ensure successful logon to your GitHub account.
Sign up	Click this link to open the Sign up for GitHub page where you can create a GitHub account.
Login	Click this button to log on to the GitHub storage using the specified credentials.

See Also

Procedures:

- [Publishing a Project on GitHub](#)
- [Using GitHub Integration](#)
- [Using Git Integration](#)
- [Version Control with PhpStorm](#)

Reference:

- [Share Project on GitHub Dialog](#)
- [GitHub Integration Reference](#)
- [Git Reference](#)
- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Create Gist Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

<context menu of an open file>| **Create gist**

Warning

The menu item and the dialog box are only available when you are logged on GitHub and a file is opened in the editor.

Item	Description
Description	In this text box, provide a brief description of the code snippet to be published.

Note

By default, the new gist will be **public**, that is, visible for all registered GitHub users and your login will be shown as the gist's owner. You can change this default behaviour by selecting the check boxes below.

Private	Select this check box to make the new gist available for you only and invisible for other GitHub users.
Anonymous	Select this check box to make the new gist visible for all registered GitHub users but without displaying your login.
Open in browser	<ul style="list-style-type: none"> Select this check box to have the new gist opened in the default PhpStorm browser. To have PhpStorm display the URL address of the new gist so you can access it later, clear this check box.

See Also

Reference:

- [GitHub Integration Reference](#)
- [GitHub](#)
- [Git Reference](#)
- [Version Control Reference](#)

Version Control:

- [Creating Git Gists](#)
- [Using GitHub Integration](#)
- [Using Git Integration](#)
- [Version Control with PhpStorm](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Mercurial Reference

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

In this part:

- [Clone Mercurial Repository Dialog](#)
- [Create Mercurial Repository Dialog](#)
- [Pull Dialog](#)
- [Push Dialog](#)
- [Switch Working Directory Dialog](#)

See Also

Procedures:

- [Using Mercurial Integration](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Clone Mercurial Repository Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Checkout from Version Control | Mercurial

Use the dialog box to set up a local repository by downloading the data from a remote repository.

Item	Description
------	-------------

Mercurial Repository URL	In this text box, type the URL of the remote repository which you want to clone.
Test	Click this button to check that connection to the remote repository has been established successfully.
Parent Directory	In this text box, specify the directory where you want PhpStorm to create a folder for your local Mercurial repository. Type the path manually or click the Browse button  and choose the desired directory in the Select Path dialog box that opens.
Directory Name	In this text box, type the name of the new folder into which the repository will be cloned.
Warning	
	The parent directory must not contain a folder with the specified name.
Clone	Click this button to start cloning the specified repository.

See Also

Procedures:

- [Setting up a Local Mercurial Repository](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Create Mercurial Repository Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

VCS | **Create Mercurial Repository**

Use this dialog box to create a local Mercurial repository in the folder of your choice.

Item	Description
Create repository for the whole project	Select this option to have a repository initialized in the project root directory. This option is helpful if you want to put the entire project under Mercurial control.
Select where to create repository	Select this option to have a repository initialized in one of the folders below the project root. With this option, you can have folders of your project under control of different version control systems. In the text box below, specify the folder to create the repository in. Type the path manually or click the Browse button  and choose the desired folder in the Select Path dialog box that opens.

See Also

Procedures:

- [Setting up a Local Mercurial Repository](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Pull Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

VCS | Mercurial | **Pull Changesets**

Use this dialog box to specify parameters for fetching changes from a remote repository and applying them to a local repository.

Item	Description
Pull From	In this text box, specify the URL address of the remote repository to fetch the changes from.

See Also

Procedures:

- [Updating a Local Mercurial Repository \(Pull\)](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Push Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

VCS | Mercurial | **Push Changesets**

Use this dialog box to specify arguments for uploading changes to a remote repository from the local repository.

Item	Description
Destination Repository URL	In this text box, specify the URL address of the target remote repository.
Options	In this area, specify the changes to be uploaded and configure the push procedure. The available options are: <ul style="list-style-type: none"> • Revision - select this check box to have a specific changeset uploaded. By default, PhpStorm will upload the latest revision (tip). To have another revision uploaded, type its number in the text box. • Branch - select this check to have a branch pushed entirely and choose the relevant branch from the drop-down list. • Force push - select this check box: <ul style="list-style-type: none"> ◦ If you have created a local branch and want to upload it to the remote repository. ◦ If you do not want to update you local repository before push.

See Also

- Concepts:
- [Version Control with PhpStorm](#)
- Procedures:
- [Uploading a Local Mercurial Repository \(Push\)](#)
 - [Using Mercurial Integration](#)
 - [Version Control with PhpStorm](#)

- Reference:
- [Mercurial Reference](#)
 - [Version Control Reference](#)

- External Links:
- <http://www.selenic.com/mercurial/hq.1.html#push>

- Web Resources:
- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>

Switch Working Directory Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

VCS | Mercurial | [Update to](#)

Use this dialog box to update the repository's [working directory](#) to the specified [changeset](#).

Item	Description
Branch	Choose this option to switch to another line of development . Choose the desired branch from the drop-down list.
Tag	Choose this option to switch to a changeset to which you have previously assigned a tag identifier . Choose the relevant tag from the drop-down list.
Revision	Choose this option to switch to a specific changeset identified by its hash (revision number). Type the hash in the text box.
Overwrite locally modified files (no backup)	Select this check box to have any local changes overwritten.

See Also

- Concepts:
- [Version Control with PhpStorm](#)
- Procedures:
- [Switching Between Working Directories](#)
 - [Using Mercurial Integration](#)
 - [Version Control with PhpStorm](#)

- Reference:
- [Mercurial Reference](#)
 - [Version Control Reference](#)

- External Links:
- <http://www.selenic.com/mercurial/hq.1.html#update>

- Web Resources:
- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>

Subversion Reference

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

In this part:

- [Authentication Required](#)
- [Changes Browser](#)
- [Check Out from Subversion Dialog](#)
- [Configure Subversion Branches](#)
- [Create Branch or Tag Dialog \(Subversion\)](#)
- [Import Into Subversion](#)
- [Integrate Project Dialog \(Subversion\)](#)
- [Integrate to Branch](#)
- [Lock File Dialog \(Subversion\)](#)
- [Select Branch](#)
- [Select Repository Location Dialog \(Subversion\)](#)
- [Set Property Dialog \(Subversion\)](#)
- [Subversion Options Dialog](#)
- [Subversion Working Copies Information Tab](#)
- [SVN Repositories](#)
- [Update Project Dialog \(Subversion\)](#)

See Also

Concepts:

- [Supported Version Control Systems](#)

Reference:

- [File Status Highlights](#)
- [Changes Tool Window](#)

External Links:

- <http://subversion.tigris.org/>

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Authentication Required

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

VCS | Browse Subversion Repository

Use this dialog to specify your credentials and gain access to the Subversion repository. The dialog box is opened, when you add a new repository location, or attempt to browse a repository.

Item	Description
Authentication realm	This read-only area displays the URL and name of the repository.
User name	By default, this field shows the current user name. You can change the name as required.
Password	Type the password, defined for your Subversion account.
Save credentials	Select this check box to preserve the specified user name and password.

See Also

Concepts:

- [Supported Version Control Systems](#)

Procedures:

- [Using Subversion Integration](#)
- [Authenticating to Subversion](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Changes Browser

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

VCS | Integrate Project

The dialog box opens when you select the **Specified** options and click the  button.

Use the dialog box to select which revision to use in integration.

See Also

Procedures:

- [Integrating SVN Projects or Directories](#)

Reference:

- [Integrate Project Dialog \(Subversion\)](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Check Out from Subversion Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

VCS | [Checkout From Version Control](#) | [Subversion](#)

Use this dialog box to create local working copies.

In this topic:

- [Toolbar](#)
- [Main Controls](#)
- [Context Menu](#)

Toolbar

Item	Tooltip and Shortcut	Description
	Add Repository Location	Click this button to configure a new repository location .
	Edit Location URL	Click this button to edit the URL address of the selected repository.
	Discard Location	Click this button to discard selected repository location.
	Show/Hide Details	Click this button to display the details for each node below the repository location (changelist number, user name, date and time of the last change).
	Refresh Ctrl+F5Command F5	Click this button to refresh the view.
	Collapse All Ctrl+Subtract Command Subtract or Ctrl+Minus Command Minus	Click this button to have all the nodes below all the repository locations collapsed.

Repositories

In this area, manage the available repositories and select the locations to check out contents from.

Item	Description
Repositories	Use this tree view to explore and manage the available repository locations. If necessary, right-click a node and choose the relevant item from its context menu.
Checkout	Click this button to check out the contents of the selected node to the specified location.

Context menu

Item	Description
New	Select this menu item to configure a new repository location , or a new remote folder in the selected repository location.
Checkout	Select this menu item to check out the contents of the selected node.
Checkout	Select this menu item to check out the contents of the selected node.
Compare With	Select this menu item to compare the selected node with the specified branch.
Browse Changes	Select this menu item to view changes that match the specified criteria (author, time range, and revision).
Export	Select this menu item to export the contents of the selected repository or folder to the specified destination. The exported contents are not under version control.
Branch or Tag	Select this menu item to create a branch or tag of the selected folder.
Move or Rename	Select this menu item to change name of the selected folder.
Delete	Select this menu item to delete the selected folder from the repository location.
Copy URL Ctrl+Alt+InsertCtrl Alt Insert	Select this menu item to put the URL string to the clipboard.
Refresh	Select this menu item to synchronize to the repository.
Edit Location URL	Select this menu item to edit the selected repository location.
Discard Location	Select this menu item to discard the selected repository location.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Checking Out Files from Subversion Repository](#)
- [Version Control with PhpStorm](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

SVN Checkout Options Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

VCS | [Checkout From Version Control](#) | [Subversion](#)

The dialog box is the third page of the Checkout From Subversion Wizard. In this dialog box, specify the revision to check out and configure the check-out procedure.

Item	Description
Checkout	This read-only field shows the selected source repository.
Destination	From this list, select the directory to create the working copy in. Choose one of the available folders or click the Browse button  and select the relevant folder in the dialog box that opens.
Update / Switch to revision	In this area, specify the revision to check out. The available options are: <ul style="list-style-type: none"> • HEAD - select this option to have the latest revision checked out. • Specified - select this option to have PhpStorm check out a specific earlier revision. Type the revision number in the text box or click the Browse button  and select the relevant revision from the Changes Browse that opens.
Depth	Use this drop-down list to specify the range of recursion into subdirectories. The available options are: <ul style="list-style-type: none"> • Empty - select this option to involve only the current file. • Files - select this option to involve the files in the folder. • Immediates - select this option to involve direct children of the current file. • Infinity - select this option to enable full recursion.
Include external locations	Select this check box to have externals  included in the working copy.

See Also

- Concepts:
- [Version Control with PhpStorm](#)
- Procedures:
- [Checking Out Files from Subversion Repository](#)
 - [Using Subversion Integration](#)
 - [Version Control with PhpStorm](#)

- Reference:
- [Check Out from Subversion Dialog](#)
 - [Subversion Reference](#)

- Web Resources:
- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>

Configure Subversion Branches

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

[Select Branch pop-up dialog box](#) - [Configure Branches](#)

Use this dialog box to compose a list of branches you work with.

Item	Description
Trunk location	In this text box, specify the URL address of the trunk in the repository. If necessary, click the Browse button  to open the Select Repository Location dialog box and select the required trunk in the repository structure tree.
Branch locations	In this list, select the URL address of the required branch.
Add	Click this button to add a branch to the list. The Select Repository Location dialog box opens where you can select the required branch in the repository structure tree.
Remove	Click this button  to remove the selected branch from the list.

See Also

- Concepts:
- [Supported Version Control Systems](#)
- Procedures:
- [Using Subversion Integration](#)
 - [Viewing and Fast Processing of Changelists](#)
 - [Creating Branches and Tags](#)
 - [Integrating Changes to Branch](#)

Reference:

- [Repository and Incoming Tabs](#)
- [Subversion Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Create Branch or Tag Dialog (Subversion)

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

VCS | Subversion | Create Branch or Tag

In this dialog box, set the arguments for creating a branch or a tag on the basis of a local working copy or a repository version.

Item	Description
Copy from	In this section, specify the source folder to create a branch or tag from. The source of the copy can be taken from the local working copy or from the repository.
Working Copy	Click this option to create a branch or tag on the basis of your local working copy. Type the path in the text box or click the Browse button  and locate the desired directory in the Select Path dialog box, that opens.
Repository Location	Click this option to create a branch or tag on the basis of the repository. Do one of the following: <ul style="list-style-type: none"> • Type the URL of the repository location in the text box. • Click the browse button and select the source repository location. • Click the  button to use the project home directory.
Revision	In this section, specify the source revision to create a branch or tag from. The available options are: <ul style="list-style-type: none"> • HEAD - select this option to have a branch or tag created on the basis of the HEAD revision. • Specified - select this option to have a branch or tag created on the basis of a specific revision. Type the revision number in the text box manually or click the Browse button  and select the desired revision in the Changes Browser dialog box, that opens.
Copy to	Use this section to define the target folder for a branch or tag. The available options are: <ul style="list-style-type: none"> • Branch or Tag - select this option to have the selected revision copied to a specific branch or tag. In the Base URL text box, specify the base URL of the branch or tag. In the Name text box, specify the name of the new branch. • Any location - select this option to have to have the selected revision copied to a location of your choice. In the text box below, specify the URL of any valid location. Type the URL manually or click the Browse button  and specify the desired location in the Select Repository Location dialog box, that opens.
Comment	Type some meaningful description in the text area.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Creating Branches and Tags](#)

Reference:

- [Select Repository Location Dialog \(Subversion\)](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Import Into Subversion

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts:  

VCS | Import into Subversion

Use this dialog box to specify the options for importing data into Subversion.

In this topic:

- [Toolbar](#)
- [Main Controls](#)
- [Context Menu](#)

Toolbar buttons

Item	Tooltip and Shortcut	Description
	Add Repository Location	Click this button to configure a new repository location .
	Edit Location URL	Click this button to edit the URL address of the selected repository.
	Discard Location URL	Click this button to discard the selected repository location and remove it from the list.

	Discard Location URL	Click this button to discard the selected repository location and remove it from the list.
	Show/Hide Details	Click this button to display the details for each node below the repository location (changelist number, user name, date and time of the last change).
	Refresh Ctrl+F5Command F5	Click this button to refresh the view.
	Collapse All Ctrl+Subtract Command Subtract OR Ctrl+Minus Command Minus	Click this button to have all the nodes below all the repository locations collapsed.

Main controls

Item	Description
Repositories	Use this tree view to explore and manage the available repository locations. Right-click nodes and examine context menus.
Import	Click this button to import to the selected repository location the contents of the directory that you choose in the Select Path dialog box.

Context menu

Item	Description
New	Select this menu item to configure a new repository location , or a new remote folder in the selected repository location.
Checkout	Select this menu item to check out the contents of the selected node.
Compare With	Select this menu item to compare the selected node with the specified branch.
Browse Changes	Select this menu item to view changes that match the specified criteria (author, time range, and revision).
Export	Select this menu item to export the contents of the selected repository or folder to the specified destination. The exported contents are not under version control.
Branch or Tag	Select this menu item to create a branch or tag of the selected folder.
Move or Rename	Select this menu item to change name of the selected folder.
Delete	Select this menu item to delete the selected folder from the repository location.
Copy URL Ctrl+Alt+InsertCommand Alt N	Select this menu item to put the URL string to the clipboard.
Refresh	Select this menu item to synchronize to the repository.
Edit Location URL	Select this menu item to edit the selected repository location.
Discard Location	Select this menu item to discard the selected repository location.

See Also

Procedures:

- [Using Subversion Integration](#)
- [Importing a Local Directory to Subversion Repository](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi> 
- <http://youtrack.jetbrains.com/issues/WI> 

Integrate Project Dialog (Subversion)

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

VCS | Integrate Project

Use this dialog box to integrate differences between two branches in the Subversion repository into a local working copy.

Item	Description
Source 1	In this text box, specify the URL address of the first branch to compare. If necessary, click the Browse button  and select the desired URL from the Select Repository Location dialog box.
Source 2	In this text box, specify the URL address of the second branch to compare. If necessary, click the Browse button  and select the desired URL from the Select Repository Location dialog box.
Revision	For each source, specify the revision to use. The possible options are: <ul style="list-style-type: none"> • HEAD - select this option to use the Head revision of the source. <p>Note</p> <p>Head revision is suggested by default</p> <ul style="list-style-type: none"> • Specified - select this option to use a revision different from the Head revision. Specify the required revision in the text field. If necessary, click the  button and select the revision from the Changes Browser dialog box.

Tip

Based on the sources and revisions you specify, the difference between source 2 and source 1 is calculated and applied to the local working copy.

- Use Ancestry** Select this check box to take the ancestry of the Source1 and Source2 URLs into consideration when comparing revisions. If the check box is not selected, only the contents of the files are compared.
- Try merge, but make no changes** Select this check box to enable the `--dry-run` switch of the `svn` command. If this check box is not selected, the sources are merged silently.
- Depth** Use this drop-down list to specify the range of recursion into subdirectories. The available options are:
 - Empty** - select this option to involve only the current file.
 - Files** - select this option to involve the files in the folder.
 - Immediates** - select this option to involve direct children of the current file.
 - Infinity** - select this option to enable full recursion.
- Run status after update** Check this option to perform the Subversion `status` command and view all files that have local modifications.

See Also

- Concepts:
- [Supported Version Control Systems](#)
- Procedures:
- [Using Subversion Integration](#)
 - [Integrating SVN Projects or Directories](#)
 - [Version Control with PhpStorm](#)

- Reference:
- [File Status Highlights](#)
 - [Update Info Tab](#)

- Web Resources:
- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>

Integrate to Branch

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

[View](#) | [Tool Windows](#) | [Changes](#) | [Repository Tab](#) | [Context menu of a changelist or a file](#) | [Subversion](#) | [Integrate to Branch](#)

Use this dialog box to specify options for integrating changes into a branch.

Item	Tooltip	Description
Source branch URL		This read-only field shows the URL address of the source branch.
Target branch URL		This read-only field shows the URL address of the target branch.
Integrate into Working Copy		In this list, select the path to the local working copy into which the changes will be integrated.
Try merge, but make no changes		Select this check box to preview the merge result by enabling the <code>--dry-run</code> switch of the <code>svn</code> command. If this check box is not selected, the sources are merged silently.
Run status after update		Select this check box to perform the Subversion <code>status</code> command and view all the files that have local modifications.
 Add		Click this button to add a working copy to the list.
 Remove		Click this button to remove the selected working copy from the list.

See Also

- Concepts:
- [Supported Version Control Systems](#)
- Procedures:
- [Using Subversion Integration](#)
 - [Integrating SVN Projects or Directories](#)
- Reference:
- [Repository and Incoming Tabs](#)
- Web Resources:
- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>

Lock File Dialog (Subversion)

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

VCS | Subversion | Lock
Context menu of a file | Subversion | Lock

Use this dialog box to lock file when it is necessary to avoid overwriting changes.

Item	Description
Lock Comment	In this text box, describe the reason for locking the file and some additional comments, if necessary.
Steal existing lock	Select this check box to override the lock previously set on the desired file by someone else.
Do not show this dialog in the future	Select this check box to suppress displaying this dialog box and have files and folders locked silently.

See Also

- Concepts:
- [Version Control with PhpStorm](#)

- Procedures:
- [Locking and Unlocking Files and Folders](#)

- Web Resources:
- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>

Select Branch

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

[Changes tool window](#) | [Repository Tab](#) | [Merge Info Pane - Browse](#)
[Changes tool window](#) | [Subversion Working Copies Information Tab](#) | [Merge from Update Project/Directory dialog box - Browse](#)

The pop-up dialog box opens when you click the **Browse** button or press **Shift+Enter** to select the path to the target branch.

Use this dialog box to select the relevant branch or working copy.

Item	Description
Trunk	Choose this option to set the current trunk as the target branch or working copy.
Branches	Choose this option to select the relevant branch in the Branches list.
Tags	Choose this option to select the relevant tag in the Tags list.
Configure Branches	Choose this option to open the Configure Subversion Branches dialog box and compose a list of branches you work with.

See Also

- Concepts:
- [Supported Version Control Systems](#)

- Procedures:
- [Using Subversion Integration](#)
 - [Viewing and Fast Processing of Changelists](#)
 - [Creating Branches and Tags](#)
 - [Integrating Changes to Branch](#)

- Reference:
- [Repository and Incoming Tabs](#)

- Web Resources:
- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>

Select Repository Location Dialog (Subversion)

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

VCS | Subversion | Branch or Tag | Copy From | Repository location
VCS | Subversion | Branch or Tag | Copy To | Any location

Item	Description
Repositories	Use this tree view to explore and manage the available repository locations. Right-click nodes and examine context menus.
Copy as	Specify the name under which the file or folder will be stored in branch.

See Also

Procedures:

- [Creating Branches and Tags](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Set Property Dialog (Subversion)

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

VCS | Subversion | Set Property

Use this dialog to define SVN-specific properties for the files and folders under SVN version control (ignore list, externals etc.)

Item	Description
Property name	Enter custom property name in the text field, or use the drop-down list to select one of the pre-defined properties.
Set property value	Click this radio-button to set value for the specified property name in the text area. Properties that accept multiple values, such as an ignore list, can be entered on multiple lines.
Delete property	Click this radio-button to remove selected property from the list.
Update properties recursively	Check this option, if you want to apply the property to every file and directory under the selected directory.

See Also

Procedures:

- [Working with Subversion Properties for Files and Directories](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Subversion Options Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

File | Settings | Project Settings | Version Control | Configure | Subversion for Windows and Linux
PhpStorm | Preferences | Project Settings | Version Control | Configure | Subversion for Mac OS

Item	Description
Use system default Subversion configuration directory	Store Subversion configuration files in the system default directory: <code>user_home\Application Data\Subversion</code>
Subversion configuration directory	Remove the content of the corresponding directory in the Subversion configuration directory. You may need to clear the authorization information from the configuration file, for example, when your credentials have changed.
Clear authentication cache	Remove the content of the corresponding directory in the Subversion configuration directory. You may need to clear the authorization information from the configuration file, for example, when your credentials have changed.

See Also

Procedures:

- [Using Subversion Integration](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Subversion Working Copies Information Tab

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

View | Tool Windows | Changes - Subversion Working Copies Information

Use this tab to configure the format of your working copies.

Tip

The tab is only available, when the current project sources are entirely or partially under Subversion control.

The tab displays a list of all detected directories under Subversion control supplied with information on the formats used.

Item	Description
Refresh	Click this button to get the information on all the detected Subversion working copies up-to-date.
Root Path	This read-only field shows the full path to the directory.
URL	This read-only field shows the URL address of the remote directory the selected local copy is mapped to.
Format	This read-only field shows the actual Subversion format used in the selected directory.
Change	Click this link to open the Convert Working Copy Format dialog box, where you can select the desired format option.
Depth	This read-only field shows the range of recursion into subdirectories specified in the Update dialog box.
Working Copy Root	This read-only field is displayed only if the directory in question is the root of a working copy.
Configure Branches	Click this link to open the Configure Subversion Branches dialog box, where you can view and update the list of branches to work with.
Merge from	Click this link to open the Select Branch pop-up dialog box and appoint the source of changes to merge to the current directory.

See Also

Concepts:

- [Supported Version Control Systems](#)

Procedures:

- [Configuring Format of the Local Working Copy](#)

Reference:

- [Changes Tool Window](#)
- [Version Control](#)

External Links:

- <http://subversion.tigris.org/>

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

SVN Repositories

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

VCS | Browse Subversion Repository

Use this tool window to view, add, and edit location of SVN repositories.

Toolbar buttons

Item	Tooltip and shortcut	Description
	Add Repository Location	Click this button to configure a new repository location . The New Repository Location dialog box opens, where you can select a repository URL in the drop-down list that contains previously added URL addresses.
	Edit Location Url	Click this button to edit the Url of the selected repository.
	Discard Location Url	Click this button to discard the selected repository location and remove it from the list.
	Show/Hide Details	Click this button to display the details for each node under the repository location (changelist number, user name, date and time of the last change).
	Refresh Ctrl+F5Command F5	Click this button to refresh the view.
	Collapse All Ctrl+Subtract Command Subtract or Ctrl+Minus Command Minus	Click this button to collapse all nodes in the tree with repositories.
	Close	Click this button to close the tool window.

Main controls

Item	Description
Repositories	Use this tree view to explore and manage the available repository locations. Right-click nodes and examine context menus.

Context menu

Item	Description
New	Select this menu item to configure a new repository location , or a new remote folder in the selected repository location. The New Repository Location dialog box opens where you can select a repository URL in the drop-down list that contains previously added URL addresses.
Show History	Select this item to open the Version Control tool window with the history of the selected repository location.
Checkout	Select this menu item to check out the contents of the selected node.
Compare with	Select this menu item to compare the selected node with the specified branch.
Browse changes	Select this menu item to view changes that match the specified criteria (author, time range, and revision).

Export	Select this menu item to export the contents of the selected repository or folder to the specified destination. The exported contents are not under version control.
Branch or Tag	Select this menu item to create a branch or tag of the selected folder.
Move or Rename	Select this menu item to change name of the selected folder.
Delete	Select this menu item to delete the selected folder from the repository location.
Copy URL	Select this menu item to put the URL string to the clipboard.
Refresh	Select this menu item to synchronize to the repository.
Edit location Url	Select this menu item to edit the selected repository location.
Discard location	Select this menu item to discard the selected repository location.

See Also

Procedures:

- [Using Subversion Integration](#)
- [Configuring Subversion Repository Location](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Update Project Dialog (Subversion)

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: 

VCS | Update Project

Ctrl+T/Command T

Context menu of a file or directory | Subversion | Update File/Directory

VCS | Subversion | Update File/Directory

Use this dialog box to update the local working copy of a file, directory, or project with a revision from the repository.

Item	Description
Update/Switch to specific Url	<ul style="list-style-type: none"> • Select this check box to synchronize your local working copy with a specific repository. Specify the source repository either in the URL text box through its full Url address or in the Use Branch text box through the branch name. • Clear this check box to bring the changes from the repository that corresponds to the current working copy.
Use Branch	<p>In this text box, specify the location of the required repository through its branch name. Click the Browse button  to choose the required branch in the Select Branch dialog box that opens.</p> <p>Note</p> <ol style="list-style-type: none"> 1. To use this method of specifying the required repository, you need to configure a list of branches you work with. If you have not done it yet, click Configure Branches in the Select Branch dialog box. 2. The text box is enabled only when the Update/Switch to specific Url check box is selected.
Url	<p>In this text box, specify the location of the required repository through its full URL address. Click the Browse button  to open the Select Repository Location dialog box.</p> <p>Note</p> <p>The text box is enabled only when the Update/Switch to specific Url check box is selected.</p>
Update/Switch to specific revision	<p>Select this check box to synchronize your local working copy with a specific revision different from the HEAD revision. The Update/Switch to specific revision text box becomes enabled.</p> <p>In this text box, specify the number of the revision to be used. Click the Browse button  to open the Changes Browser dialog box. By default, PhpStorm suggests to update your local working copy the HEAD revision. This option corresponds to the <code>--non-recursive (-N)</code> switch of the Subversion <code>updateupdate</code> command.</p>
Depth	<p>Use this drop-down list to specify the range of recursion into subdirectories. The available options are:</p> <ul style="list-style-type: none"> • Working copy - select this option to get files/directories from repository subtrees that have not been checked out yet. • Empty - select this option to involve only the current file. • Files - select this option to involve the files in the folder. • Immediates - select this option to involve direct children of the current file. • Infinity - select this option to enable full recursion.
Force Update	<p>Select this check box to have local files replaced with the files from the repository even if the local files have modifications and thus abandon the local modifications.</p>
Update administrative information only in changed subtrees	<p>Select this check box to have PhpStorm update the administrative information only for the subtrees to which changes have been applied.</p>
Do not show this dialog in the future	<p>Select this check box to perform update silently.</p>

See Also

Concepts:

- [Supported Version Control Systems](#)

Procedures:

- [Using Subversion Integration](#)

Reference:

- [File Status Highlights](#)
- [Update Info Tab](#)

External Links:

- <http://subversion.tigris.org/>
- <http://svnbook.red-bean.com/en/1.1/ch09.html>

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Perforce Reference

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

In this part:

- [Perforce Options Dialog](#)
- [Link Job to Changelist Dialog](#)
- [Edit Jobs Linked to Changelist Dialog](#)
- [Integrate File Dialog \(Perforce\)](#)

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Using Perforce Integration](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Perforce Options Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

[File](#) | [Settings](#) | [Project Settings](#) | [Version Control](#) | [Configure](#) | [Perforce for Windows and Linux PhpStorm](#) | [Preferences](#) | [Project Settings](#) | [Version Control](#) | [Configure](#) | [Perforce for Mac OS](#)

In this dialog box, configure connection to the specified Perforce server.

Item	Description
Perforce is online	Select this check box to enable establishing connection to the Perforce server. If there is a connection but the server does not respond (for example, because the server is backing up currently), you can disable such attempts by clearing this check box, or PhpStorm will suggest to do that after a timeout. After that the version control-specific operations will be disabled.
Use P4CONFIG or default connection	Use server information and user credentials from the P4CONFIG environment variable or default Perforce client connection. Otherwise, specify port, user, client, and password.
Charset	Select the charset corresponding to the one set on the server.
Dump Perforce commands to <i>IDEA_Home\bin\p4.output</i>	Log Perforce commands in the specified file.
Use login authentication	Toggle authentication.
Try to log in silently	Skip prompt dialog. This option works, when login authentication is required.
Use native Perforce API	Speed up connection to the server using a special library.
Path to P4 executable	Specify the path to the Perforce client executable file.
Test connection	Make sure that connection to the Perforce server is established.
Show branching history	See branches for files in the dialogs showing File History. When you work with several branches, it is recommended to enable this option so that the file branches are correctly displayed.

See Also

Procedures:

- [Debug Tool Window. Threads](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Link Job to Changelist Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

The dialog box opens when you click the  button in the [Edit Jobs Linked to Changelist](#) dialog box. Use this dialog box to search for available jobs, view their details, and link jobs to the changelist.

Tip

When you need to attach only one job to a changelist, you can use the [quick search](#) functionality. This requires that you know the exact name of the job or at least can specify a search pattern for it.

Specify search parameters

Use the controls in this area for specifying various criteria to limit the search output. Follow the Perforce jobs [syntax rules](#). The specified values are joined in the generated command line query via the AND operation.

Tip

At least one of the fields should be filled in.

Item	Description
Job name pattern	In this text box, type the desired job name search pattern.
Status	Use this drop-down list to specify the status of the job you are looking for. The available options are: <ul style="list-style-type: none"> • * - when this option is selected, all the jobs that match the remaining search criteria are displayed, regardless of their job statuses. <p>Tip</p> <p>If you specify Status as the only criterion and select this option, all the jobs that are currently present on the Perforce server will be retrieved.</p> <ul style="list-style-type: none"> • Open • Closed • Suspended
User name pattern	In this text box, specify the search pattern or the exact name of the user who created the desired job.
Date before/Data after	Use these text boxes to specify the time period the desired job is created in. The appropriate formats are <code>yyyy/mm/dd</code> or <code>yyyy/mm/dd:hh:mm:ss</code> .
Description pattern	In this text box, type the desired job description search pattern.
Search	Click this button to start searching for jobs that match the specified criteria.

Search results

Use this area to view the details of found jobs, select the desired job, and attach it to the changelist.

Item	Description
Search results	The list contains the jobs found according to the specified search criteria. When you select a job, the read-only area shows its details.
OK	Click this button to link the selected job to the changelist.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Attaching and Detaching Perforce Jobs to Changelists](#)
- [Using Perforce Integration](#)

Reference:

- [Local Tab](#)
- [Commit Changes Dialog](#)
- [Edit Jobs Linked to Changelist Dialog](#)

External Links:

- <http://www.perforce.com/perforce/doc.081/manuals/cmdref/jobs.html#1040665>

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>

- <http://youtrack.jetbrains.net/issues/WI>

Edit Jobs Linked to Changelist Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

VCS | [Show Changes View - Local View](#) | [Tool Windows](#) | [Changes - Local VCS](#) | [Commit Changes](#)

The dialog box opens in the following cases:

- When you select a changelist and then select **Edit Associated Jobs** from the context menu.
- When you click the  button in the **Perforce** area of the [Commit Changes](#) dialog box.

Use the dialog box to search for Perforce jobs, link jobs to the selected changelist, and detach currently linked jobs.

Item	Tooltip and Shortcut	Description
	Unlink selected jobs	Click this button to detach the selected job from the changelist.
	Search	Click this button to open the Link Job to Changelist dialog box, where you can search for available jobs, view their details, and link the desired job to the changelist.
	Find and link job matching the pattern	Click this button to start quick search for the job that matches the pattern specified in the text box and attach the job to the changelist. In the text box, specify the exact name of the desired job or a search pattern according to the Perforce jobs syntax rules .

Tip

If only one job matching the pattern is found, it is attached to the changelist automatically. Otherwise, to select a job among several available jobs, click  and find the desired job using the [Link Job to Changelist](#) dialog box.

Close Click this button to save the specified settings and leave the dialog box.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Attaching and Detaching Perforce Jobs to Changelists](#)
- [Using Perforce Integration](#)

Reference:

- [Local Tab](#)
- [Commit Changes Dialog](#)
- [Link Job to Changelist Dialog](#)

External Links:

- <http://www.perforce.com/perforce/doc.081/manuals/cmdref/jobs.html#1040665>

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Integrate File Dialog (Perforce)

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

VCS | [Integrate Project](#)
Context menu of a file or directory | [Perforce](#) | [Integrate File/Directory](#)

Use this dialog to integrate changelists from one branch spec to another.

Item	Description
Branch Spec	Select the branch spec that will be used for change integration. Consider the following: <ul style="list-style-type: none"> • If the Reverse option is enabled, changes are integrated from the selected branch to the local copy. • If the Reverse option is disabled, changes are integrated from the local copy to the selected branch.
Integrate changelist	Use this option to invoke the Changes Browser , where you can select the changelist that will be integrated into the current branch/local copy.
Store Changes To Changelist	Specify the changelist where the integrated changes should be stored.
Revert unchanged files before sync (p4 revert -a)	Select this option to revert unchanged files.
Run resolve automatically after the	Select this option to automatically resolve the files that can be resolved without conflicts.

sync (p4 resolve -am)

See Also

Procedures:

- [Integrating Perforce Files](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Apply Patch Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

VCS | [Apply Patch](#)

Use the dialog box to restore changes that were preserved in a patch file in the specified directory.

Item	Tooltip and Shortcut	Description
	Patch file name	In this text box, specify the name of the *.patch file to be applied. Type the fully qualified name manually or click the Browse button  and locate the desired patch file using Select Patch File dialog box, that opens.
	Group by Directory Ctrl+P+Meta P	Use this button to toggle between the flat view and the directory tree view. Select the check boxes next to the changes that you want to be applied.
	Expand All Ctrl+Add Command Add Or Ctrl+Equals Command Equals	Click this button to have all nodes expanded.
	Collapse All Ctrl+Subtract Command Subtract Or Ctrl+Minus Command Minus	Click this button to have all nodes collapsed.
	Select All Ctrl+A+Command A	Click this button to select all the files in the list or directory tree.
	Map base directory	Click this button to open the Select Path dialog box, where you can choose the directory relative to which file names in the patch file will be interpreted. You can map a base directory to a single file, directory, or to a selection.
	Show Differences Ctrl+D+Command D	Click this button to open the Differences Viewer that shows the differences between your local working copy, the repository version, and the patch. Use the buttons Compare Previous File  and Compare Next File  to have the files in patch compared in a chain.
	Tip	If the patch cannot be applied without conflicts, the lines with conflicts are highlighted red.
	Strip Directory	Use this button to have the changes applied to files located in other directories than specified in the patch. Clicking this button removes one slash in the path to the target file. Click the button as many times as many leading directories you need to strip. The number of removed slashes is indicated in square brackets.
	Restore Directory	Use this button to revert the last <code>strip directory</code> action. Click the button as many times as many previously stripped leading directories you need to restore.
	Reset Directories	Use this button to revert all the <code>strip directory</code> actions in the selection.
	Remove Directories	Click this button to have all the leading directories stripped and have the changes applied to the file with the specified name in the base directory.
	Summary	This section displays summary information for the currently selected change list (the number of modified, new, and deleted files).
	Existing Changelist	Select this option to have the patched files added to an existing changelist and select the desired changelist from the drop-down list.
	New Changelist	Select this option to have a new changelist created and add the patched files to it. <ul style="list-style-type: none"> • Name - in this text box, type the name of the new changelist. By default, the field is filled in with the name of the current patch. • Comment in in this text box, type the comment to the new changelist. • Make this changelist active - select this check box to have all the modified files automatically added the new changelist. • Track Context - select this check box to have PhpStorm reload the context of the task associated with the new changelist, when the changelist is activated.

See Also

Concepts:

- [Patches](#)
- [Changelist](#)

Procedures:

- [Applying Patches](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Enable Version Control Integration Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

VCS | [Enable Version Control Integration](#)

In this dialog box, choose one of the registered Version Control Systems to use in your project and assign it to the Project Root.

Note

The dialog box and the menu item are available only if the project does not use any Version Control System.

Item	Description
Please select version control system to make your <Project Root> under	In this drop-down list, select one of the registered Version Control Systems.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Procedures:

- [Associating a Project Root with Version Control System](#)

Reference:

- [Version Control](#)
- [Version Control Reference](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Commit Changes Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

VCS | [Commit Changes](#)

VCS | [Show Changes View - Local - Context menu of a file or change list - Commit Changes](#)

View | [Tool Windows](#) | [Changes - Local - Context menu of a file or change list - Commit Changes](#)

Use this dialog box to commit (check in) changes from the selected change list to the repository and optionally to create a patch file.

Note

The contents of the dialog box depend on the version control system used.

Toolbar

Item	Tooltip and Shortcut	Description	Available in
	Show Differences Ctrl+DCommand D	Click this button to open the Differences dialog box that points at the inconsistencies between your local working copy of the selected file and the file in the repository.	All VCSs
	Move to Another Changelist F6F6	Click this button to add the selected file(s) to another changelist. The Choose Changelist dialog box opens where you can select an existing changelist or create a new one.	All VCSs
	Refresh Changes Ctrl+F5Command F5	Click this button to reload the Changed files tree view so it is up-to-date.	All VCSs
	Rollback	Click this button to revert all the changes made to the local working copy of the selected file(s).	All VCSs
	Jump to source F4F4	Click this button to open the source code of the selected file in the editor.	All VCSs
	Revert Unchanged Files	Click this button to revert the files that have not been modified locally.	Subversion Perforce
	Group by Directory Ctrl+PCommand P	Click this button to toggle between the flat view and the directory tree view.	All VCSs
	Expand or collapse all nodes Ctrl+Add Command Add or Ctrl+Equals Command Equals Ctrl+Subtract Command	Click these buttons to expand or collapse all nodes in the directory tree. These buttons are not available in flat view.	All VCSs

	<p>Subtract Or Ctrl+Minus Command Minus</p> <p>Select All Ctrl+ACommand A</p>	<p>Click this button to select all the files in the list or directory tree.</p>	<p>All VCSs</p>
--	---	---	-----------------

Controls

Item	Description	Available in
Changed files	This tree view displays the list of changed files. Select check boxes next to the files to be checked in.	All VCSs
Comment	In this text box, describe the changes to be checked in. This comment will be also used as the name of the patch file, if you decide to create a patch. As you type, PhpStorm checks the spelling and highlights words in question. This functionality is available if the Spelling code inspection is enabled.	
Change list	From this drop-down list, select the change list that contains the modified files to be checked in or included in the patch. By default, the active change list is suggested.	All VCSs
Summary	This section displays summary statistics on the currently selected change list, such as the number of modified, new, and deleted files. The area also shows how many files of each type are shown and how many of them will be committed.	All VCSs
Author	Use this drop-down list to specify the person who created the changes to be checked in. Tip This may be necessary when you are checking in changes made by another person.	Git
Amend Commit	Select this check box to replace a previous commit with the current changes.	Git
Keep files locked	Select this check box to keep the files locked after they are checked in.	Subversion
Jobs	This area is available only if you select the Enable Perforce Jobs Support check box on the Perforce page of the Settings dialog box. Use the controls in this area to search for Perforce jobs , link jobs to the selected changelist, and detach currently linked jobs. <ul style="list-style-type: none"> • Unlink selected jobs button  - click this button to detach the selected job from the changelist. • Edit associated jobs button  - click this button to open the Edit Jobs Linked to Changelist dialog box where you can search for available jobs, view their details, and link the desired job to the changelist. • Find and link job matching the pattern button  - click this button to start quick search for the job that matches the pattern specified in the text box and attach the job to the changelist. In the text box, specify the exact name of the desired job or a search pattern according to the Perforce jobs syntax rules. Tip If only one job matching the pattern is found, it is attached to the changelist automatically. Otherwise, to select a job among several available jobs, click the  button and find the desired job using the Edit Jobs Linked to Changelist dialog box. Note The list box in the bottom of the area displays the jobs that are currently attached to the changelist.	Perforce
Before Submit/ Before Commit	Use this area to define which additional activities you want PhpStorm to perform before checking in the selected files. The available options are: <ul style="list-style-type: none"> • Optimize imports - select this check box to have redundant import statements removed. • Reformat code - select this check box to perform code formatting according to the Project Code Style settings. • Perform code analysis for affected files - select this check box to run code inspection on the files to be committed. • Check TODO (<filter name>) - select this check box to review TODO items. Click the filter icon  to specify which TODO items should be taken into account. If this check box is selected, PhpStorm will suggest you to review the encountered TODO items matching the specified filter. • Revert unchanged - select this check box to revert unchanged files. Note This option is only available for Perforce. • Update copyright - select this check box to add or update a copyright notice , according to the selected copyright profile - scope combination .	CVS Git, Subversion, Perforce
After Submit/ After Commit	Use this area to define which additional activities you want PhpStorm to perform after checking in the selected files. The available options are: <ul style="list-style-type: none"> • Upload files to: - in this drop-down list, specify the server access configuration to use for uploading the committed files to a local or remote host, a mounted disk, or a directory. To suppress uploading, choose None. • Always use selected server - select this check box to have PhpStorm by default upload files according to the server access configuration specified in the Upload files to: drop-down list. • Tag committed files - select this check box to have a tag assigned to the checked in files. In the text box, type the name of the tag. To have a previously assigned tag replaced with the new one, select the Override existing tags check box. Note This option is only available for CVS.	All VCSs
Details	The pane is by default hidden, to unfold it click the arrow icon next to the pane title. In this pane, explore the differences between the base repository version	All VCSs

of the file selected in the [Changed Files](#) list and the version to be checked in.

The pane consists of two areas:

- The affected code as it was in the base revision.
- The affected code as it is after the change is introduced.

The pane can be [split horizontally or vertically](#).

In each area, PhpStorm numbers both changes and the lines involved in them.

To close the pane, click the **Change Details** button  once more.

Item Tooltip and Shortcut Description

	More/Less Lines	Click this button to open a slider and specify the number of lines to be shown above and below the updated code fragment at the caret.
	Next Change	Click this button to move to the next updated piece of code.
	Previous Change	Click this button to return to the previous updated code fragment.
	Settings	Click this button to show the list of options that define the appearance of the pane: <ul style="list-style-type: none"> • Top/Bottom: Select this option to have the pane split horizontally, so the base revision is shown in the upper part and the locally updated version is shown in the bottom part of the pane. • Left/Right: Select this option to have the pane split vertically, so so the base revision is shown in the left-hand part and the locally updated version is shown in the right-hand part of the pane. • Use soft wraps: Select this option to have the <i>soft wraps</i> (or <i>word wraps</i>) used.

Submit/Commit Click this button with drop-down to do one of the following:

All VCSs

- **Submit/Commit:** select this option to check in the changes to your version control.
- **Commit and Push:** select this option to have the changes pushed to the remote repository immediately after commit.

Note

This operation is available when you are using [Git](#) or [Mercurial](#) integration.

- **Create Patch:** select this option to have PhpStorm generate a patch based on the changes to be checked in. In the **Create Patch** dialog box that opens, type the name of the patch file and specify whether you need a reverse patch.
- **Remote Run:** select this option to [run your personal build](#) .

Note

This operation is available when you are logged in to TeamCity. Refer to TeamCity plugin documentation for details.

See Also

Concepts:

- [Patches](#)
- [Version Control with PhpStorm](#)

Procedures:

- [Checking in Files](#)
- [Creating Patches](#)
- [Version Control with PhpStorm](#)
- [Deployment](#)

External Links:

- <http://www.perforce.com/perforce/doc.081/manuals/cmdref/jobs.html#1040665> 

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi> 
- <http://youtrack.jetbrains.com/issues/WI> 

Configure Ignored Files Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Changes tool window - 

Use this dialog box to create a list of files and directories that you do not want to put under version control. These can be file names associated with VCS administration, backup files, and any other artifacts that you want to remain unversioned. You can also specify patterns that define files or directories to ignore.

Item	Description
Add	Click this button to add an item to the list. The Ignore Unversioned Files dialog box opens where you can type an exact path to a file or directory to be ignored or specify a pattern that defines the names of files and directories to be ignored.
Edit	Click this button to edit the selected path or pattern in the Ignore Unversioned Files dialog box.
Remove	Click this button to remove the selected path or pattern from the list.

See Also

Concepts:

- [Version Control with PhpStorm](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Create Patch Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

VCS | Create Patch

View | Tool Windows | Changes - Local - Context menu of a file or change list - Commit Changes

Use this dialog box to have a patch file for the specified changelist or files generated.

Toolbar

Item	Tooltip and Shortcut	Description	Available in
	Show Differences Ctrl+DCommand D	Click this button to open the Differences dialog box that points at the inconsistencies between your local working copy of the selected file and the file in the repository.	All VCSs
	Move to Another Changelist F6F6	Click this button to add the selected file(s) to another changelist. The Choose Changelist dialog box opens where you can select an existing changelist or create a new one.	All VCSs
	Refresh Changes Ctrl+F5Command F5	Click this button to reload the Changed files tree view so it is up-to-date.	All VCSs
	Rollback	Click this button to revert all the changes made to the local working copy of the selected file(s).	All VCSs
	Jump to source F4F4	Click this button to open the source code of the selected file in the editor.	All VCSs
	Revert Unchanged Files	Click this button to revert the files that have not been modified locally.	Subversion Perforce
	Group by Directory Ctrl+PCommand P	Click this button to toggle between the flat view and the directory tree view.	All VCSs
	Expand or collapse all nodes Ctrl+Add Command Add or Ctrl+Equals Command Equals Ctrl+Subtract Command Subtract or Ctrl+Minus Command Minus	Click these buttons to expand or collapse all nodes in the directory tree. These buttons are not available in flat view.	All VCSs
	Select All Ctrl+ACommand A	Click this button to select all the files in the list or directory tree.	All VCSs

Controls

Item	Description	Available in
Changed files	This tree view displays the list of changed files. Select check boxes next to the files to be included in the patch.	All VCSs
Comment	In this text box, describe the changes to be included in the patch. As you type, PhpStorm checks the spelling and highlights words in question. Note This functionality is available if the Spelling code inspection is enabled.	
Change list	From this drop-down list, select the change list that contains the modified files to be included in the patch. By default, the active change list is suggested.	All VCSs
Summary	This section displays summary statistics on the currently selected change list, such as the number of modified, new, and deleted files. The area also shows how many files of each type are shown and how many of them will be included in the patch.	All VCSs
Before commit	Use this area to define which additional activities you want PhpStorm to perform before creating a patch based on the changes in the selected files. The available options are: <ul style="list-style-type: none"> • Optimize imports - select this check box to have redundant import statements removed. • Reformat code - select this check box to perform code formatting according to the Project Code Style settings. • Perform code analysis for affected files - select this check box to run code inspection on the changed files. • Revert unchanged - select this check box to revert unchanged files. Note This option is only available for Perforce. <ul style="list-style-type: none"> • Update copyright - select this check box to add or update a copyright notice, according to the selected copyright profile - scope combination. 	CVS Git, Subversion, Perforce

After Commit Use this area to define which additional activities you want PhpStorm to perform after creating a patch based on the changes in the selected files. The available options are: All VCSs

- **Upload files to:** - use this drop-down list to have the changed files uploaded to a local or remote server, a mounted disk, or a directory. Select the relevant [server configuration](#) from the list. To suppress uploading, choose **None**.
- **Tag committed files** - select this check box to have a tag assigned to the changed files. In the text box, type the name of the tag. To have a previously assigned tag replaced with the new one, select the **Override existing tags** check box.

Note

This option is only available for CVS.

Create patch Click this button to type the name of the patch file and specify whether you need a reverse patch generated in the **Create Patch** dialog box that opens. All VCSs

See Also

- Concepts:
- [Patches](#)
- Procedures:
- [Creating Patches](#)
- Web Resources:
- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>

File Status Highlights

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

In PhpStorm each file has its own status marked with a specific color. The file status denotes correspondence of the actual file content with the one marked as 'current'. In the editor, each line in a file is checked whether it corresponds to the state at the 'current' point and marked with a specific color at the left gutter area.

- Tip
- You can customize the default colors:
- For files - in the **File Status** page of the [Colors and Fonts](#) settings.
 - For the lines in the editor - in the **Diff** page of the [Colors and Fonts](#) settings.

- In this section:
- [File Status in Views.](#)
 - [Line Status in the Editor.](#)

File status in views

Color	File Status	Description
Black	Up to date	File is unchanged.  PhantomsTest.java
Gray	Deleted	File is scheduled for deletion from the repository.  privatent.txt
Blue	Modified	File has changed since the last synchronization.  Advice.java
Green	Added	File is scheduled for addition to the repository.  ResultTest.java
Violet	Merged	File is merged by your VCS as a result of an update.  VcsHistoryDialog.java
Brown	Unknown	File exists locally, but is not in the repository, and is not scheduled for adding.  Test.xml
Olive	Ignored	File will be ignored in any VCS operation.  Test.html
Light brown	Hijacked	File is modified without checkout . This status is valid for the files under Perforce, ClearCase and VSS.  HelpTOC.xml
Red	Merged with conflicts	During the last update, file was merged with conflicts.  HistoryTestCase.java
Lilac	Externally deleted	File is deleted locally, but was not scheduled for deletion, and still exists in the CVS repository.  test1.txt
Dark cyan	Switched	The file is taken from a different branch than the whole project. This status is valid for CVS and SVN.

Line status in the editor

Color	File Status	Description
	Modified	Denotes the lines modified since the last synchronization.
	Added	Denotes the lines added since the last synchronization.

Deleted Denotes the lines removed since the last synchronization.

See Also

Procedures:

- [Using Change Markers to View and Navigate Through Changes in the Editor](#)
- [Configuring IDE Settings](#)

Reference:

- [Editor. Colors and Fonts. Diff](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

New Changelist Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Changes tool window -

Alt+InsertCommand N

Use this dialog box to create a new changelist.

Item	Description
Name	Type the name of the new changelist.
Comment	Type optional comment. When the new changelist will be submitted to the repository, this comment will appear in the Comment text area of the Commit Changes dialog box.
Make this changelist active	If this check box is selected, the new changelist will automatically get the <code>active</code> status. See Changelist .
Track context	If this check box is selected, the current context will be preserved on changelist deactivation. On changelist becoming active, the context will be restored.

See Also

Concepts:

- [Changelist](#)

Procedures:

- [Creating a New Changelist](#)
- [Managing Tasks and Context](#)

Reference:

- [Changes Tool Window](#)
- [Tasks](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Revert Changes Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

VCS | Show Changes View - Local - Context menu of a file or change list - Revert Changes

View | Tool Windows | Changes - Local - Context menu of a file or change list - Revert Changes

Use this dialog box to roll back changes that have not yet been committed to the repository.

Toolbar

Item	Tooltip and Shortcut	Description
	Show Differences Ctrl+DCommand D	Click this button to open the Differences dialog box that points at the inconsistencies between your local working copy of the selected file and the file in the repository.
	Move to Another Changelist F6F6	Click this button to add the selected file(s) to another changelist. The Choose Changelist dialog box opens where you can select an existing changelist or create a new one.
	Group by Directory Ctrl+PMeta P	Click this button to toggle between the flat view and the directory tree view.
	Expand or collapse all nodes Ctrl+Add Command Add or Ctrl+Equals Command Equals Ctrl+Subtract Command Subtract or Ctrl+Minus Command	Click these buttons to expand or collapse all nodes in the directory tree. These buttons are not available in flat view.

Minus



Select All
Ctrl+ACommand A

Click this button to select all the files in the list or directory tree.

Controls

Item	Description
Changed files	This tree view displays the list of changed files. Select check boxes next to the files to be reverted.
Change list	Use the drop-down list to select the change list that contains the modified files to be reverted. By default, the active change list is suggested.

See Also

- Concepts:
- [Version Control with PhpStorm](#)

- Procedures:
- [Reverting Local Changes](#)

- Web Resources:
- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>

Select Target Changelist Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

VCS | Show Changes View - Repository / Incoming - Context menu of a file or change list - Revert Changes
View | Tool Windows | Changes - Repository / Incoming - Context menu of a file or change list - Revert Changes

Use this dialog box to roll back changes from a certain change list.

Item	Description
Existing Changelist	Select this option to select the target changelist from the existing ones.
New Changelist	select this option to define a new changelist.
Name	In this text box, type the new changelist name. This field is only available, when the New Changelist option is selected.
Comment	In this text box, type an optional description of the new changelist. This field is only available, when the New Changelist option is selected.
Make this changelist active	Select this check box to make new changelist active .

See Also

- Concepts:
- [Version Control with PhpStorm](#)

- Procedures:
- [Reverting Local Changes](#)

- Web Resources:
- <http://www.jetbrains.net/devnet/community/wi>
 - <http://youtrack.jetbrains.net/issues/WI>

Shelve Changes Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

VCS | Shelve Changes

Use this dialog box to put changes in the specified files or changelists to the shelf.

Toolbar

Icon	Tooltip and Shortcut	Description
	Show differences Ctrl+DCommand D	Click this button to have differences shown.
	Move to another changelist F6F6	Click this button to open the Choose Changelist dialog box where you can specify the changelist to move the selected file to.
	Group by Directory Ctrl+PMeta P	Use this button to toggle between the flat view and the directory tree view.
	Ctrl+Add Command Add or Ctrl+Equals Command Equals	Click this button to have all nodes expanded.
	Ctrl+Subtract Command	Click this button to have all nodes collapsed.

	Subtract Or Ctrl+Minus Command Minus	
	Rollback	Click this button to roll the changes back.
	Revert Unchanged Files	Click this button to have unchanged files reverted.
	F4F4	Click this button to open the source code of the selected file in the editor.
	Message History Ctrl+M	Click this button to reuse one of the previous commit messages (comments) from the list.

Controls

Item	Description
Changed files	This tree view displays the list of changed files. Check the files to be included in the patch.
Comment	Type the comment in the text area. This comment is used as the name of the new patch file.
Changelist	Use the drop-down list to select the changelist that contains the modified files to be included in the patch. By default, the active changelist is suggested.
Summary	This section displays summary information for the currently selected changelist (the number modified, new and deleted files).
Before commit	Use this section to define the following options:
Optimize imports	Check this option to remove redundant import statements.
Reformat code	Check this option to perform code formatting according to the Project Code Style settings .
Perform code analysis for affected files	Check this option to run code inspection on the affected files.
Shelve changes	Click this button to perform shelving and close the dialog.

See Also

Concepts:

- [Shelved Changes](#)

Procedures:

- [Shelving Changes](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Show History for File / Selection Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

VCS | [Local History](#) | [Show History](#)

VCS | [Local History](#) | [Show History for Selection](#)

Use this dialog box to explore changes to a file, or selection. The dialog box consists of two views:

- [History](#) view.
- [Differences](#) view.

Tip

The same dialog boxes are available on the context menu of a file or selected text in the editor.

History view

This view shows the list of revisions of a file, with the date and time when the revision has been stored. Some of the revisions are supplied with tags and labels.

Revisions are tagged automatically, for example, on opening a project, committing changes, or performing test. You can also [set your own labels](#).

Item	Description
	Click this button to revert the selected action.
	Click this button to create a patch based on the selected local version.
	Click this button to have the corresponding reference topic shown.

Tip

The same actions are available on the context menu of each revision.

Differences view

The Differences view is a powerful editor that supports [basic search and replace](#), [undo/redo actions](#), and [code completion](#).

If a revision is selected in the [History view](#), the left pane of the Differences view shows this read-only revision, with the differences against the current revision, which is displayed in the right pane. The current revision is editable.

Item	Shortcut	Description
	Ctrl+C Command C or Ctrl+Insert Command Insert	Click this button to copy the line at the caret or the selected fragment of text to the Clipboard.
	F7F7 / Shift+F7Shift F7	Use these buttons to move to the next / previous difference.
Ignore whitespace		Use this drop-down list to define how the differences viewer should treat white spaces in the text. <ul style="list-style-type: none"> • Do not ignore - when this option is selected, white spaces are considered unimportant and the differences are highlighted. • Leading and Trailing - select this option to have differences in the end and in the beginning of a line ignored. • All - when this option is selected, white spaces are considered unimportant regardless of their location in the source code.
		Use these buttons to apply differences.
Legend		This area shows summary information about the encountered differences: the number of differences found and the color map. The color map for the Differences viewer is configured in the Colors and Fonts page.

See Also

- Concepts:
- [Local History](#)
- Procedures:
- [Using Local History](#)
- Web Resources:
- <http://www.jetbrains.com/devnet/community/wi>
 - <http://youtrack.jetbrains.com/issues/WI>

Show History for Folder Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts:  

VCS | [Local History](#) | [Show History](#)

Use this dialog box to explore changes to a folder. The dialog box consists of two views:

- [History](#) view.
- [Changes](#) view.

Tip

The same dialog box is available on the context menu of a folder.

History view

The History view shows a list of folder revisions, each one being supplied with a time stamp, revision number, and an indication of the action that resulted in this state.

Item	Description
	Click this button to revert the selected action.
	Click this button to create a patch based on the selected local version.
	Click this button to have the corresponding reference topic shown.

Changes view

The Changes view shows a tree of changed files as per the selected version in the [History view](#).

Item	Shortcut	Description
	Ctrl+DCommand D	Click this button to show the differences between the current local version and the one selected in the History view. Alternatively, double-click the current local version in the Changes view.
		With a file, selected in the Changes view, click this button to roll back the selected action.
	Ctrl+PCommand P	Click this button to show changed files as a tree view of folders. If this button is not pressed, the files are shown as a flat list.
	Ctrl+Add Command Add or Ctrl+Equals Command Equals Ctrl+Subtract Command Subtract or Ctrl+Minus Command Minus	Click these buttons to have all nodes expanded or collapsed. These buttons are only available when the changed files are shown as a tree view.

-  **Ctrl+A Command A** Click this button to select all the files in the list or a tree view.
-  **Ctrl+F Command F or Alt+F3 Alt F3** In this area, type the search string. Note also that [speed search](#) is available in the Changes pane.
-  Click this button to clear the search area.

Tip
[Speed search](#) is available in the Changes view.

See Also

- Concepts:
- [Local History](#)
 - [Patches](#)
- Procedures:
- [Using Local History](#)
 - [Using Patches](#)
- Web Resources:
- <http://www.jetbrains.com/devnet/community/wi>
 - <http://youtrack.jetbrains.com/issues/WI>

Unshelve Changes Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

[Changes Tool window](#) | [context menu](#) | [Unshelve Changes](#)

Use this dialog box to restore shelved changes from a shelf to a change list.

Item	Description
Existing Changelist	Select this option to select the target changelist from the existing ones.
New Changelist	select this option to define a new changelist.
Name	In this text box, type the new changelist name. This field is only available, when the New Changelist option is selected.
Comment	In this text box, type an optional description of the new changelist. This field is only available, when the New Changelist option is selected.
Make this changelist active	Select this check box to make new changelist active .

See Also

- Concepts:
- [Shelved Changes](#)
- Procedures:
- [Shelving Changes](#)
- Web Resources:
- <http://www.jetbrains.com/devnet/community/wi>
 - <http://youtrack.jetbrains.com/issues/WI>

Visual SourceSafe Options Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

[File](#) | [Settings](#) | [Project Settings](#) | [Version Control](#) | [Configure](#) | [SourceSafe for Windows and Linux PhpStorm](#) | [Preferences](#) | [Project Settings](#) | [Version Control](#) | [Configure](#) | [SourceSafe for Mac OS](#)

Item	Description
Path to VSS client (ss.exe)	Specify executable file of the Visual SourceSafe client
Path to VSS configuration file	Specify configuration file <code>srcsafe.ini</code> . The configuration file <code>srcsafe.ini</code> can be located either locally (usually, when a local repository is used) or under a shared network folder. In the latter case, you should use the UNC names.
User name	Enter your VSS user name
Enter user named	Enter your VSS password
Working directories	View the mapping list between VSS project and local working directories
Add	Create a new VSS project and assign a working directory for it in the Add Mapping dialog.

In the VSS Project field, specify the directory location within the VSS repository, starting with \$/.

In the Working directory field, specify the local working folder.

- Edit** Change name or working directory of an existing VSS project in the Edit Mapping dialog.
- Remove** Delete the selected project from the list.

See Also

Procedures:

- [Using Visual SourceSafe Integration](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Table Editor

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

Data Sources tool window > select data source > select table > use one of the following:

- F4F4
-  on the toolbar.
- Open Table Editor in the context menu.

Use the toolbar buttons, context menu commands, and keyboard shortcuts available in the Table Editor to manipulate database records and perform associated tasks.

Toolbar buttons, context menu commands and keyboard shortcuts

Button	Command	Shortcut	Function
	First Page		Go to the first page.
			Note
			Table rows are distributed among <i>pages</i> which are shown one at a time. Each page accommodates the same number of rows. The number of rows per page is configurable .
	Previous Page	Ctrl+Alt+UpCommand Alt Up	Go to the previous page .
	Next Page	Ctrl+Alt+DownCommand Alt Down	Go to the next page .
	Last Page		Go to the last page .
	Reload Page	Ctrl+RCommand R	Refresh the current table view. This function may be used in order to: <ul style="list-style-type: none"> • Synchronize the data currently shown with the actual contents of the table in the database. • Apply the page size setting after its change.
	Add New Row	Alt+InsertCommand N	Add a new record to the table.
	Delete Selected Rows	Ctrl+YCommand Y	Remove the selected range of rows from the table.
	Filters and Ordering	Ctrl+F12Command F12	Open the Filters and Ordering dialog to specify sorting and filtering conditions.
	Properties		Open the Table Editor Properties dialog to set the number of rows per page or to specify data export format.
	Copy Query	Ctrl+Alt+Shift+CCommand Alt Shift C	Copy the query that generated the current table view to the clipboard.
	Copy	Ctrl+C Command C or Ctrl+Insert Command Insert	Copy the values currently selected in the table to the clipboard. The format of data placed to the clipboard is defined by the data export format .
	Save LOB As		Save the large object (LOB) currently selected in the table in a file.

See Also

Concepts:

- [Data Sources](#)

Procedures:

- [Manipulating Table Data in the Table Editor](#)
- [Accessing Data Sources Via the Database Console](#)
- [Data Sources](#)

Reference:

- [Data Sources Tool Window](#)
- [Database Console Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Filters and Ordering Dialog

Previous | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts:  

To access this dialog, do one of the following in the [Table Editor](#):

- Press `Ctrl+F12`/`Command F12`.
- Click  **Filters and Ordering** on the toolbar.
- Select **Filters and Ordering** from the context menu.

In this dialog you can:

- Specify which table columns should be shown or hidden.
- Define the order in which the records should be shown.
- Restrict the table records to be displayed by specifying conditions.

Item	Description
Include / Exclude	Select the check boxes to have the corresponding fields shown. Clear the check boxes for the fields that should be hidden. Note Alternatively, you can right-click the header row in the Table Editor and select which columns should be shown and which shouldn't.
Name	This column shows the field names (read-only).
Order	Select the desired sorting order (Ascending or Descending) and priority.
Filter	Specify filtering conditions for the corresponding fields.

See Also

Concepts:

- [Data Sources](#)

Procedures:

- [Manipulating Table Data in the Table Editor](#)
- [Data Sources](#)

Reference:

- [Table Editor](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Table Editor Properties Dialog

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

To access this dialog, do one of the following in the [Table Editor](#):

- Click  **Properties** on the toolbar.
- Select **Properties** from the context menu.

The dialog box contains the following tabs:

- [General](#). Use this tab to specify the number of records to be shown on one page.
- [Data Export](#). Use this tab to define the data export format.

General tab

Item	Description
Page Size	Type the maximum number of records to display per page. To set unlimited page size, type zero.

Data export tab

Item	Description
String Quotation	Type the symbols that should be used to enclose the exported string values.
Values Separator	Type the symbol to be used to separate exported values.
Include Table Header	Select this check box to include the table header (column names) in exported data.
Export Table Data	Select this check box to export the table data to the specified file.

Include Row Number Select this check box to have the row numbers included in exported data.

See Also

Concepts:

- [Data Sources](#)

Procedures:

- [Manipulating Table Data in the Table Editor](#)
- [Accessing Data Sources Via the Database Console](#)
- [Data Sources](#)

Reference:

- [Table Editor](#)
- [Data Sources Tool Window](#)
- [Database Console Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Regular Expression Syntax Reference

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

This section provides a brief summary of regexp syntax that can be helpful for creating search and issue navigation patterns.

Character	Description
\	Marks the next character as either a special character or a literal. For example: <ul style="list-style-type: none"> • <i>n</i> matches the character <i>n</i>. "\n" matches a newline character. • The sequence "\\" matches "\" and "\" matches "(".
^	Matches the beginning of input.
\$	Matches the end of input.
*	Matches the preceding character zero or more times. For example, "zo*" matches either z or zoo.
+	Matches the preceding character one or more times. For example, "zo+" matches zoo but not z.
?	Matches the preceding character zero or one time. For example, "a?ve?" matches the ve in never.
.	Matches any single character except a newline character.
(subexpression)	Matches <i>subexpression</i> and remembers the match. <p>If you need to use the matched substring within the same regular expression, you can retrieve it using the backreference (\$num, where num = 1..n).</p> <p>If you need to refer the matched substring somewhere outside the current regular expression (for example, in another regular expression in the Replacement field), you can retrieve it using the dollar sign (\$num, where num = 1..n).</p> <p>If you need to include the parentheses characters into the <i>subexpression</i>, use "\(" or "\)".</p>
x y	Matches either x or y. For example, "z wood" matches z or wood. "(z w)oo" matches zoo or wood.
{n}	<i>n</i> is a nonnegative integer. Matches exactly <i>n</i> times. For example, "o{2}" does not match the o in Bob, but matches the first two o's in foood.
{n,}	<i>n</i> is a nonnegative integer. Matches at least <i>n</i> times. For example, "o{2,}" does not match the o in Bob and matches all the o's in "fooooood." "o{1,}" is equivalent to "o+". "o{0,}" is equivalent to "o*".
{ n , m }	<i>m</i> and <i>n</i> are nonnegative integers. Matches at least <i>n</i> and at most <i>m</i> times. For example, "o{1,3}" matches the first three o's in "fooooood." "o{0,1}" is equivalent to "o?".
[xyz]	A character set. Matches any one of the enclosed characters. For example, "[abc]" matches the a in plain.
[^ xyz]	A negative character set. Matches any character not enclosed. For example, "[^abc]" matches the p in plain.
[a-z]	A range of characters. Matches any character in the specified range. For example, "[a-z]" matches any lowercase alphabetic character in the range a through z.
[^ m-z]	A negative range characters. Matches any character not in the specified range. For example, "[m-z]" matches any character not in the range m through z.
\b	Matches a word boundary, that is, the position between a word and a space. For example, "er\b" matches the er in never but not the er in verb.
\B	Matches a non-word boundary. "ea*r\B" matches the ear in never early.
\d	Matches a digit character. Equivalent to [0-9].
\D	Matches a non-digit character. Equivalent to [^0-9].
\f	Matches a form-feed character.
\n	Matches a newline character.
\r	Matches a carriage return character.
\s	Matches any white space including space, tab, form-feed, etc. Equivalent to "[\f\n\r\t\v]".
\S	Matches any nonwhite space character. Equivalent to "[^\f\n\r\t\v]".
\t	Matches a tab character.

<code>\v</code>	Matches a vertical tab character.
<code>\w</code>	Matches any word character including underscore. Equivalent to "[A-Za-z0-9_]".
<code>\W</code>	Matches any non-word character. Equivalent to "[^A-Za-z0-9_]".
<code>\$num</code>	Matches <i>num</i> , where <i>num</i> is a positive integer, denoting a reference back to remembered matches. For example, "(.)\1" matches two consecutive identical characters.
<code>\n</code>	Matches <i>n</i> , where <i>n</i> is an octal escape value. Octal escape values should be 1, 2, or 3 digits long. For example, "\11" and "\011" both match a tab character. "\0011" is the equivalent of "\001" &I. Octal escape values should not exceed 256. If they do, only the first two digits comprise the expression. Allows ASCII codes to be used in regular expressions.
<code>\x n</code>	Matches <i>n</i> , where <i>n</i> is a hexadecimal escape value. Hexadecimal escape values must be exactly two digits long. For example, "\x41" matches A. "\x041" is equivalent to "\x04" &I. Allows ASCII codes to be used in regular expressions.
<code>\\\$</code>	Escapes \$.

See Also

Reference:

- [Finding and Replacing Text in File](#)
- [Version Control](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Scope Language Syntax Reference

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

The scopes language syntax enables you to specify project scopes involved in dependency analysis.

Sets of classes

- Single class set is defined by a class name, i.e. `com.intellij.OpenApi.MyClass`.
- Set of all classes in a package, not recursing into subpackages, is defined by an asterisk after dot, for example: `com.intellij.OpenApi.*`.
- Set of all classes in a package including contents of subpackages, is defined by an asterisk after double dot, for example `com.intellij.OpenApi..*`

Modifiers

Location Modifiers

help you specify whether the desired set is located in the source files, library classes or test code in the form of location modifiers `src:`, `lib:`, `file:`, or `test:`. For example, the following scope `src:com.intellij.OpenApi.*` implies all classes under the source root in the `com.intellij.OpenApi` package, excluding subpackages.

Module Modifiers

help you narrow down the scope by specifying the name of the related module in one of the following ways:

```
src[module name]:<E>
lib[module name]:<E>
test[module name]:<E>
```

For example, the following scope `src[MyJavaModule]:com.intellij.OpenApi.*` implies all classes under the source folders related to the module `MyJavaModule` in the package `com.intellij.OpenApi` excluding subpackages.

Logical operators

The language allows you to use common logical operators like AND (`&&`), OR (`||`) and NOT (`!`).

See Also

Procedures:

- [Validating Dependencies](#)
- [Analyzing Duplicates](#)

Reference:

- [Scopes](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

Diagram Reference

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

In this section:

- [Diagram Toolbar and Context Menu](#)
- [Diagram Preview](#)
- [General Techniques of Using Diagrams](#)
- [UML Class Diagram Toolbar and Context Menu](#)

See Also

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Diagram Toolbar and Context Menu

Previous | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

In this section:

- [Toolbar](#)
- [Context menu](#)

Toolbar

Item	Description
------	-------------

	If this button is pressed, you can move the entire diagram image. If this button is released, only the selected node is moved.
	Click this button to show grid in the diagram background.
	Click this button to align elements against the grid.
	Click this button to increase the scale of the diagram. Alternatively, press NumPad+NumPad .
	Click this button to decrease the scale of the diagram. Alternatively, press NumPad-NumPad-.
	Click this button to restore the actual size of the diagram.
	Click this button to make the contents fit into the current diagram size.
	Click this button to apply the current layout, selected on the context menu of the diagram.
	Click this button to save the current diagram in the specified location as xml file.
	Click this button to save the diagram in an image file with the specified name and path. The possible formats are:jpeg, png, svg, svgz, or gif.
	Click this button to print the diagram.
	Click this button to open the diagram preview in a separate frame, where you can configure the page layout, scale, and headings information.

Context menu

Note

The table below contains commands that are not available from the toolbar.

Item	Description
------	-------------

New	Use this node to add new elements to a diagram.
Refactor	This node contains refactoring commands, enabled in the current context.
Jump to Source	Choose this command to open the selected diagram node element in the editor.
Find Usages	Choose this command to search for usages of the selected node element.
Layout	Select the desired diagram layout from the submenu.
Show Edge Labels	Check this command to show multiplicities in diagram.

See Also

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

```
ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js'; var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s); }());
```



PhpStorm 3.0.0 Web Help

Diagram Preview

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Use the Structure view as a Preview, that allows to get a "10,000-feet" look on a diagram. The shadow area represents the visible part of a diagram. As you zoom in or out, or change shape of the PhpStorm windows, the size of the shadow area changes accordingly.

To enable diagram preview

- Open the Structure tool window.

With the preview pane, the following actions are available:

- Keeping the mouse button pressed, move the shadow area to obtain the desired view.
- Select one or more nodes in diagram, and the corresponding nodes in the Preview are marked dark gray.

See Also

Reference:

- [Diagram Toolbar and Context Menu](#)
- [UML Class Diagram Toolbar and Context Menu](#)
- [Structure Tool Window](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi/>
- <http://youtrack.jetbrains.com/issues/WI>

General Techniques of Using Diagrams

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

In this section:

- [Selecting elements in diagram.](#)
- [Managing diagram layout.](#)
- [Zooming in and out.](#)
- [Using the magnifier tool.](#)
- [Navigating to source code.](#)
- [Invoking refactoring commands.](#)
- [Finding usages of the selected node element.](#)

To select elements in diagram

- To select an element, just click it in diagram.
- To select multiple adjacent elements, keep **Shift** pressed and click the desired elements, or just drag a *lasso* around the elements to be selected.
- To select multiple non-adjacent elements, keep **Ctrl+Shift** pressed and click the desired elements.
- To select a member of a node element, double-click the node element, and then use the arrow keys, or the mouse pointer.

To manage diagram layout

- Right-click the diagram background, and choose **Layout** command of the diagram context menu. Next, select the desired layout from the submenu.
- Use **Drag-and-drop** technique to lay out entities in diagram manually.
- Apply the current layout selected on the context menu of the diagram, by clicking .

To zoom in and out, do one of the following

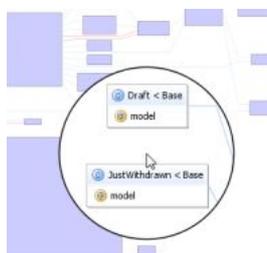
- Use the  and  toolbar buttons.
- Keeping **Ctrl** key pressed, rotate your mouse wheel up or down.
- Press **NumPad+** or **NumPad-**.

Tip

As you zoom in or out, the size of the shadow area in the diagram preview changes accordingly.

To use the magnifier tool

- Keep the **Alt** key pressed, and hover your mouse pointer over the most interesting, or problematic areas of the diagram.



To jump from an element in diagram to the underlying source code

1. Select an element in diagram.
2. Do one of the following:
 - On the context menu of the diagram, choose **Jump to Source**
 - Press **F4**.
 - Double-click selected element.

The source code of the corresponding source file opens in a separate tab in the editor.

See Also

Reference:

- [Diagram Toolbar and Context Menu](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

UML Class Diagram Toolbar and Context Menu

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#) | Shortcuts: Mac

In this section:

- [Toolbar](#)
- [Context Menu](#)

Toolbar

Item	Description
	Click this button to show fields in the class nodes.
	Click this button to show constructors in the class nodes.
	Click this button to show methods in the class nodes.
	Click this button to enable creating <code>extends</code> or <code>implements</code> links between node elements. If this button is not pressed, links cannot be drawn.
	Click this button to increase the scale of the diagram, or press <code>NumPad+NumPad .</code>
	Click this button to decrease the scale of the diagram, or press <code>NumPad-NumPad-</code> .
	Click this button to restore the actual size of the diagram.
	Click this button to make the contents fit into the current diagram size.
	Click this button to apply the current layout, selected on the context menu of the diagram, or press <code>F5F5</code> .
	Click this button to save the current diagram as a <code>*.uml</code> file.
	Click this button to save the diagram in an image file with the specified name and path. The possible formats are: <code>jpeg</code> , <code>png</code> , <code>svg</code> , <code>svgz</code> , or <code>gif</code> .
	Click this button to print the diagram.
	Click this button to open the diagram preview in a separate frame, where you can configure the page layout, scale, and headings information.

Context menu

Note

This section describes only those context menu commands that are not available from the toolbar.

Item	Shortcut	Description
Add class to diagram	<code>SpaceSpace</code>	Choose this command to add existing class to the diagram background.
Collapse nodes	<code>CC</code>	Choose this command to show the containing package of the selected node.
Expand nodes	<code>EE</code>	Choose this command to show class diagram of the selected package.
New	<code>Alt+InsertCommand N</code>	Choose this command to create a new node element or member .
Refactor		Point to this node to select one of the refactoring commands available in this context.
Analyze		Point to this node to select one of the code analysis commands available in this context.

See Also

Procedures:

- [General Techniques of Using Diagrams](#)
- [UML](#)

External Links:

- <http://www.omg.org>

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Icons Reference

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

- [File Types Recognized by PhpStorm](#)
- [Symbols](#)

See Also

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

File Types Recognized by PhpStorm

Previous | [Next](#) | [See Also](#) | [Comments](#)

PhpStorm recognizes numerous file types and provides extensive support. Each file type is denoted with a special icon. Custom files types are also allowed. Each file type is associated with one or more extensions that match a certain pattern.

The file types and their extensions are configurable in the [File Types](#) dialog.

The default types include:

File Type	Icon
ActionScript files	
Active Server Pages files	
Apache Config files	
Archive files	
AspectJ files	
C# files	
C/C++ files	
Command Shell files	
CSS files	
CoffeeScript files	
Erlang files	
Files marked as plain text	
Files opened in associated applications	
HAML files	
HTML files	
IDL files	
Image files	
JavaFX files	
JavaScript files	
JSON files	
JSTestDriver Config files	
Java Server Pages files	
JSPx files	
LESS files	
Patch files	
Perl files	
PHP files	
Regular expressions	
RELAX NG Compact Syntax	
SASS files	
SCSS files	
Smarty, Smarty config files	
Text files	
XHTML files	
XML DTD files	
XML files	
YAML files	

See Also

Procedures:

- [Creating and Registering File Types](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Symbols

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

In this section:

- [Icons in the tree view](#)
- [Toolbar](#)
- [Data Sources](#)

Icons in the tree view

Icon	Description
	Class
	Final class
	Method/function
	Variable
	Constant
	Field
	Parameter
	Element
Visibility modifiers	
	Private
	Protected
	Public

Toolbar

Icon	Tooltip	Description
	Sort alphabetically	Toggle this button to have the elements within a class sorted alphabetically.
	Group methods by defining type	Toggle this button to have all the methods that override/implement the methods of a particular class grouped under the node that corresponds to this class/interface.
	Show fields	Toggle this button to have fields shown in the structure view.
	Show constants	Toggle this button to have constants shown in the structure view.
	Show inherited	Toggle this button to have PhpStorm show all the methods and fields inherited by the current class and accessible from it. The inherited members are displayed gray to distinguish them from the members defined in the current class.
	Show includes	Toggle this button to have included files shown in the tree.
	Expand All	Toggle this button to have all nodes in the tree expanded.
	Collapse All	Toggle this button to have all nodes in the tree collapsed.
	Autoscroll to Source	Toggle this button to enable automatic navigation to the line of source code that corresponds to the selected node when the focus switches to the editor.
	Autoscroll from Source	Toggle this button to have PhpStorm automatically move the focus in the Structure tool window to the node that corresponds to the code where the cursor is currently positioned in the editor.

Data sources

Icon Description

	DB data source
	DDL data source
	Schema
	Table
	Column
	Column with a primary key
	Column with a foreign key
	Column with an index
	Primary key

-  Foreign key
-  Index

 [Database connection](#) node for a data source.

PhpStorm establishes a database connection automatically first time it needs to execute a query, and closes it automatically upon the session end. If necessary, a connection can be terminated manually, using the toolbar button .

See Also

Procedures:

- [Auto-Completing Code](#)

Reference:

- [File Types](#)
- [Structure Tool Window](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

IntelliLang

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

The **IntelliLang** plugin offers a number of features related to the use of custom languages in PhpStorm.

- [Overview](#)
- [General Usage](#)
- [Code Inspections](#)
- [Quick Edit Language](#)
- [Usage Examples](#)
- [IntelliLang Configuration](#)

See Also

Concepts:

- [Using Language Injections](#)

Reference:

- [Language Injections](#)
- [IntelliLang Configuration](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Overview

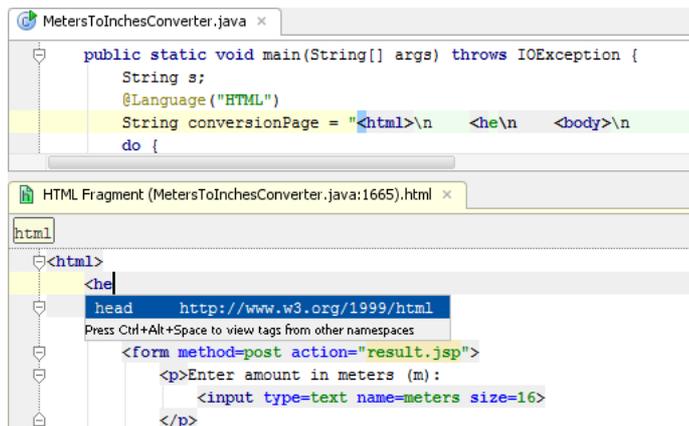
[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

IntelliLang is a combination of three basic kinds of functionality that are meant to support the developer in dealing with certain tasks that relate to (custom) languages in PhpStorm:

- Language Injection:** Editing the code that is embedded into String literals and XML fragments natively.

Note

You can open and modify an injected language code fragment in the [editor](#). In this way you get all the necessary coding assistance, as if you were working with the source code in the corresponding language.



To open an injected language code fragment in the editor, use the `Edit <Language> Fragment` [intention action](#).

- **Pattern Validation:** Provides assistance in making sure that Strings being passed to and from methods match a particular regular expression
- **Regular Expression Support:** A custom language implementation for regular expressions

Language injection

This makes use of the new possibilities of PhpStorm to treat String literals, XML text and attributes as fragments of an arbitrary language (called *Language Injection*). The plugin makes this newly introduced API readily available to everybody for their daily use through two very simple means: Either by using some Java annotations to mark String fields, local variables, method parameters and methods returning Strings as containing a certain language, or by just using a simple UI configuration. There is a set of annotations provided by the plugin, but the actual annotations are freely configurable to avoid any unwanted dependencies.

This enables the developer to get the benefit of a wide range of edit-time features, such as syntax error highlighting, completion, inspections and a lot more while editing fragments of e.g. JavaScript inside regular Java code or in XML files of custom schemas that PhpStorm usually doesn't know about.

Pattern validation

Additionally, the plugin allows to annotate Java elements of type String to have them checked for compliance with certain Regular Expressions. This can be useful for very simple *languages* where the developer needs to make sure that an expression conforms to a certain syntax, e.g. that a String is a legal Java identifier or a valid printf-like pattern used by `java.util.Formatter`.

This can both be validated on-the-fly while editing the code as well as during runtime (method parameters and return value only, like the `@NotNull` instrumentation of PhpStorm core) by instrumenting the compiled classes with assertions that match the value against the supplied pattern.

Regular expression support

This part of the plugin implements language-support for `java.util.regex.Pattern` and has been mainly created to support the IntelliJLang plugin by adding support for the micro-language that is probably one of the most often used one inside Strings. It features complete support for the syntax of the SDK's regular expression implementation and adds some further features, such as

- Completion and validation for character property names (e.g. `\p{javaJavaIdentifierStart}`) which nobody can usually remember anyway
- Validation and navigation for the use of back-references (e.g. `\1`), e.g. `ctrl-b` navigates to the capturing group the backref refers to.
- Intention Actions to simplify usages of repeated character occurrences, e.g. `a{0,1}` is offered to be converted to `a?`
- "Surround With" capturing/non-capturing group
- and more

See Also

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi> 
- <http://youtrack.jetbrains.net/issues/WI> 

General Usage

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

The plugin's usage is simple and straightforward, yet very flexible. Either add the provided set of annotations to your project and start using them, configure the plugin to use a custom set of annotations or simply use the UI configuration to make PhpStorm learn that e.g. the String argument of `Pattern.compile()` should be treated as a regular expression.

Using the annotations

IntelliLang makes use of three base-annotations: `@Language`, `@Pattern` and `@Subst`

- `@Language` is responsible for the Language injection feature
- `@Pattern` is used to validate Strings against a certain regular expression pattern
- `@Subst` is used to substitute non-compile time constant expressions with a fixed value. This allows to validate patterns and build the prefix/suffix of an injected language (see below) even for non-constant expressions that are known to contain certain kinds of values during runtime.

The annotations supplied with IntelliLang are located in the file `annotations.jar` which can be found in `%IDEA-CONFIG%/plugins/IntelliLang/lib`.

@language

The `@Language` annotation can be used to annotate String fields, local variables, method parameters and methods returning Strings. This will cause String-literals that are assigned to a field/variable passed as a parameter or are used as a method's return value to be interpreted as the specified language.

Additionally, there's the **Language Mismatch** inspection which checks for clashes between the *expected language* and the *actual language* when a field/variable is assigned or a value returned from method.

The plugin supports *direct* and *indirect* annotations, i.e. you can either directly use the annotation like this:

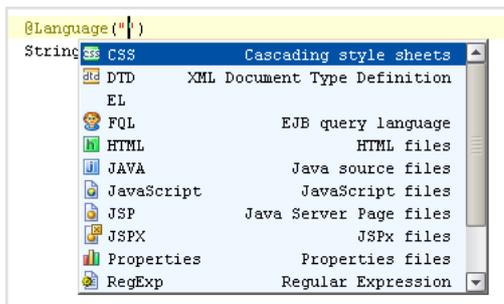
```
@Language("JavaScript")
String code = "var x = 1 + 2";
```

or annotate another annotation class like this:

```
@Language("XPath")
public @interface XPath { }
```

which can then simply be used to annotate elements as `@XPath`.

It's very easy to obtain the language-id that must be supplied as the annotation's value attribute: IntelliLang provides the list of available language via the regular code-completion action. Just select the appropriate language from the `ctrl-space` pop-up window:



@pattern

The @Pattern annotation is responsible for marking Strings that have to comply with a certain regular expression and can be used in just the same way as the @Language annotation. That means, it's possible to create derived annotations, such as an @Number annotation that requires a String to consist of one or more digits:

```
@Pattern("\\d+")
public @interface Number { }
```

In fact, the predefined annotations already contain two of such derived annotations: The first one, @PrintFormat, matches the printf-like pattern used by java.util.Formatter and another one, @Identifier, describes a valid Java identifier. These are ready to use without having to specify any additional pattern.

@subst

The @Subst annotation is used to substitute references that are not compile-time constant which enables the plugin to do **Pattern Validation** based on the assumption that the substituted value is compatible with the values that are expected during runtime. The plugin will complain if the value does not match the expected pattern.

It also helps to build a valid context of prefix/suffix (see next section) for the Language Injection feature. Consider this example:

```
@Subst("Tahoma")
final String font = new JLabel().getFont().getName();

@Language("HTML")
String message = "<html><span style='font: " + font + "; font-size:smaller'>" + ... + "</span></html>";
```

Without substituting the value of the variable font with the value Tahoma (actually it could just be a single character here), the injected fragment would be syntactically incorrect, causing the error a term expected to be displayed after the "font:" instruction.

Supplying context: prefix and suffix

When annotating an element, it's possible to supply a prefix and a suffix that should be prepended/appended when the language fragment is being parsed. This can be used to supply context information, i.e. if the prefix for a JavaScript injection is "var someContextVariable;", PhpStorm will know that the variable someContextVariable is declared and will not warn about it being undeclared when it's used.

Apart from the manual possibility to supply a prefix and suffix, IntelliJLang dynamically determines those values from the context a String literal is being used in:

```
@Language(value =JavaScript, prefix = "function doSomethingElse(a){}")
String code = "function doSomething() {\n" +
"  var x = 1;\n" +
"  doSomethingElse(x);\n" +
"}";
```

In this example, the JavaScript language will be injected into each of the three String literals, and each one's prefix and suffix will be calculated from the preceding and following expressions so that the resulting text that's being parsed is valid JavaScript syntax:

- no "missing '}'" error will be displayed: The closing brace is part of the first literal's suffix.
- the variable x that is used in doSomethingElse(a); will be declared: Its declaration is part of the second literal's prefix
- the function doSomethingElse() will be known as well: It's defined in the statically supplied prefix

Warning

There are some issues with the PhpStorm Language Injection API that impose certain restrictions on the prefix/suffix of an injected language fragment. For instance, it's not allowed that a token of the language spans across the prefix/suffix of an element. This could e.g. happen if the prefix ends with a whitespace character and the fragment starts with whitespace. The plugin deals with this special situation in the way that it trims the prefix/suffix and inserts exactly one space character as a separator. However, this doesn't work if a space character is no token separator, which e.g. applies to JavaScript string literals. Such cases cannot be automatically dealt with and PhpStorm core will produce an assertion.

Note

Even though the dynamic prefix/suffix calculation provides a proper context for the language fragment, some things may not work as expected. Most notably this are the refactoring (rename) and navigation functions. See also the known issues below.

See Also

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Code Inspections

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

IntelliLang contains a set of inspections that validate the correct use of the supplied annotations (or custom configured ones). These inspections can be configured through the regular Settings | Error configuration dialog.

Language injection

The inspections in this category apply to the language injection feature related to the `@Language` annotation.

- [Unknown Language ID](#)
- [Language Mismatch](#)
- [Injection not applicable](#)

Unknown language id

This inspection provides validation for the use of non-existing Language-IDs. It flags usages of incorrect values of the `@Language` value attribute, such as `@Language("NonExistingID")`.

Language mismatch

Validation for using references to elements that are annotated as containing different languages or are not annotated at all. The inspection offers a Quick-Fix to annotate such elements with the right annotation for the expected language.

Injection not applicable

This inspection checks whether an `@Language` or any derived annotation is used for anything other than elements of type `String` or `String[]`.

Pattern validation

This category contains inspections about validating the use of the `@Pattern` or its derived annotations.

- [Validate Annotated Patterns](#)
- [Pattern Annotation not applicable](#)
- [Non-annotated Method overrides @Pattern Method](#)

Validate annotated patterns

This inspection validates that expressions (String literals, as well as other compile-time constant or substituted expressions) match the pattern required by the `@Pattern` annotation. The inspection has an option to ignore non-constant expressions that contain non-substituted references and offers a Quick-Fix to add a substitution where applicable.

Pattern annotation not applicable

Checks whether a pattern-validation annotation (`@Pattern` or derived ones) is valid to be applied to the annotated element. Only elements of type `String` may be annotated.

Non-annotated method overrides @pattern method

This inspection checks whether a method without any `@Pattern` or derived annotation overrides an annotated method from its base classes. This is not necessary for the error-highlighting inside the editor, however the runtime-check instrumentation doesn't pick up annotations from base-class methods.

A Quick-Fix is provided to add an annotation that matches the one from the base-class method. This ensures that the runtime-check instrumentation works correctly.

See Also

Concepts:

- [Code Inspection](#)

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

Quick Edit Language

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

This is an experimental feature that can be extremely helpful, especially for editing Regular Expression patterns inside String literals due to the *double escaping* requirement. For example, in a regular expression, a literal backslash character has to be written as a double backslash and each of them has to be escaped with another backslash when written inside a String literal.

The Quick Edit function that appears as an Intention Action for any injected language fragment displays a pop-up dialog that allows to edit the string's value without the double escaping requirement. The dialog also shows the particular prefix/suffix of the fragment in a non-editable area.



The pop-up window can be dismissed by pressing Escape or clicking somewhere outside the pop-up window. Any changes made in the pop-up window are committed by pressing Ctrl-Enter.

Please note that the formatting of the non-editable prefix/suffix may differ from the actual value due to some problems. However, even though the formatting/alignment of the editable text may differ from the expected text, making changes to the text still works as expected.

Warning

Accepting Inspection Quick-Fixes inside the QuickEdit pop-up window may cause strange things to happen when the Quick-Fix attempts to open an Editor for the element being edited. The *Create Method* Quick-Fix of the JavaScript language is an example for that.

See Also

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Usage Examples

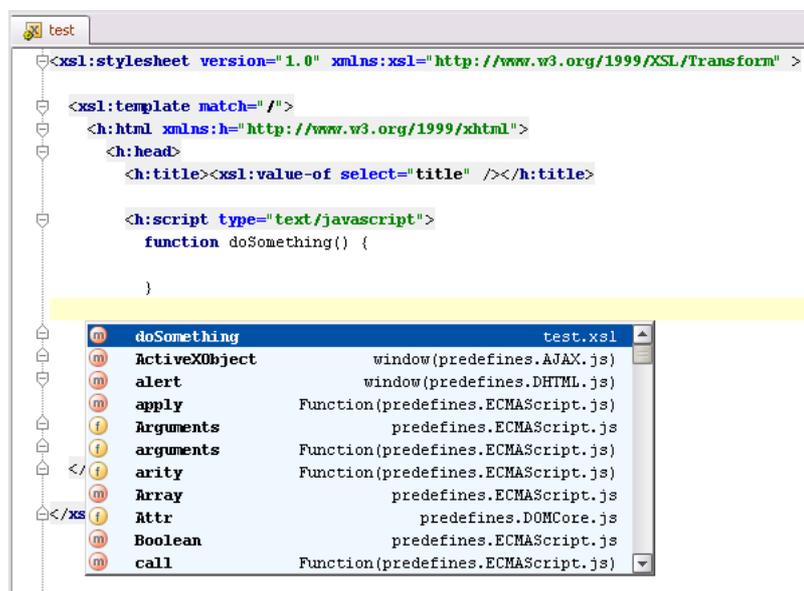
[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

Among the rather simple use-cases, like having detailed syntax checks for all kinds of "micro-languages" that are used e.g. in `Pattern.compile()`, `XPath.compile()` and so on, here are some less obvious, yet very useful examples how IntelliLang can leverage PhpStorm's support for a much better coding assistance.

The new scripting support in the upcoming Java 1.6 is another case where it will be important to get as much as possible edit-time assistance when script-code is constructed from Java code.

Extended JavaScript support

When dealing with JavaScript that's not directly embedded inside an HTML page, PhpStorm usually just treats it as plain text. Consider the following example that creates an HTML page from an XSLT script. Without the JavaScript language being injected into the `script` tag with the XHTML namespace as shown in the screenshot below, this would be treated as plain text, with no further code assistance.



Support for jsp custom tags

With IntelliLang it's also possible to make the content and attributes of custom JSP tags being treated as another language. This can be useful e.g. for server-side scripting using JavaScript or any other Language implementation available for PhpStorm.

One thing that's important to know is that the taglib's URI which supplies a custom tag should be used as the namespace URI of the XML tag to inject a language into. The namespace-textfields contain a list of all known taglib URIs in the project.

Warning

Unfortunately, at the moment the support for refactoring and navigation inside JSP custom tags seems to be broken and attempting to use code completion may result in exceptions thrown in PhpStorm core. See also the known issues below.

Pattern validation

Here's an obvious example right from PhpStorm's OpenAPI:

```
/** com.intellij.codeInspection.LocalInspectionTool
 *
 * @return descriptive name to be used in suppress comments and annotations,
 *         must consist of [a-zA-Z_0-9]+
 */
@Nonnull @NotNull public String getID() {
    return getShortName();
}
```

The contract of the method `getID()` is that it should only return strings that match the pattern "[a-zA-Z_0-9]+". The short note in the JavaDoc can be easily overlooked though because the contract isn't specified in an automatically verifiable way.

However, if this method were annotated as `@Pattern("[a-zA-Z_0-9]+")`, any attempt to return a string that doesn't match that pattern would be flagged in the editor:

```

public abstract class CheckedLocalInspectionTool extends LocalInspectionTool {
    @NonNls @NotNull
    @Pattern("[a-zA-Z_0-9]+")
    public abstract String getID();
}

class MyLocalInspectionTool extends CheckedLocalInspectionTool {
    @NonNls @NotNull
    @Pattern("[a-zA-Z_0-9]+")
    public String getID() {
        return "an incorrect id";
    }
}

```

Expression 'an incorrect id' doesn't match pattern: [a-zA-Z_0-9]+

Pattern completion

If a regular expression pattern represents an enumeration of different literal values, the plugin offers completion for those values:

```

void test(@Pattern("abc|xyz|123|456") String s) {...}

void doTest() {
    test("")
}

```

Completion list: 123, 456, abc, xyz

Regular expression editing

Here are some examples of the enhanced coding support for regular expression patterns:

Backref Validation

```

@Language("RegExp")
String s = "(abc)\\1\\2";

```

Unresolved backreference

Surround With

```

@Language("RegExp")
String s = "ab"

```

Surround With menu: 1. Capturing Group (pattern), 2. Non-Capturing Group (?pattern)

Character Category Validation

```

@Language("RegExp")
String s = "\p{xxx}";

```

Unknown character category

Character Category Completion

```

@Language("RegExp")
String s = "\p{}";

```

Completion list: all (ALL), Alnum (Alphanumeric characters), Alpha (Alphabetic characters), ASCII

See Also

Web Resources:

- <http://www.jetbrains.com/devnet/community/wi>
- <http://youtrack.jetbrains.com/issues/WI>

IntelliLang Configuration

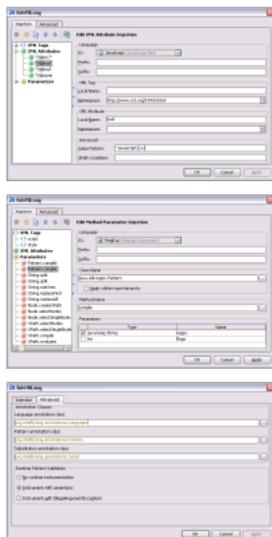
[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

The configuration dialog provides the following options:

- Language injection for XML text
- Language injection for XML attributes
- Language injection for method parameters
- Advanced settings that allows to change the names of the recognized base-annotations and the way how the pattern-validation is performed during runtime.

Screen shots

The screen shots below show some of the configuration options that can be tweaked and will be described in the following sections.



Language injection

This tab allows to configure the language injection feature for XML text, attributes and method parameters. Use the buttons in the toolbar or the context menu actions to add, remove, copy or import new entries. To add a new entry, the appropriate group has to be selected first.

Reordering the entries is possible with the **Move Up** and **Move Down** buttons. This can be important to define the precedence of different XML-injection entries. If an entry matches, no more injections are applied unless the injection specifies a [value pattern](#).

- [XML Text](#)
- [XML Attributes](#)
- [Advanced XML Options](#)
- [Method Parameters](#)
- [Importing Configuration Entries](#)

XML text

After adding a new XML text injection, select the ID of the Language to inject and optionally specify the prefix/suffix that make up the injection's context.

In the *XML Tag* pane, specify the local name (i.e. the name *without* any namespace prefix) and the namespace URI of the XML tag surrounding the text that should be treated as the selected language. The *name* field should not be empty, however the *namespace* field is optional.

The *Local Name* field takes a regular expression which makes it is possible to specify e.g. multiple tag names (`name1|name2`), case-insensitive names (e.g. `(?)tagname` matches `tagname` as well as `TagName`), etc. Be sure not to enter any whitespace characters as they would be significant for the match.

XML attributes

This is similar to the XML text configuration. However, it is possible to leave the *name* of the *XML Tag* empty which means that the configuration will apply to any attribute that matches the configured name, regardless of its containing XML tag.

The attribute's name takes the local name of the attribute and is also specified as a regular expression. This can be e.g. used to match e.g. HTML event handler attributes by specifying the name `"on.*"`. The attribute name may also be empty (unless the tag name is empty as well) which means that the configuration applies to all attributes of the containing tag.

Advanced XML options

The *Advanced* pane for XML Text and Attributes allows an even more fine-grained control over the injection process.

- [Value Pattern](#)
- [XPath Condition](#)

Value pattern

This field takes a regular expression that determines what part of the XML text's or attribute's value the language should be injected into. This can be used to inject the language only into values that match a certain pattern or to inject it into multiple parts that match the pattern. This is done by using the first capturing group of the pattern as the target for the injection.

Examples:

```
[${#} \\{ (.*?) \\}
```

This matches the pattern used by the JSP/JSF Expression Language.

```
^javascript: (.*)
```

This matches the *javascript*-protocol that can be used in hyperlink-hrefs to execute JS code.

XPath condition

This field takes an XPath expression that can be used to address the injection-target more precisely than just by supplying its name and namespace URI. The context the expression is evaluated in is the surrounding XML tag for XML Text injection and the attribute itself for XML Attribute injection.

Example:

```
lower-case (@type) = 'text/javascript'
```

This limits the injection to tags whose *type* attribute contains the value "text/javascript".

Note

It's possible to use the XPath [extension functions](#) that are provided by the [Jaxen](#) XPath engine, e.g. `lower-case()`. Also, there are three additional functions that can be used to determine the current file's name, extension and file type: `file-name()`, `file-ext()` and `file-type()`. You can also use normal code-completion to get a list of available functions.

Warning

For performance reasons, it's recommended to keep these expressions as simple as possible. Especially expressions that cause the whole document to be scanned, such as `//foo/bar` might cause performance problems with large files.

Method parameters

This is a possibility to make use of IntelliJLang's features, if, for any reason, the injection annotations cannot be used. This mainly applies to configuring 3rd party/library methods as well as projects that still have to use Java 1.4.

The language selection is identical to the one described above. To select one or more parameters of a certain method, first choose its containing class either by typing in the name (the textfield supports completion) or by using the class chooser available through the [...] button. Next, select a method using the [...] button inside the "Method-Name" pane (it's not possible to edit the method name manually). Once a method has been chosen, the table in the *Parameters* pane is populated with the parameters of the selected method. Select the check box in the first column to have the selected language injected into arguments passed for this parameter. Note that only parameters of type String can be selected.

Note

There's already an IntelliJ IDEA-core (Inspection Gadgets rather) inspection that checks the arguments of some well-known methods to be a valid regular expression. However, IntelliJLang can provide more detailed error messages and all the features of the Regular Expression Support plugin.

Tip

The XPath language that is configured by default for some of the standard XPath-APIs is provided by the XPathView + XSLT-Support plugin, which is available [here](#).

Importing configuration entries

To import the injection configuration from another PhpStorm installation use the *Import* button from the toolbar and select the file "IntelliLang.xml" from `%IDEA-CONFIG-HOME%/options` or from a JAR file that contains some exported (via File | Export Settings) PhpStorm settings. After selecting the file, a new dialog displays the entries contained in that file. Use the *Delete*-button to remove all entries you don't want to import. Note that this will *not* change the selected configuration file.

This selective import-feature makes it easy to share certain configurations in a team without losing any local entries like when importing settings via the core File | Import Settings action.

Note

To prevent inconsistent data, the import is only possible if the existing configuration is unchanged or has been saved with the *Apply* button.

Advanced settings

The advanced configuration tab allows to specify different names for the base-annotations that should be used. This helps to avoid any dependencies on foreign code where this is not desired or possible. The custom annotations should just provide the same properties as the original ones, i.e. `value` for all of them and an optional (default = "") `prefix` and `suffix` for the `@Language` replacement.

Here it's also possible to configure the kind of runtime checks the plugin should generate for the `@Pattern` validation:

- No instrumentation: No checks will be inserted and doesn't touch any compiled class files
- Instrumentation using `assert`: Pattern-validation is controlled with the `-ea` JVM switch and throws `AssertionError`. This is the recommended way due to the potentially negative impact on the performance for methods that are invoked very often.
- Instrumentation using `IllegalArgumentException`- (for method parameters) and `IllegalStateException` (for return values). This is the same the `@NotNull` instrumentation of PhpStorm does.

Contributed configurations

Additional configurations can be downloaded and imported from here. This community effect can help to minimize the configuration efforts and make IntelliJLang even easier to use.

See Also

External Links:

- [IntelliLang](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

XPath and XSLT Support

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

The XPathView plugin has been created to interactively evaluate XPath Expressions inside PhpStorm.

Note

PhpStorm supports XSLT 2.0 and XPath 2.0 syntax.

Tip

XSLT support is available in all XML files that declare the XSLT-Namespace (<http://www.w3.org/1999/XSL/Transform>) on their root element.

The Expression Evaluation dialog allows to highlight matching expressions in the current editor or to open the [Find](#) tool window to display a list of matching lines. Editing XPath expressions is enhanced by on-the-fly error-checking including a set of customizable [XPath Inspections](#) and a wide range of code completion suggestions.

The plugin both allows to evaluate expressions against the currently focused document, including support for the `document()` function to make cross-document queries and to evaluate an expression against multiple XML documents in a *Find in Path* style way by the new [Find by XPath](#) action.

The plugin also offers a way to display a unique XPath expression for a selected element in an editor. This is available via **View | Unique XPath** in the **Main Menu**

XSLT support is not limited to XPath expressions, it also supports a wide range of XSLT constructs, like checking the existence of templates that are called via `xsl:call-template` and their parameters, refactoring and navigation enhancements. Find out more about this in the [XSLT Support](#) section.

See Also

Reference:

- [XPath Viewer](#)
- [XSLT](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

XPath Expression Evaluation

Previous | [Next](#) | See Also | [Comments](#)

Evaluating XPath expressions in PhpStorm has two main purposes: Testing XPath expressions that are to be used in program code or XSLT scripts, and making structured queries against XML documents. The **Evaluate XPath** action can be invoked either from the editor context menu, the main **Search Menu** or the main toolbar by clicking the action's icon .

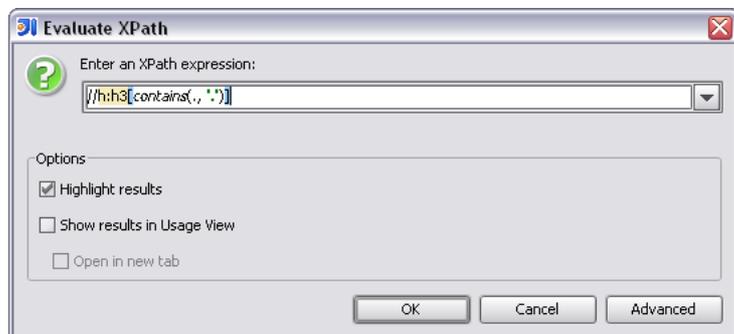
The **Expression Evaluation** has two different modes, a *simple* mode, that allows to enter simple one-line expressions and doesn't allow to configure and Context settings. For more convenient editing of long expressions in a multi-line style way, there's the *advanced* mode dialog that also has a button to edit some context settings, such as namespaces and their prefixes, and variables to use for the evaluation.

Both dialogs feature a history of the recently evaluated expressions, completion, syntax-checking and highlighting, as well as some semantic error checking of the entered expression. Semantic check include validation of used namespace prefixes, useless XPath expressions (e.g. `@comment()`) and node tests for element/attribute names that don't occur in the context document and would not be successfully matched.

Some error checks and XPath inspections also provide Quick Fixes for detected problems, e.g. the possibility to map an unresolved namespace-prefix to a URI by intention.

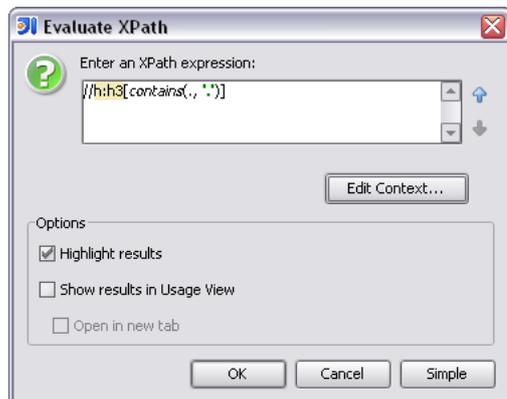
Simple mode

The simple mode comes with an input field that can be used to enter simple one-line expressions that don't require any customization of namespace prefixes or make use of predefined variables. The last recently used expressions can be selected from the drop-down list.



Advanced mode

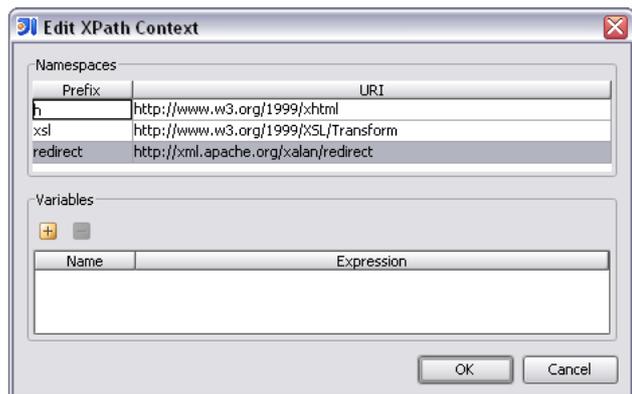
The advanced mode adds the possibility to edit expressions in a multiline editor and has another button to edit the [XPath Context](#). The expression history can be browsed by the Up- and Down arrays on the right of the dialog or the keyboard shortcuts for the previous/next history element (usually **Ctrl-Alt-Up**, **Ctrl-Alt-Down**).



XPath context

This dialog allows to assign custom prefixes to the namespace URIs that are used in the context document. This can be useful to assign a shorter prefix, resolve prefix clashes or to actually define a prefix for the default namespace. This can be essential because XPath does not automatically match elements in the default namespace without specifying a prefix for the element to be matched.

The XPath Context also includes the possibility to define custom XPath variables that can be used in queries for repeating expressions. Each variable in the table can be assigned an expression that will be evaluated once when the query is executed. The resulting value is then available for multiple use at no additional computational cost.



Options

- [Highlight results](#)
- [Show results in Usage View](#)

Highlight results

This highlights the matched nodes in the current editor. Matched nodes that don't belong to the current editor (may happen by using the `document()` function) are not highlighted. It's recommended to display such cross-document results in the **Find Usages** toolwindow.

Show results in usage view

Shows all matched nodes in the **Find Usages** toolwindow. Check **Open in new tab** to open the result in a new tab instead of reusing the last one.

See Also

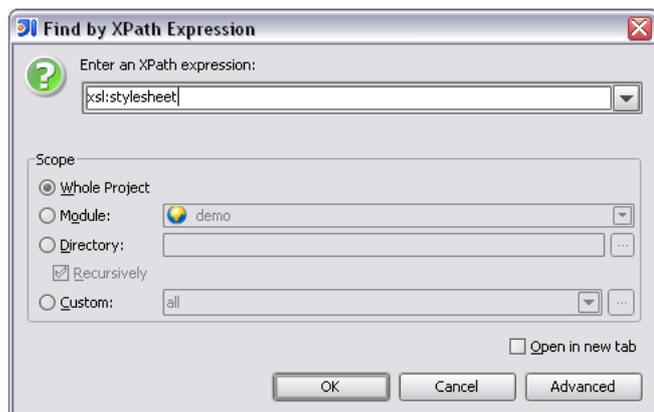
Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

XPath Search

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

This new feature is the XML-aware counterpart to PhpStorm built-in functions *Find in Path* and *Search Structurally*: It allows to find occurrences of certain XPath expressions in all XML files in a specific scope. It is available as **Find by XPath...** under the **Search** menu.



The scope that can be chosen is either the full project scope, a specific module or a simple directory. There's also the possibility to use *custom* scopes, but this is somewhat limited as it will only scan XML files inside source-folders.

This dialog provides the same functionality regarding completion, error checking and intentions as described in [Evaluate Expression](#).

See Also

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

XPath Expression Generation

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

This action computes a unique XPath expression that matches the currently selected node in the document. The action is available from the View-Menu (Unique Path) and the Editor-Context Menu (Show Unique XPath). The action is only enabled when the caret is placed on an element that a useful expression can be generated for.



Tip
The generated expression in the pop-up can be selected and copied into the clipboard for further use.

If a simple XPath expression like /root/something/else doesn't produce a unique result, the action has two strategies to make it unique:

- If the non-unique node is an element, the action looks for attributes with the name 'id', 'name' and attributes that are of ID-type, as defined by the document's DTD or XML Schema.
Example: /root/something[@id="foo"]/else
- For nodes other than elements (comments, processing instructions), or if the above rule doesn't produce a unique result, the index of the node inside its parent is appended.
Example: /root/something/else[2]

See Also

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Plugin Settings

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

XPathView plugin settings are configured in the [Settings](#) dialog box.

Item	Description
Scroll first hit into visible area	Select this check box to have the editor automatically scroll to the first XPath match.
Use node at cursor as context node	Select this check box to have the entered XPath expression use the currently selected node (tag/attribute/pi, etc.) as its context node and evaluate the expression relatively to this node.
Highlight only start tag instead of whole tag content	Do one of the following: <ul style="list-style-type: none"> • Select this check box to have only the name of a matching tag highlighted. • Clear this check box to have the entire content of a matching tag highlighted.
Add error stripe markers for each result	Select this check box to have each match supplied with an error stripe marker which can be quickly navigated to. The tooltip of each marker shows the matched content.
Show actions in Toolbar	Select this check box to have buttons that invoke XPath-related actions displayed on the Main Toolbar .
Show actions in Main Menu	When this check box is selected, XPath-related actions are available from the main menu.
Colors	In this area, configure color indication during execution of XPath expressions. <ul style="list-style-type: none"> • Highlight Color - in this area, select the color to indicate XPath matches in the editor. • Context Node Color - in this area, select the color to indicate the current context node.

See Also

Reference:

- [XSLT](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

3.0.0.+

XSLT Support

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

- [Completion](#)
- [Error Highlighting](#)
- [Refactoring](#)
- [File Associations](#)

See Also

Reference:

- [XSLT](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Completion

Previous | [Next](#) | See Also | [Comments](#)

With Code Completion being one of the key features of PhpStorm, the plugin provides several possibilities to complete keywords, predefined functions, variables and parameters used in XPath expressions, template names and names of parameters that can be passed to a template invocation.

Completion in XPath expressions

It's possible to complete all parameters/variables in scope inside an XPath expression in a normal expression attribute or inside an [attribute value template](#)

```
<xsl:message>
<xsl:value-of select="concat('Param: ', $my-)" />
</xsl:message>
```

Also, all predefined functions and keywords are available for completion, including function signatures.

```
<xsl:value-of select="st" />
```

Note

The [Quick Documentation Lookup](#) also works in completion lookup lists.

Completion for template names in xsl:call-template

The template's name that is to be called can be completed from a list of all named templates in the current document and included stylesheets

```
<xsl:call-template name="my-template">
<xsl:with-param name="my-template" />
<xsl:with-param name="my-template2" />
</xsl:call-template>
```

Completion for template parameters

There's a special completion for parameters that should be passed to a template in a xsl:call-template invocation. The completion lists all parameters that are declared by the template and are not yet present in the *argument list* of the invocation, i.e. there's no xsl:with-param yet.

```
<xsl:call-template name="my-template">
<xsl:with-param name="my-param1" select="123" />
<xsl:with-param name="my-" />
<xsl:with-param name="my-param2" select="" />
</xsl:call-template>
```

Tip

That's especially useful when also using completion to create the xsl:with-param tag, because this will automatically trigger the completion for the parameter's name which is required according to the schema; e.g.

```
<xsl:with-p <ctrl-space>
=>
<xsl:with-param name="[lookup list]"
```

See Also

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

Error Highlighting

Previous | [Next](#) | See Also | [Comments](#)

The XSLT support can detect a range of errors in XSLT constructs, such as misspelled template names, missing template parameters, bad match-patterns, references to undeclared variables, wrong or useless embedded XPath expressions, etc. and also offers Quick Fixes to automatically fix some of those errors.

- [Syntax Highlighting](#)
- [XPath Syntax checks](#)
- [XPath Type Checking](#)
- [Pattern validation](#)
- [Unresolved References](#)
- [Duplicate declarations](#)
- [Other checks](#)

Syntax highlighting

Where allowed, XPath function-calls, Axis names, Numbers, Strings, etc. are highlighted according to the currently active color scheme. By default, the plugin uses the colors that are defined for the corresponding Java types, such as number- and string literals, etc. If a different coloring is desired, those colors can be configured in IDE-Settings | Colors & Fonts on the tab XPath.

XPath syntax checks

Just like the interactive XPath Expression Evaluation, the XSLT support catches any syntax error in XPath expressions used inside a stylesheet.

```
<xsl:value-of select="foo/bar/" />
location step expected
```

XPath type checking

In XPath, almost all types are assignable to each other with certain well-defined conversion semantics. However, the conversion to NODESET isn't defined for any type and there's also no (portable) conversion function available. Such type-conversions are highlighted as an error.

```
<xsl:value-of select="name(12345)" />
Expected type 'nodeset', got 'number'
```

Pattern validation

A special form of XPath expression are patterns in XSLT. They are e.g. used as the value of the match-attribute in xsl:template elements. Only a certain subset of XPath expressions is allowed here, which the XSLT support checks for.

```
<!-- Pattern validation -->
<xsl:template match="string()" />
Bad pattern
```

Unresolved references

References to variables that have not been declared or are not accessible from the current scope are detected and highlighted as an error. There are Quick Fixes available to create a variable or parameter declaration for such unresolved variables references.

```
<!-- undefined variable check -->
<xsl:message>
  <xsl:value-of select="concat('Variable: ', $my-variable)" />
</xsl:message>
Unresolved variable 'my-variable'
```

Quick-Fixes:

```
<xsl:message>
  <xsl:value-of select="concat('Variable: ', $my-variable)" />
  Create Variable 'my-variable'
  Create Parameter 'my-variable'
```

Duplicate declarations

In XSLT there must not be more than one variable or parameter declared on the same scope level. It's also not allowed that there is more than one template with the same name. The plugin will identify such duplicate declarations and highlight them in the editor.

Other checks

- [Shadowed variables](#)
- [Missing Template Arguments](#)
- [Superfluous Template Arguments](#)
- [Function Call Arguments](#)
- [XPath Inspections](#)

Shadowed variables

Even though it is allowed to have identically named variables or parameters in different nesting levels, this can be confusing and is likely to cause programming mistakes. The plugin can identify shadowed declarations and offers Quick-Fixes to either rename the local or the outer variable.

Missing template arguments

Another check that the XSLT support performs is whether all required parameters are specified with a xsl:call-template. A parameter is considered required if there is no default value, i.e. if there's no select attribute and the parameter's declaring element has an empty body.

```
<xsl:call-template name="my-template">
Missing template parameter: my-param2
```

Quick-Fixes:

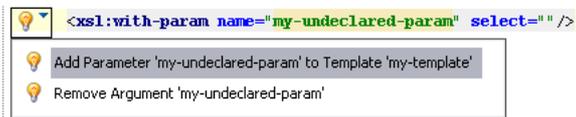
```
<xsl:call-template name="my-template">
Add Argument for 'my-param2'
```

Superfluous template arguments

There's also a supplemental check that flags arguments that are not declared as template parameters. There are Quick Fixes available to either remove the argument from the template-call or to add a corresponding parameter to the called template.

```
<xsl:with-param name="my-undeclared-param" select="" />
:/xsl:call-template>
Superfluous template parameter: my-undeclared-param
```

Quick-Fixes:



Function call arguments

The XSLT support does, just like the interactive XPath Expression Evaluation, check whether the number and types of function arguments match their declaration for built-in functions of XPath and XSLT.

```
<!-- missing function parameter check -->
<xsl:message select="concat($my-param2)" />
```

Function 'concat' requires 2 arguments

XPath inspections

All [XPath Inspections](#) are supported for editing XSLT documents. Those inspections can also be suppressed in a way that is similar to the standard suppression-mechanism PhpStorm uses for Java code by using *noinspection* XML comments. The suppression is possible on different levels, either on instruction level, template-level (if applicable) or stylesheet level.

See Also

Web Resources:

- <http://www.jetbrains.com/idea/2012/03/2012-03-02-what-is-new-in-idea-12-0-3/>
- <http://youtrack.jetbrains.com/issues/WI>

XPath Inspections

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

The XPath language implementation provides a few built-in inspections that can check for common coding mistakes when writing XPath expressions both in the interactive mode and when writing XSLT scripts. Those inspection also provide a number of configuration options which can be configured on the [Inspections](#) page of the [Settings](#) dialog box.

Note

Due to the way how these inspections are implemented and integrated, they only work for the on-the-fly editor highlighting and *not* if the inspections are run via **Code | Inspect Code**.

XPath type checking

There are two inspections that deal with type-conversion in XPath expressions: **Implicit Type Conversion** and **Redundant Type Conversion**

Implicit type conversion

This inspection checks for any implicit conversions between the predefined XPath-types STRING, NUMBER, BOOLEAN and NODESET. While this is usually not a problem as the conversions are well-defined by the standard, this inspection can help to write XSLT scripts that are more expressive about types and can even help to avoid subtle bugs:

```
<xsl:if test="foo" /> is not the same as <xsl:if test="string(foo)" />
```

The first test checks whether the element `foo` exists (`count(foo) > 0`), the latter one however is only true if the element actually contains any text (`string-length(foo) > 0`). The plugin will then offer to make the type-conversion more explicit.

There are several options to adjust the inspection to personal preferences by offering the possibility to individually enable it for implicit conversions between certain types.

The plugin can also be told to always flag explicit conversions that do not result in the actually expected type, such as `<xsl:if test="number(foo)" />` and provides a special option to ignore the conversion from NODESET to BOOLEAN by using the `string()` function as a shortcut for writing `string-length() > 0`.

Redundant type conversion

This inspection checks whether any type-conversion with the functions `string()`, `number()` or `boolean()` is redundant, i.e. whether the type of argument is the same as the functions return type or if the expected type of the expression is of type *any*. While such an explicit conversion may sometimes be intentional to emphasize the type, this can usually be safely removed.

Expression validity checks

Those inspections check whether an expressions contains any potential semantic mistakes, such as referencing element/attribute names that don't occur in instance documents or using predicates that don't potentially match anything.

Check node test

This inspection checks whether any element/attribute names that are used in XPath-expressions are actually part of an associated XML file or are defined in a referenced schema. This helps to avoid problems caused by typos in XPath-expressions that would otherwise occur when running the script and may even then not be recognized immediately.

Example:

```
<xsl:template match="h:textarea" />
```

If the prefix `h` is bound to the XHTML namespace, the inspection will flag this part of the match-expression as an unknown element name because the correct name of the element is `textarea`.

Index zero usage

This inspection checks for any accidental use of zero in a predicate index or in a comparison with the function `position()`. Such is almost always a bug because in XPath, the index starts at one, *not* at zero.

Example:

```
//someelement[position() = 0] or //something[0]
```

Developing custom XPath inspections

The XPath inspections make use of the normal inspections API of PhpStorm. However, due to the way the XPath Language-Support is integrated, this is a bit more complicated and it's at the moment not readily possible to develop full-blown 3rd-party XPath inspections. While it's theoretically possible to develop custom inspections that make use of the XPath-PSI API and are derived from `org.intellij.lang.xpath.validation.inspections.XPathInspection`, this is not recommended and not supported.

Please contact me if there's a need for a special inspection or there is significant interest that would justify the effort to make this more pluggable.

See Also

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

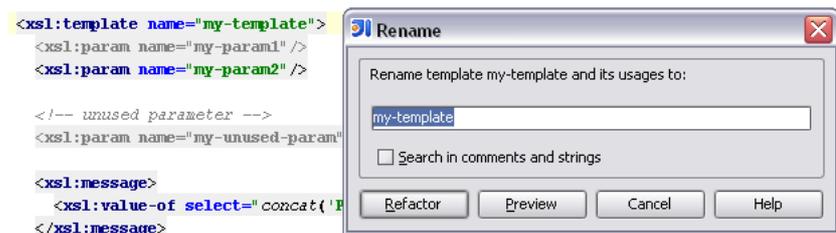
Refactoring

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

The plugin supports some of PhpStorm built-in refactorings **Rename** and **Safe Delete** for XSLT items such as templates, variables and parameters. It's also possible to create XSLT variables from selected XPath expressions with the well-known **Introduce Variable** shortcut. Inlining variables is possible with the **Inline** shortcut as well.

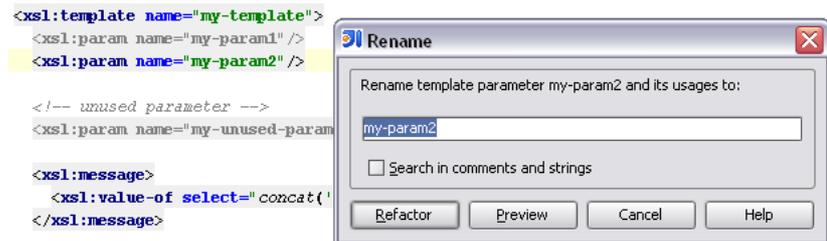
Renaming templates

Named templates can be renamed in PhpStorm just like any other symbol. All `xsl:call-template` invocations that refer to this template will be updated accordingly.



Rename variables and parameters

Just as named templates, it is possible to rename XSLT variables and template parameters, either at a point of their use or at their declaration.



Safe delete

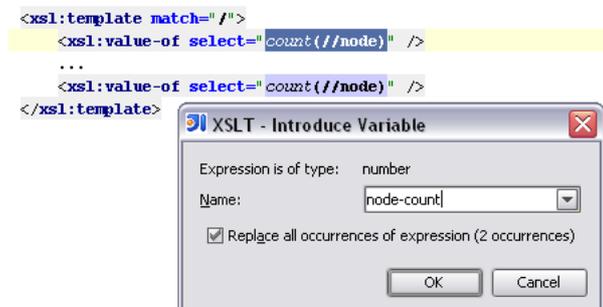
Named templates, parameters and variables can be deleted using PhpStorm Safe Delete feature, i.e. the item will be removed if there aren't any references left to it in other stylesheets across the project.

Note

This is especially useful if the stylesheet may be included in other ones via `xsl:include` or `xsl:import` to make sure that nothing will be deleted that is still used somewhere else.

Introduce variable

It is possible to extract XPath-Expressions and turn them into an `xsl:variable` declaration. Check the **Replace all occurrences** check box to replace all other occurrences of the same expression.



Introduce parameter

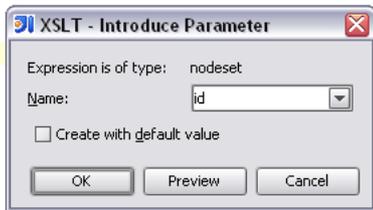
This is similar to **Introduce Variable**, but it creates a new parameter instead of a variable. It also has an additional option **Create with default value** that determines if the selected expression should be added as the introduced parameter's default value or if all calls to the template should be updated to pass the selected expression. That option is only available when introducing parameter to named templates.

```

<xsl:template name="test">
  <xsl:value-of select="foo/@id" />
</xsl:template>

<xsl:template match="/">
  <xsl:call-template name="test" />
</xsl:template>

```



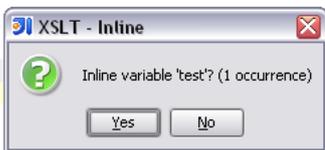
Inline variable

This is just the opposite of **Introduce Variable**, it replaces all usages of a variable with the expression that is specified in the variable's `select`-attribute. Variables that don't have such an attribute cannot be inlined. Inlining variable references that resolve to parameters is not possible as well.

```

<xsl:variable name="test" select="foo/@id" />
<xsl:value-of select="$test" />

```



See Also

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

File Associations

[Previous](#) | [Next](#) | [See Also](#) | [Comments](#)

File Associations are used to associate an XSLT file with other XML files. This is currently used for three purposes:

- Enhanced completion for element- and attribute names in XSLT node-selections. The completion will offer all element- and tag names that are found in the associated documents.
- Enhanced error highlighting for XSLT node-selections. If an XSLT script has been associated with one or more XML files, any references to element- and attribute names that are not part of the associated files will be flagged with the warning message: "Tag name '...' is not part of the document".
- File Associations are also used for creating **Run Configurations**. The XML input file to use for the transformation can be conveniently chosen from the list of associated files.

Managing file associations

XSLT File Associations are defined per project and managed in the [Settings - XSLT File Associations](#) dialog box.

- [Managing Associations from the editor](#)
- [Removing an Association](#)

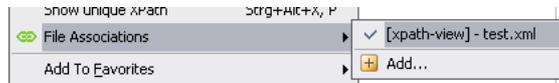
Managing associations from the editor

Associations can be also created by invoking the **Add...** action from the **File Associations** group in the Editor Context Menu. A file-selection dialog opens that can be used to select one or more X(HT)ML files to associate the XSLT file with.



Removing an association

An Association can be removed by clicking on the corresponding file name in the **File Associations** group. The file names are displayed with a path that is relative from the current file. If the associated file is part of any module, the module name is included in square brackets.



It's also possible to invoke the associations configuration dialog through the **Configure...** action. This opens the associations configuration dialog and preselects the file that's currently opened in the editor.

See Also

Reference:

- [XSLT File Associations](#)

Web Resources:

- <http://www.jetbrains.net/devnet/community/wi>
- <http://youtrack.jetbrains.net/issues/WI>

