



PhpStorm Workshop

Maarten Balliauw
Technical Evangelist

maarten.balliauw@jetbrains.com

This workshop presentation is property of JetBrains and can
only be used commercially with prior permission.



Before we start

Make sure you have PhpStorm 7.1 installed
either full or free trial from <http://jetbrains.com/phpstorm>

Make sure you have a PHP runtime, webserver, MySQL, for example
XAMPP (Windows) - <http://bit.ly/phpstorm-xampp>
MAMP (Mac OS X) - <http://bit.ly/phpstorm-mamp>

Make sure you have git command line

Who am I?

Maarten Balliauw

Belgium (Antwerp)

Technical Evangelist, JetBrains

Focus on web

ASP.NET MVC, Windows Azure, PHP, API's, ...

Big passion: cloud (Windows Azure)

<http://blog.maartenballiauw.be>

[@maartenballiauw](https://twitter.com/maartenballiauw)



How it all works

This course covers PhpStorm 7.1

But most is applicable to earlier versions

There will be some theory

And some practice

We will not cover every knob and bolt

I talk a lot, shut me up

Do ask questions!

Keyboard shortcuts

PhpStorm features multiple keymaps

Defaults per platform   

Other IDEA-based IDE's use a similar keymap

Cheat sheet: <http://bit.ly/phpstorm-shortcuts>

Agenda

Look at the IDE

Navigation

Editing

Inspections

Live Templates

Refactoring

Debugging

Todo Explorer

Unit Testing

Version Control

Databases

Deployment

Tools

Plugins



Look at the IDE

Newly installed IDE

Welcome screen

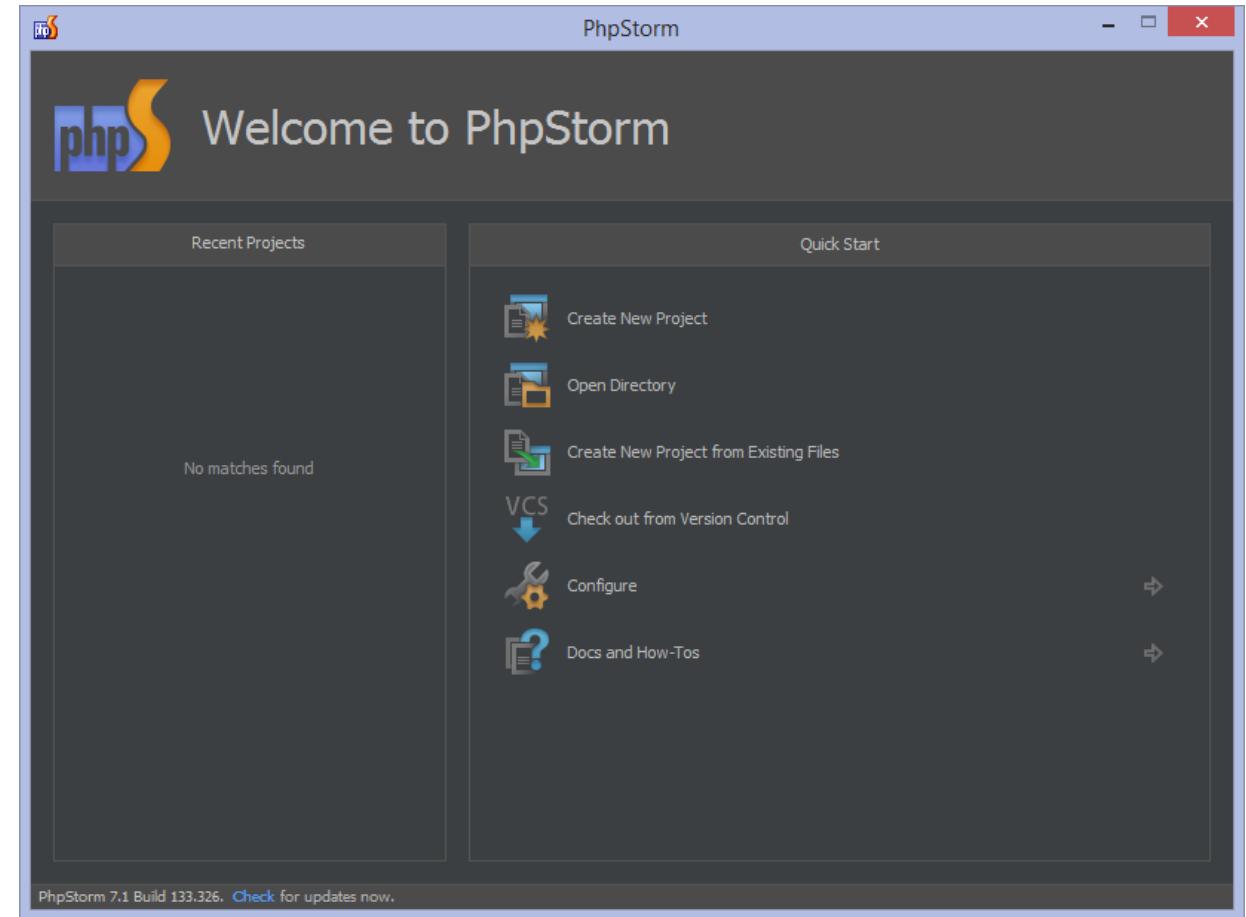
Create project

Open project

Checkout from version control

Configuration

Project? The basis for coding assistance,
bulk refactoring, coding style consistency,
etc.



Opened project

Menus and toolbars

Execute various commands

Navigation bar

Navigate through project

Status bar

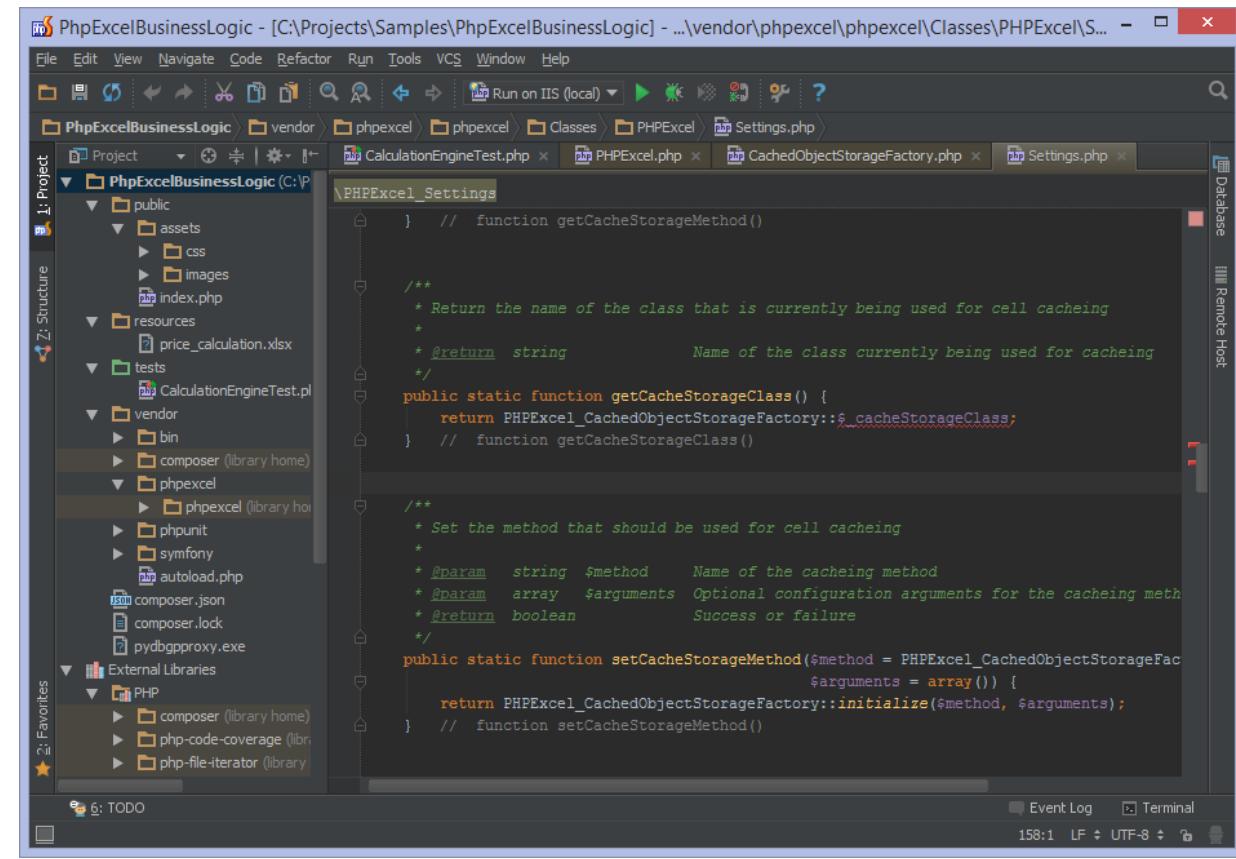
Information about IDE and project

Editor

Where coding happens

Tool windows

Numerous helpers (e.g. database, TODO, ...)



Configuring PHP

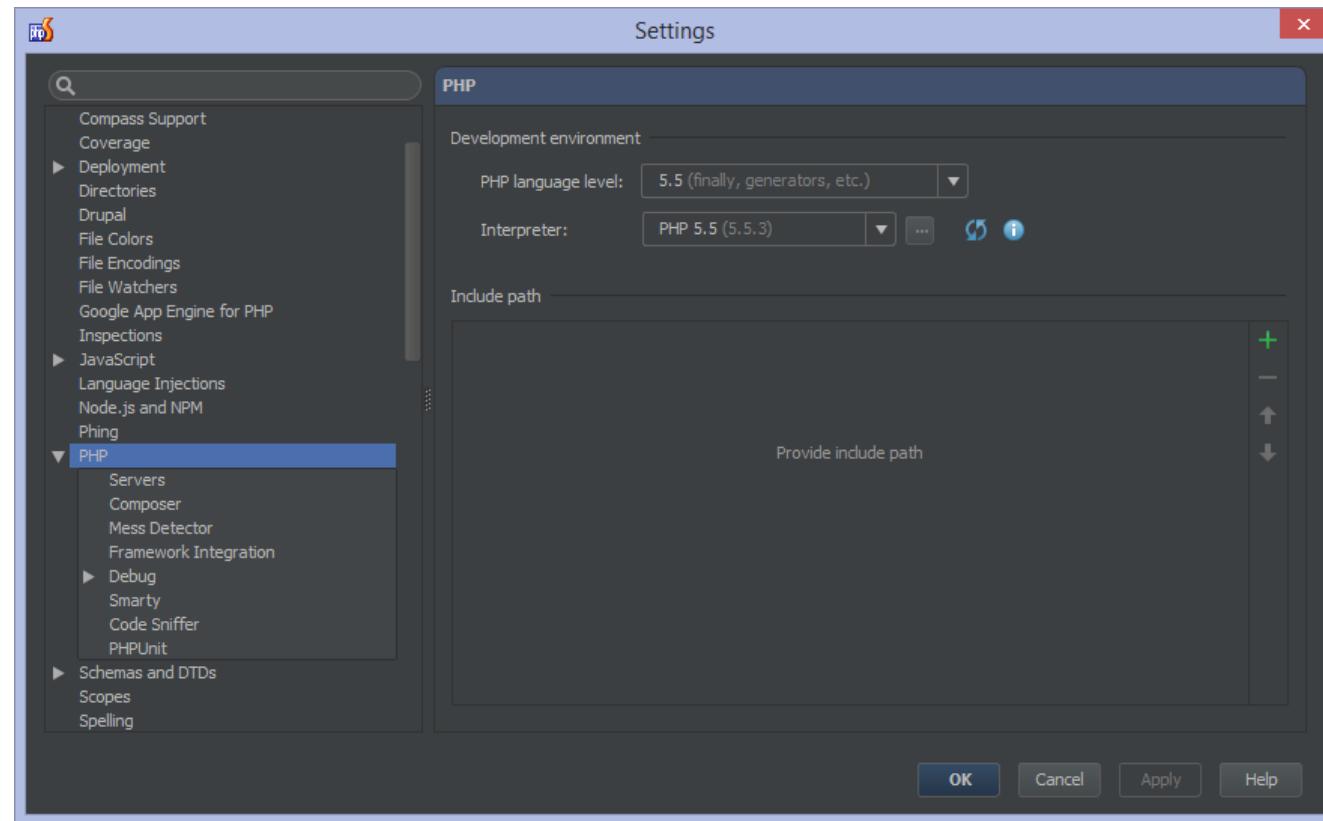
Under Project Settings | PHP,
set PHP options.

Language level
Interpreter
Include paths
Debugging

...

  Ctrl+Alt+S

 Command+,



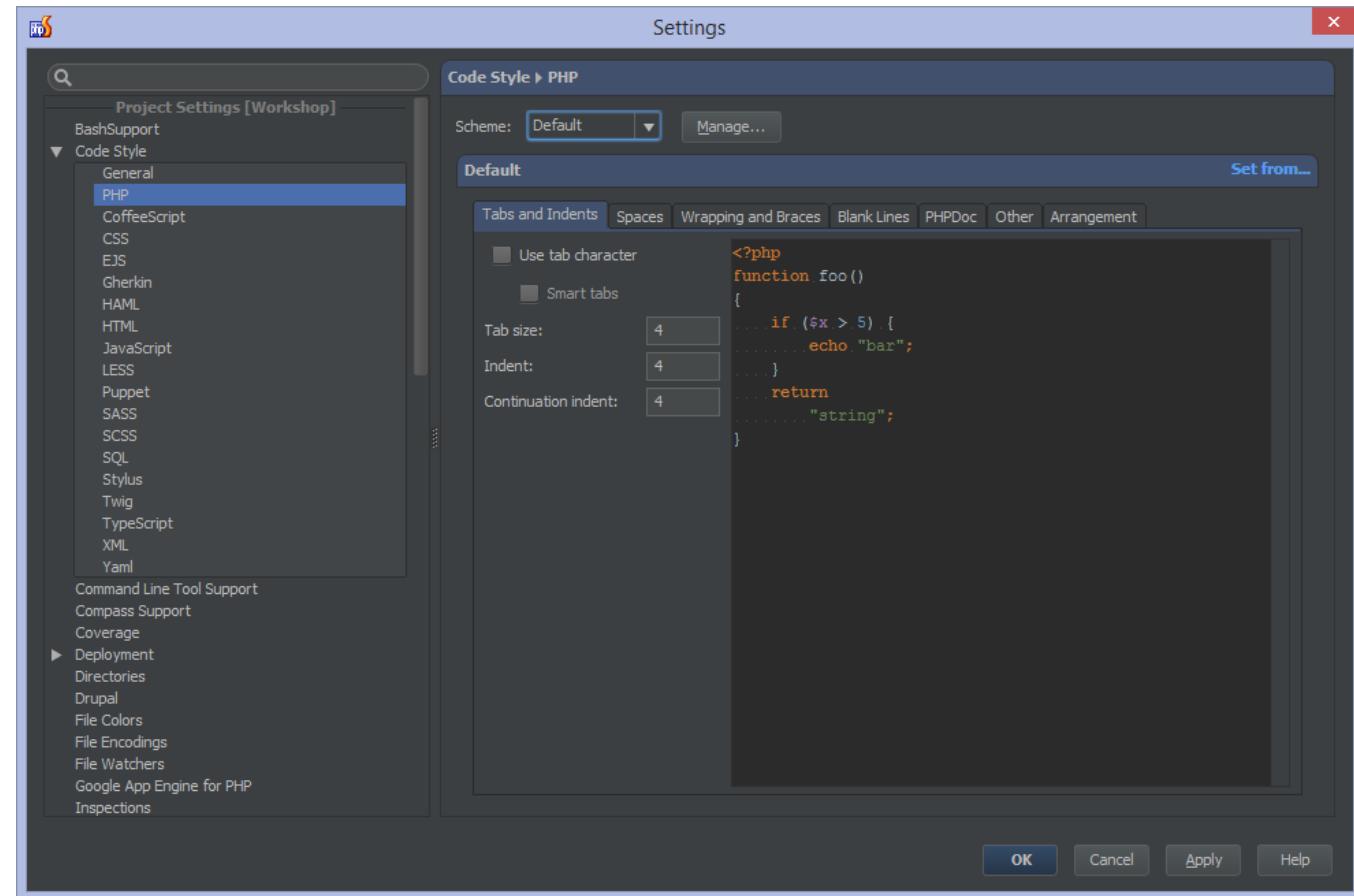
Configuring PhpStorm

Project settings
Specific for current project

IDE settings
Global for all projects

Windows icon Linux icon Ctrl+Alt+S

Mac icon Command+,



Tabs

Open a document each
Navigate between tabs

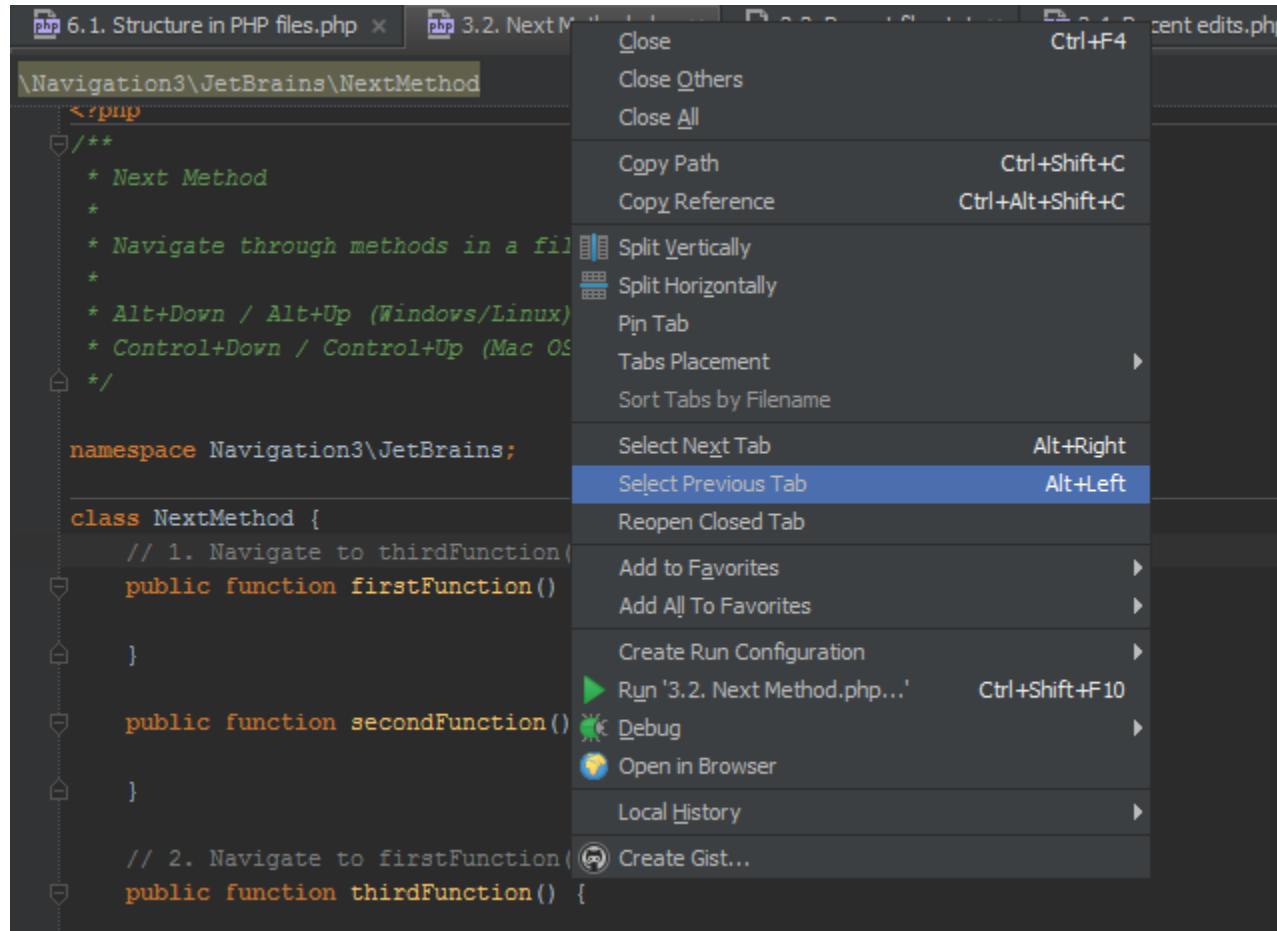
Pin tab

Split editor

Tabs can be favorited

  Alt+Left / Right

 Command+Alt+Left / Right



Let's fetch the workshop materials

Start PhpStorm and if a project is opened, close it

Create a new project, name it “Workshop” and select the “Composer project” type

Find the “jetbrains/phpstorm-workshop” package and click OK

Alternatively, use “Checkout from Version Control”

Repository: <https://github.com/maartenba-demo/phpstorm-workshop.git>

Project / directory name: Workshop

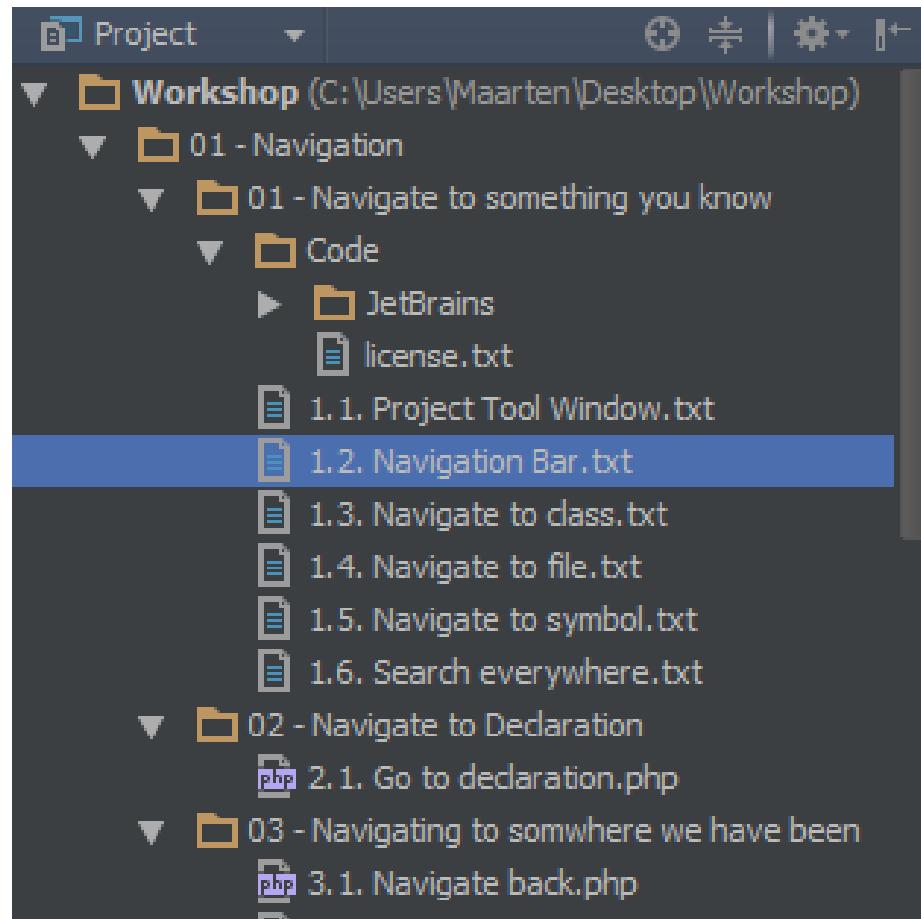
Navigation

Project Tool Window

Shows all files in project.

Windows icon Alt+1

Mac icon Command+1



Navigation Bar

Navigate through files in project.

Left/Right navigates path.

Up/Down opens other node in path.



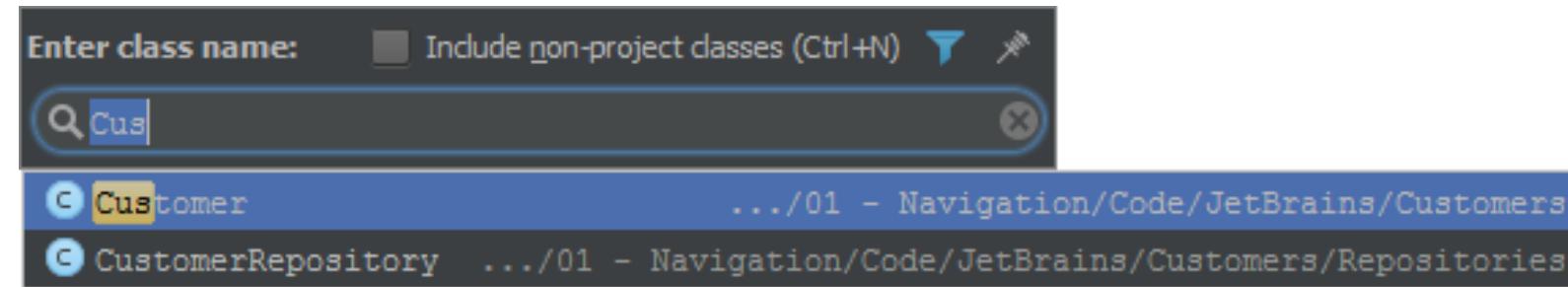
Windows icon Alt+Home

Mac OS X icon Alt+Home

Navigate to Class

Navigates to a given class
can be in PHP or any other language
supported by PhpStorm

use CamelHumps / wildcards



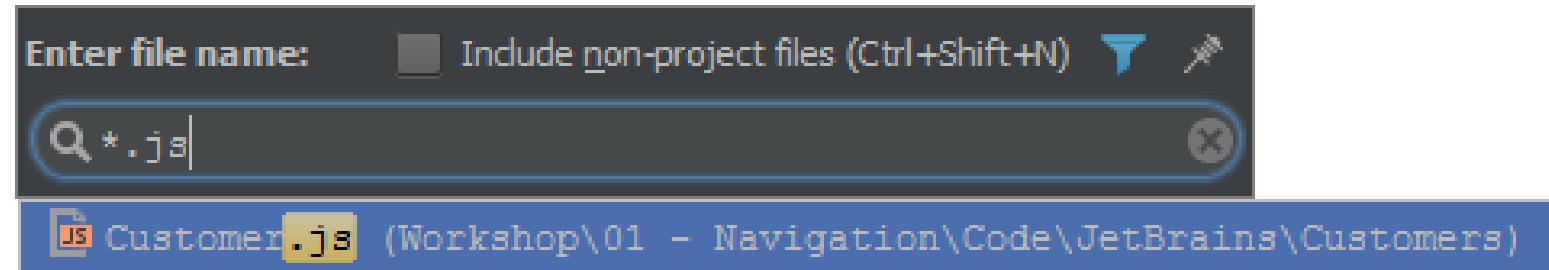
Windows icon Ctrl+N

Mac icon Command+0

Navigate to File

Navigates to a given file
any file type

use CamelHumps / wildcards



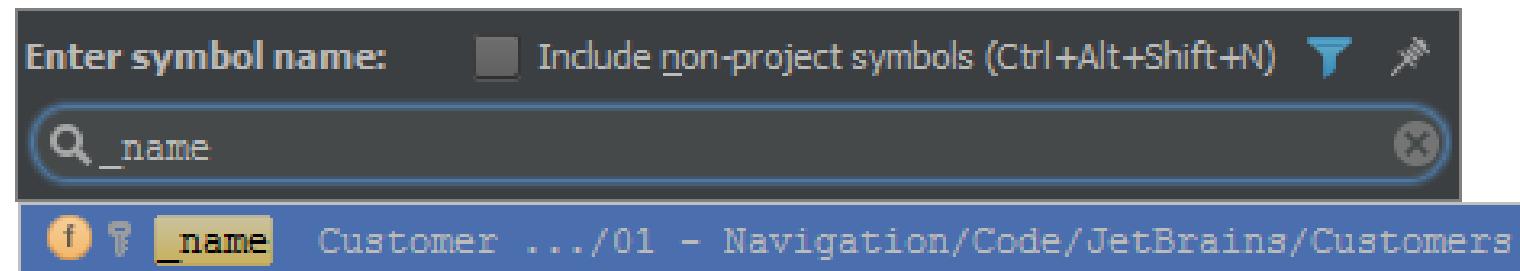
Windows icon Ctrl+Shift+N

Mac icon Command+Shift+0

Navigate to Symbol

Navigates to a given symbol
can be in PHP or any other language
supported by PhpStorm

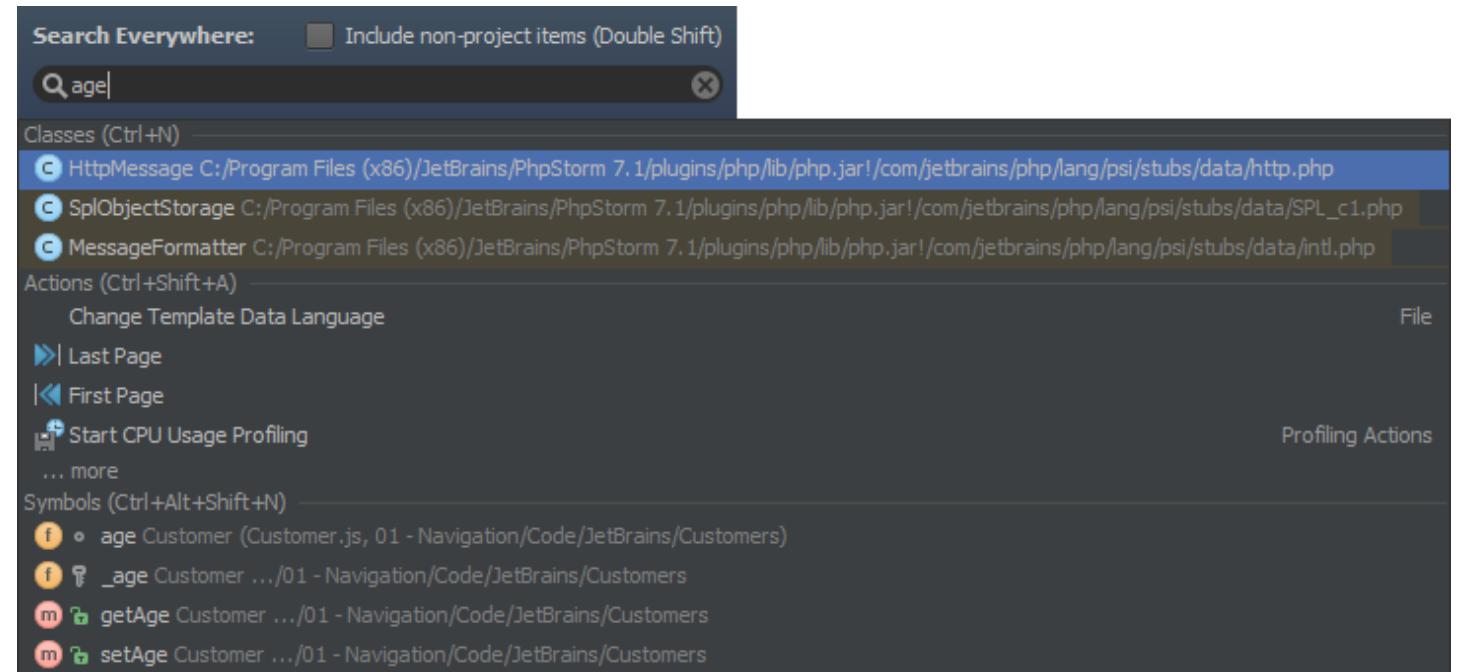
use CamelHumps / wildcards



Windows: Ctrl+Shift+Alt+N
Mac OS X: Command+Shift+0

Search Everywhere

Search classes, file, symbols and
menu actions all at once
use CamelHumps / wildcards



Navigate to Declaration

Ctrl-click for hyperlink
or use shortcut

```
// 1. Ctrl+Click on the Customer symbol
function Customer:: construct($name, $age) : Customer
// 3. Go to declaration for the $age variable passed
$person = new Customer($name, $age);
```

```
    /**
     * @return int
     */
    public class Customer implements Comparable
    {
        return $this->_age;
    }
```

  Ctrl+B

 Command+B

Navigate Back / Forward

Use the keyboard shortcut



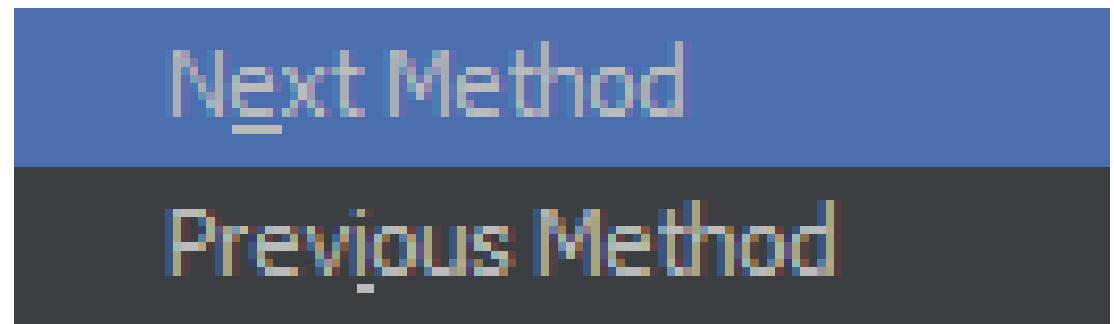
Windows icon Linux icon Ctrl+Alt+Left / Right

Mac icon Command+Alt+Left / Right

Next / Previous Method

Use the keyboard shortcut to navigate through methods in a file/class.

Navigates between tags in HTML. Also works in other file types.



Alt+Down / Alt+Up

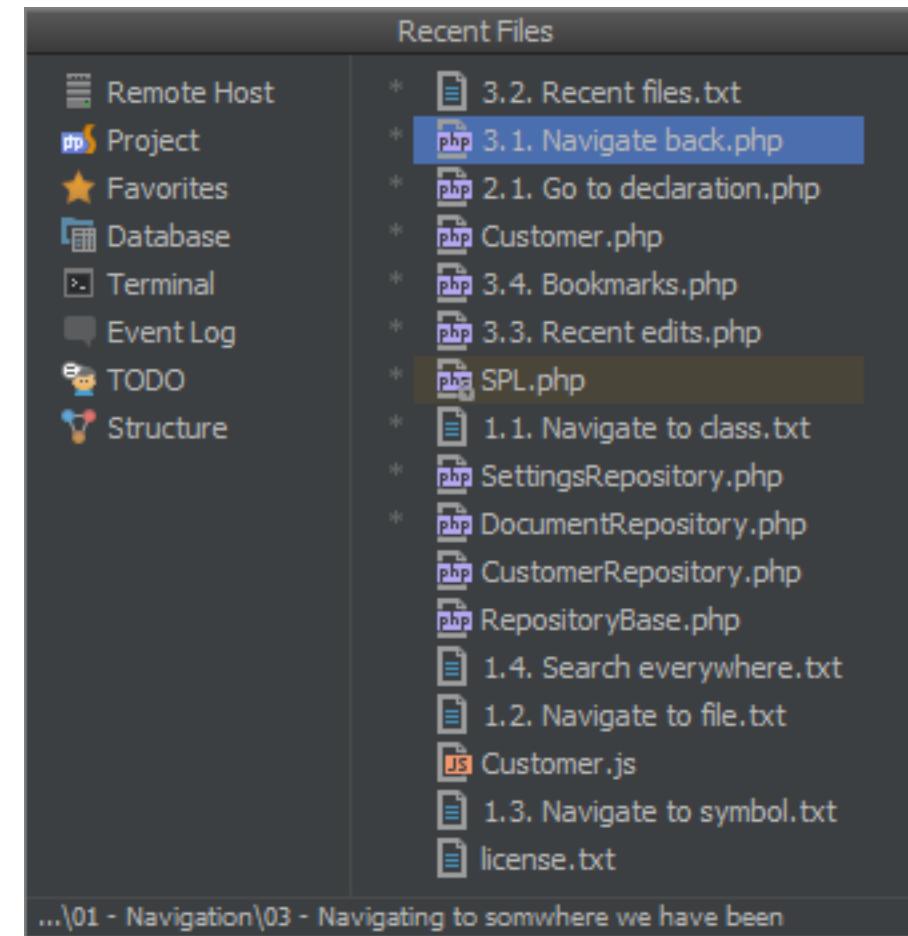
Control+Down / Control+Up

Recent Files

List of recent files
use CamelHumps / wildcards

Windows icon Linux icon Ctrl+E

Mac icon Command+E



Navigate to Last Edit Location

Use the keyboard shortcut to navigate back to the last edit location.

Last Edit Location

  Ctrl+Shift+Backspace

 Command+Shift+Backspace

Bookmarks

Navigate between “bookmarked” locations.



Penguin icon F11 Toggle Bookmark

Ctrl+F11 Toggle Numbered Bookmark

Shift+F11 Show bookmarks

Ctrl+0..9 Navigate to numbered bookmark



F3 Toggle Bookmark

Alt+F3 Toggle Numbered Bookmark

Shift+F3 Show bookmarks

Ctrl+0..9 Navigate to numbered bookmark

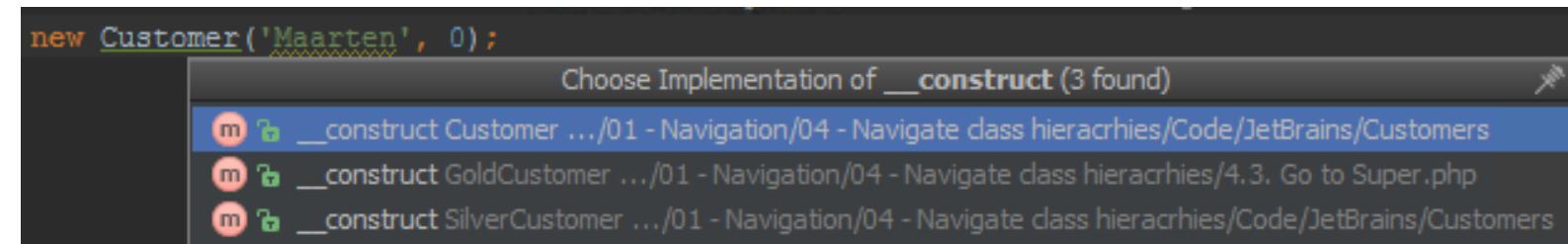
```
3.4. Bookmarks.php:37
3.4. Bookmarks.php:33
3.4. Bookmarks.php:29

23 * 5. Navigate to the numbered bookmark (Ctrl+3)
24 * 6. Remove the bookmark from secondFunction()
25 * 7. From the bookmarks list, add a description to one of
26 */
27
28 class Bookmarks {
29     public function firstFunction() {
30
31 }
32
33     public function secondFunction() {
34
35 }
36
37     public function thirdFunction() {
38
39 }
40 }
```

C:\Users\Maarten\Desktop\Workshop\01 - Navigation\03 - Navigating to somewhere we have been\3.4. Bookmarks.php

Go to Implementation

Navigates to the implementation of a given class.

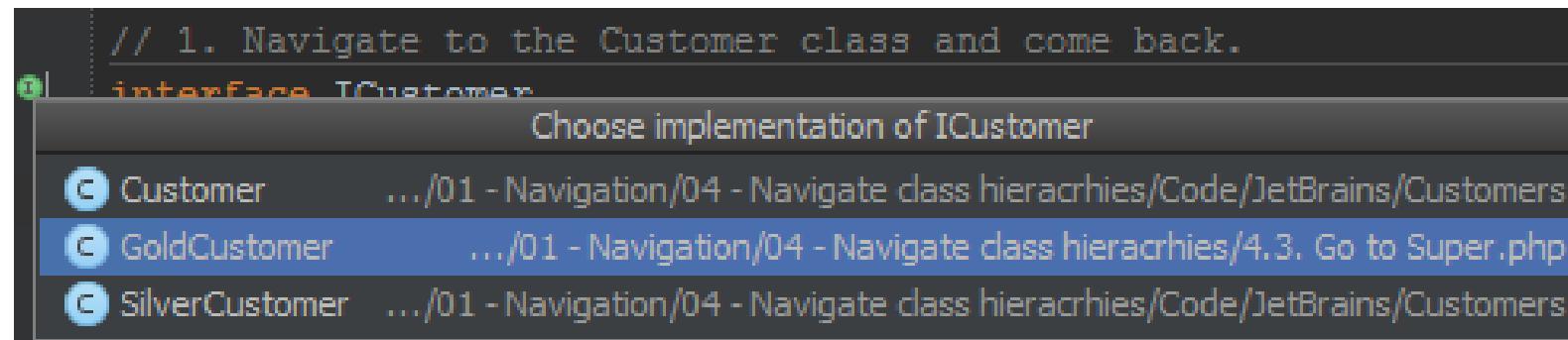


Windows icon Ctrl+Alt+B

Mac icon Command+Alt+B

Go to Derived

Navigates to the implementation of a given interface or to a subclass.

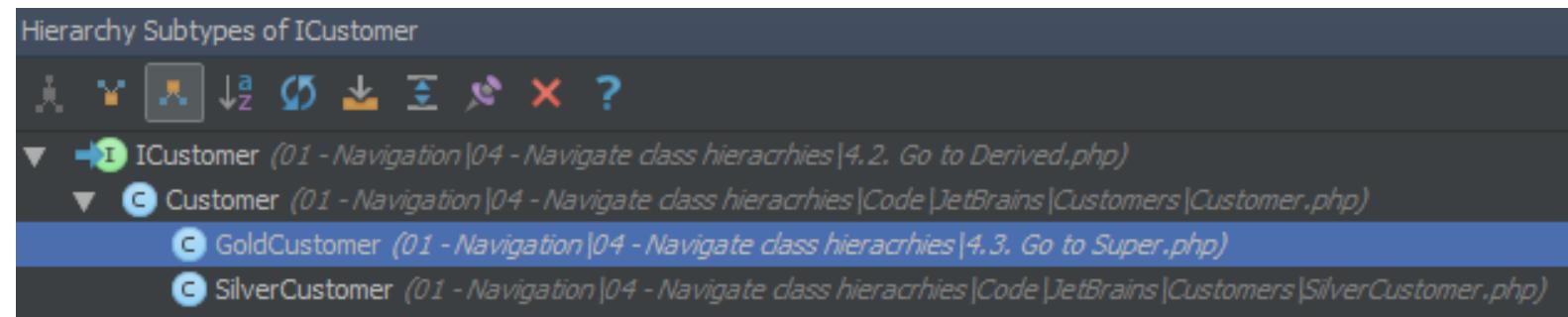


Windows icon Use the left gutter

Linux icon Use the left gutter

Go to Super class/method

Navigates to the super class or method.



- Ctrl+U (Ctrl+H for hierarchy)
- Command+U (Ctrl+H for hierarchy)

Highlight Usages in File

Highlights usages of a symbol in the current file. Different colors for read/write.

Esc to clear highlighting.

  Ctrl+Shift+F7
 Command+Shift+F7

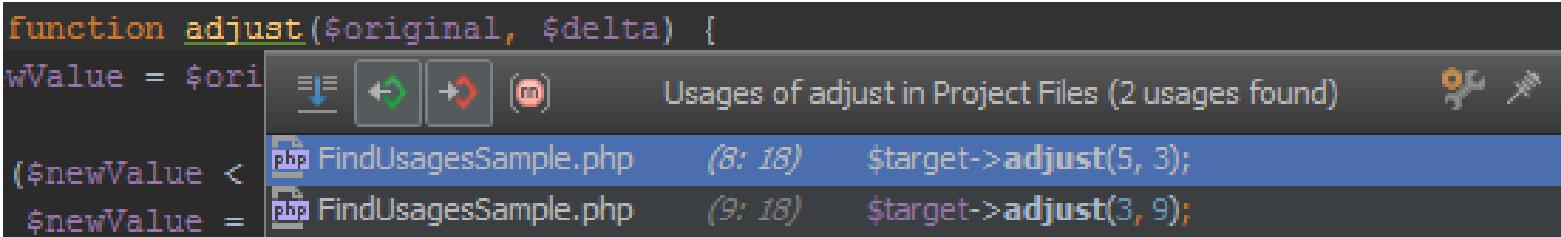
```
public function adjust($original, $delta) {
    $newValue = $original + $delta;

    if ($newValue < self::MIN_VALUE) {
        $newValue = self::MAX_VALUE;
    }
    if ($newValue > self::MAX_VALUE) {
        $newValue = self::MIN_VALUE;
    }

    return $newValue;
}
```

Find Usages

Find usages of a symbol in the current project. Use tool window or popup.



```
function adjust($original, $delta) {
    $newValue = $ori
    if ($newValue <
        $newValue =
```

Usages of **adjust** in Project Files (2 usages found)

File	Line	Usage
FindUsagesSample.php	8: 18	\$target-> adjust (5, 3);
FindUsagesSample.php	9: 18	\$target-> adjust (3, 9);

Windows: Alt+F7 (Ctrl+Alt+F7 for popup)

Mac OS X: ⌘ F7 (Command+Alt+F7 for popup)

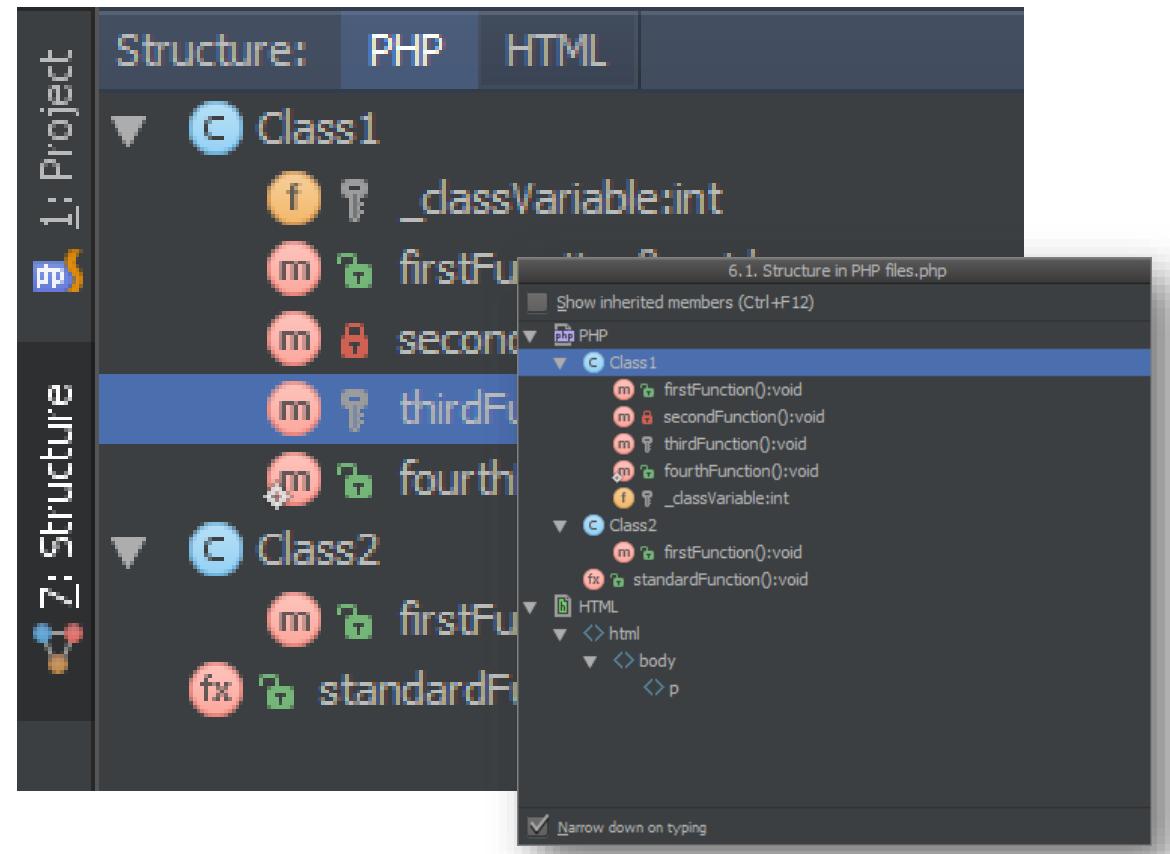
File Structure Tool Window

Displays outline of file: classes, functions (with icons displaying accessibility).

Also shows HTML, JavaScript, CSS, ...

Windows: Alt+7 (Ctrl+F12 for popup)

Mac: Command+7 (Command+F12 for popup)



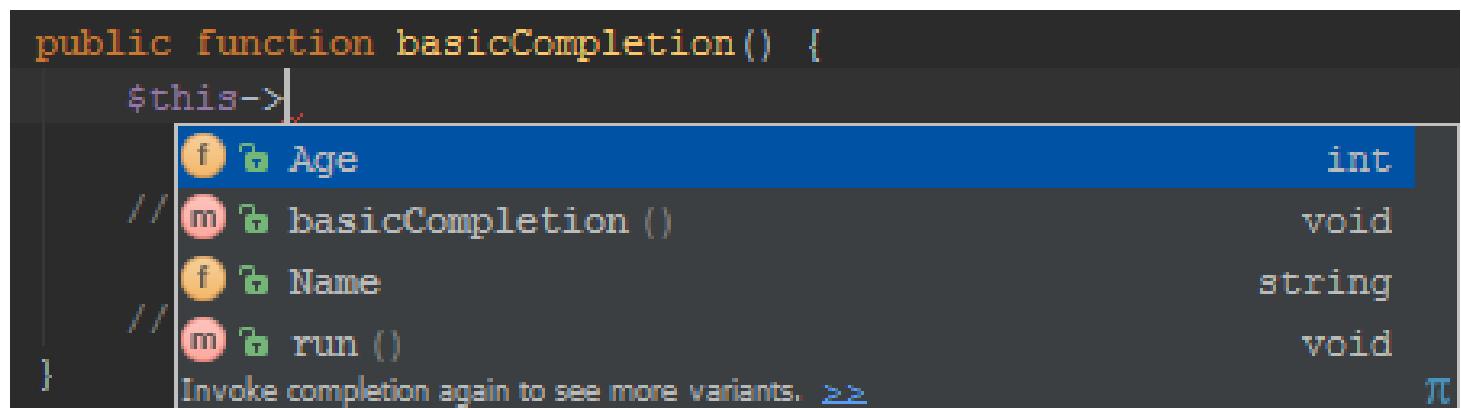
Editing

Basic Completion

Basic code completion for the name
of any class, method or variable.

Hit Ctrl+Space twice for more variants

Also allows path completion



A screenshot of an IDE showing code completion. The code being typed is `$this->`. A completion dropdown menu is open, listing the following options:

Completion Item	Type
<code>Age</code>	<code>int</code>
<code>basicCompletion ()</code>	<code>void</code>
<code>Name</code>	<code>string</code>
<code>run ()</code>	<code>void</code>

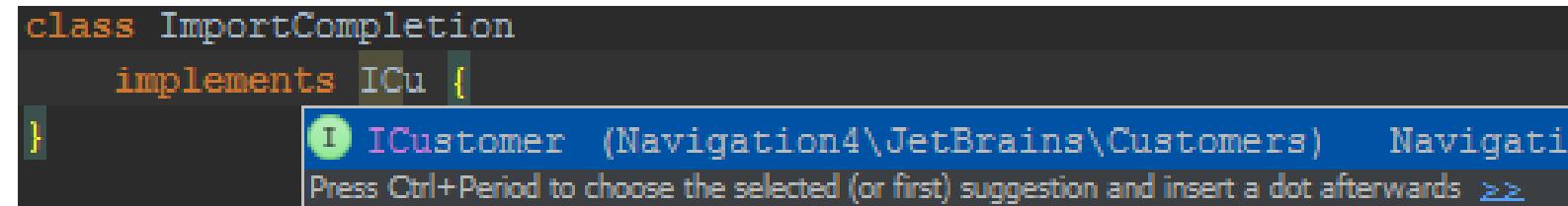
At the bottom of the dropdown, there is a message: `Invoke completion again to see more variants. >>`.

  Ctrl+Space (Ctrl+Shift+Enter to complete statement)

 Ctrl+Space (Command+Shift+Enter to complete statement)

Import Completion

Basic code completion when using classes or functions from other namespaces. Automatically adds import when selected.



A screenshot of an IDE showing code completion. The code being typed is:

```
class ImportCompletion
    implements ICu {
```

The cursor is at the end of the line 'implements ICu {'. A tooltip is displayed below the cursor, showing the suggestion 'ICustomer' with a green circular icon containing an 'I'. The tooltip also contains the text '(Navigation4\JetBrains\Customers)' and 'Press Ctrl+Period to choose the selected (or first) suggestion and insert a dot afterwards >>'. This indicates that the IDE is suggesting the import statement 'import Navigation4.JetBrains.Customers;'. The background shows parts of the IDE interface, including a navigation bar with icons for file, edit, and search.

Windows: Ctrl+Space (Ctrl+Shift+Enter to complete statement)

Mac OS X: ⌘+Space (Command+Shift+Enter to complete statement)

Selecting Code

Various ways of selecting code.

```
public function extendAndShrinkSelection() {  
    $importantValue = 32;  
    if ($importantValue > 42) {  
        try {  
            echo 'More important than 42?';  
        } catch (Exception $ex) {  
            // 3. Place the cursor on the $ex variable. I  
            echo $ex->getMessage();  
        }  
    }  
}
```

Windows icon Linux icon Ctrl+W to expand, Ctrl+Shift+W to shrink

Mac icon Alt+Up / Down

Column Selection

Toggle column selection. Allows editing multiple lines in one go.

Windows: Alt+Shift+Insert

Mac: Command+Shift+Insert

```
public function columnSelection() {  
    $fo = 3;  
    $fo = 6;  
    $fo = 9;  
    $fo = 'a';  
    $fo = 'b';  
    $fo = 'c';  
}
```

Moving Code

Move code or entire statement up/down.

```
if (3 == 3) {  
    // ...  
    if (1 == 1) {  
        // ...  
        echo 'Number one';  
    }  
    echo 'Number three';  
}
```

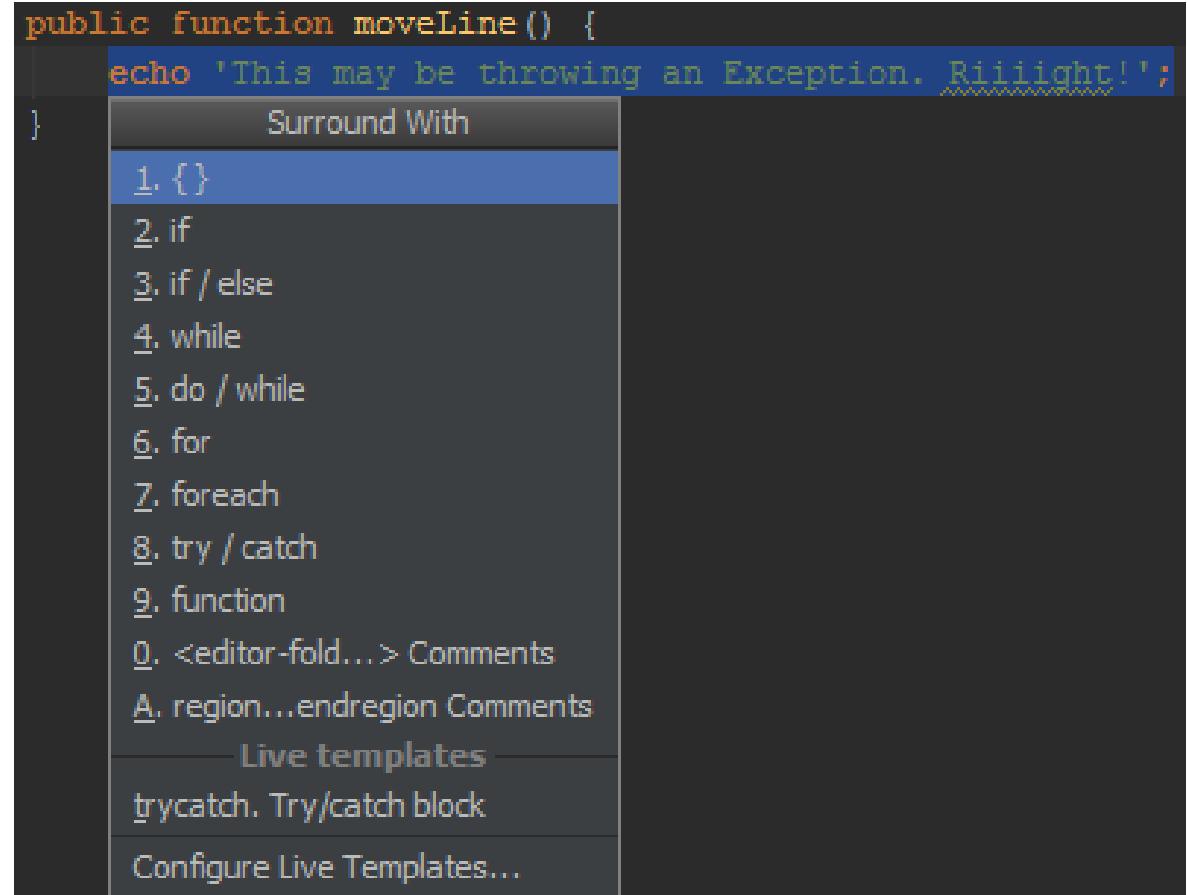
- Windows: Shift+Alt+Up / Down (or Shift+Ctrl+Up / Down for statement)
- Mac OS X: Shift+Alt+Up / Down (or Shift+Command+Up / Down for statement)

Surround With

Wraps selected text with new content, e.g. try/catch or if statement.

Windows: Ctrl+Alt+T and Alt+Enter

Mac: Command+Alt+T and Alt+Enter



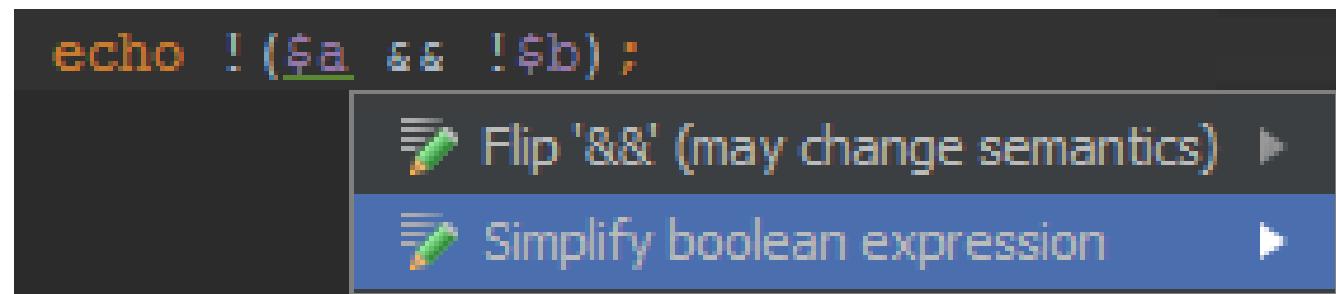
A screenshot of an IDE's code editor showing a context menu. The menu is titled 'Surround With' and lists several options: 1. {} (which is highlighted), 2. if, 3. if / else, 4. while, 5. do / while, 6. for, 7. foreach, 8. try / catch, 9. function, 0. <editor-fold...> Comments, A. region...endregion Comments, Live templates (disabled), trycatch. Try/catch block, and Configure Live Templates... The code in the editor is:

```
public function moveLine() {  
    echo 'This may be throwing an Exception. Riiight!';  
}
```



Intentions

Let the IDE figure out possible actions from context and execute them.



Windows icon Alt+Enter

Linux icon Alt+Enter

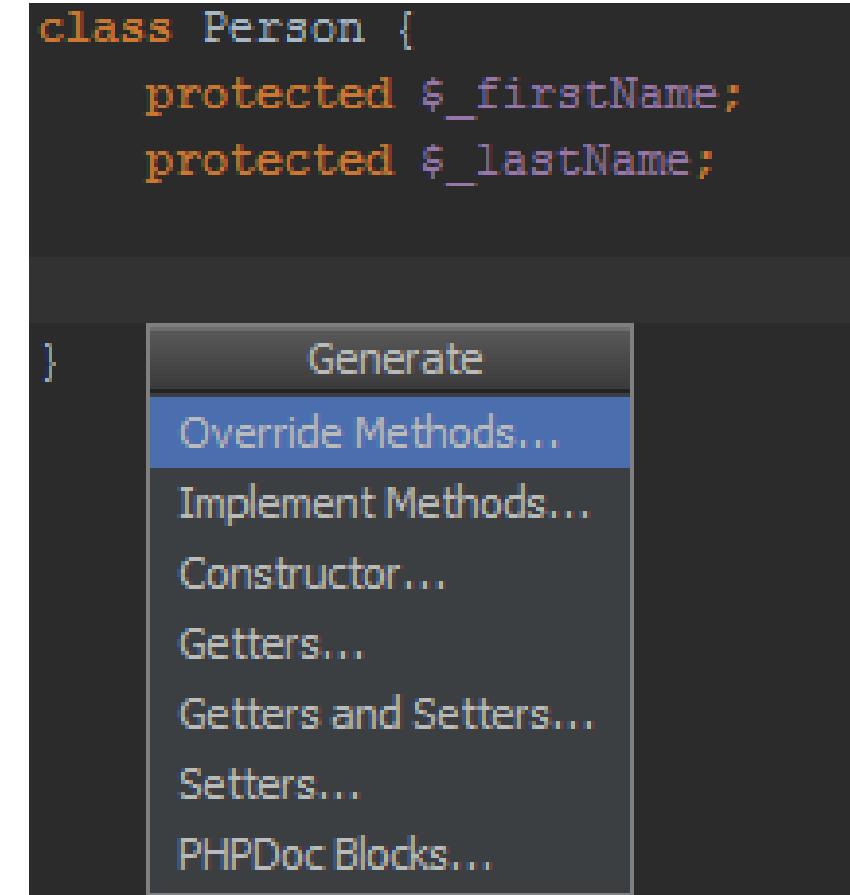
Generate Code

Generate code, e.g. class members, constructor, docblock comments, fields, ...

When used in navigation bar / project tool window, generates a new file.

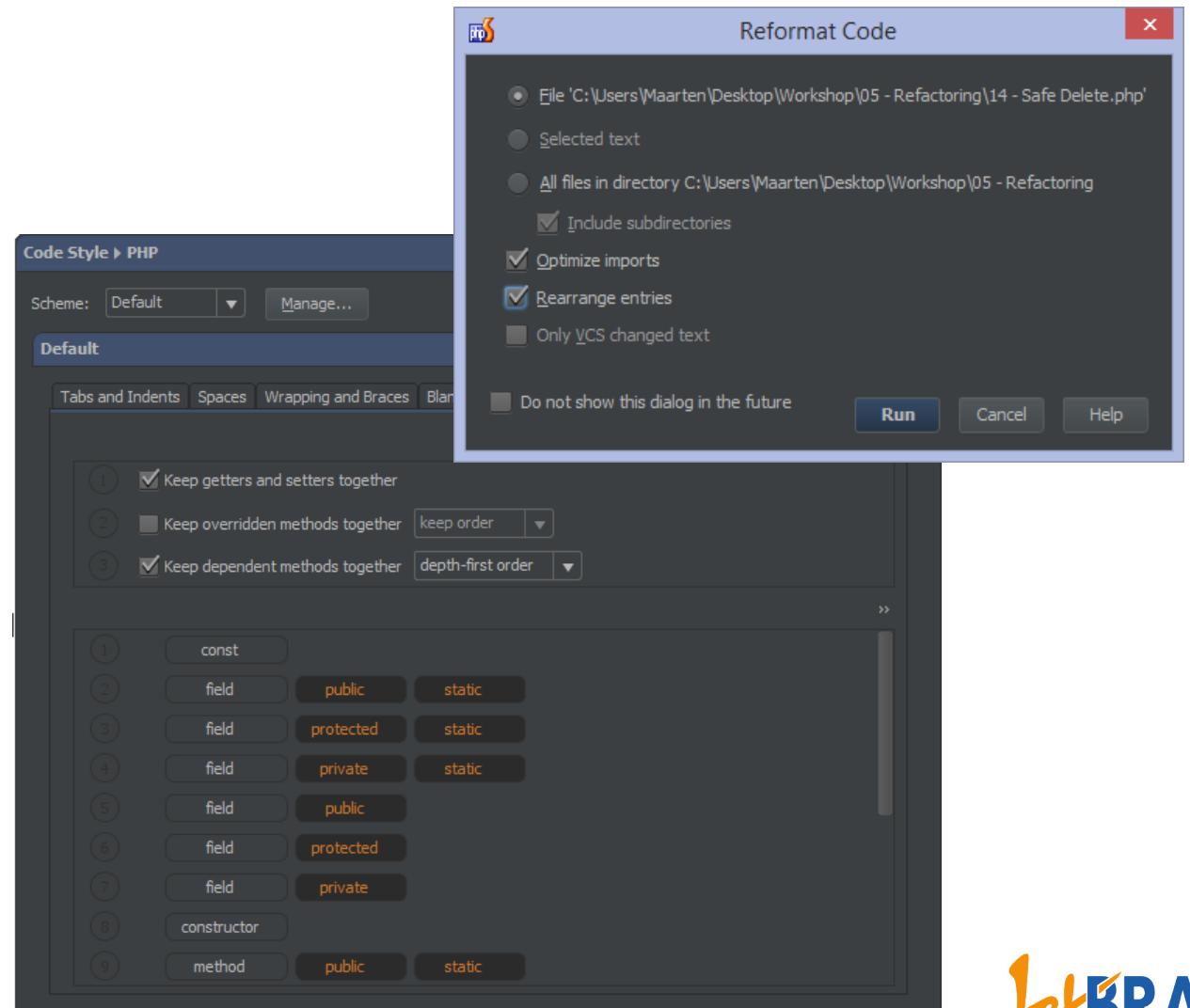
Windows icon Alt+Insert

Mac OS X icon Command+N



Rearrange/Reformat Code

Rearranges entries in code according to settings. Reformats all code according to code style.



Windows: Ctrl+Alt+L (reformat)

Mac: Command+Alt+L (reformat)

JetBrains

Inspections

Highlights

In-editor highlighting of code issues.

Errors

Warnings

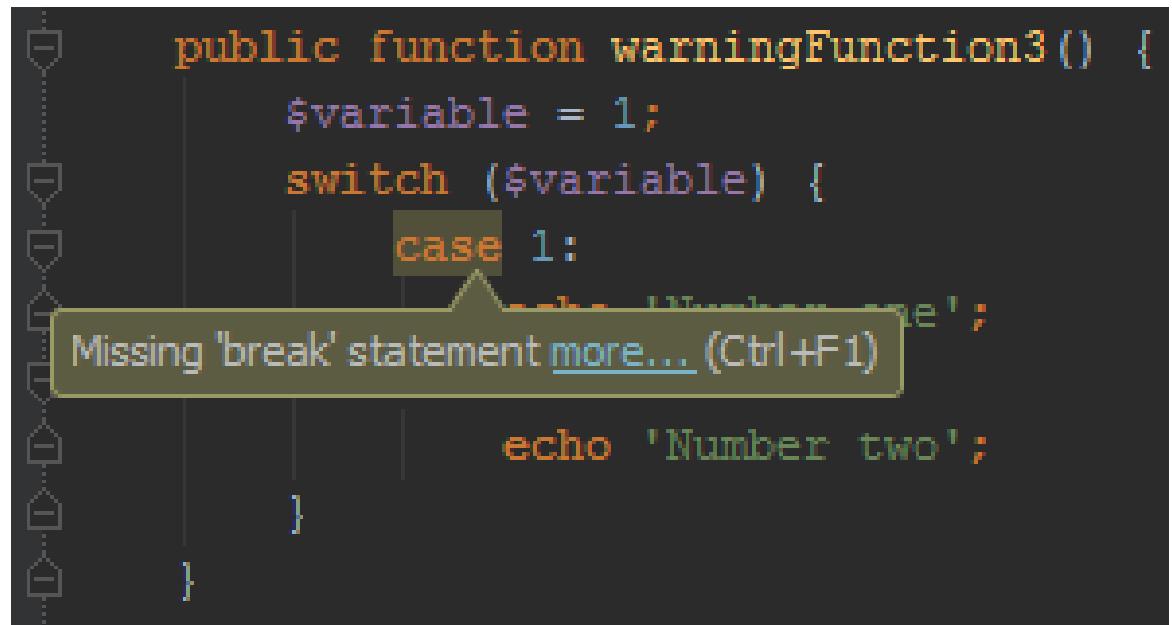
Suggestions

Hints

Dead Code

Can be suppressed.

Support for PHP Code Sniffer / PHP Mess Detector.



A screenshot of an IDE showing a PHP code editor. The code is as follows:

```
public function warningFunction3() {  
    $variable = 1;  
    switch ($variable) {  
        case 1:  
            echo 'Number one';  
        case 2:  
            echo 'Number two';  
    }  
}
```

A tooltip is displayed over the 'case 1:' line, pointing to the word 'case'. The tooltip contains the text "Missing 'break' statement [more...](#) (Ctrl+F1)".

Quick Fixes

Highlights provide a Quick Fix action to help remove the warning.



Windows: Alt+Enter

Mac: Alt+Enter

Navigation

Navigate back and forth between code issues.

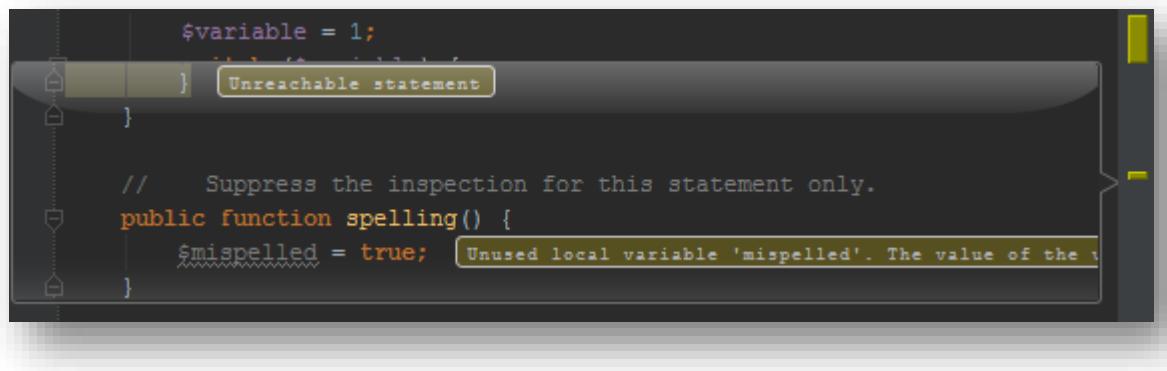


Windows icon F2 / Shift+F2

Mac OS X icon F2 / Shift+F2

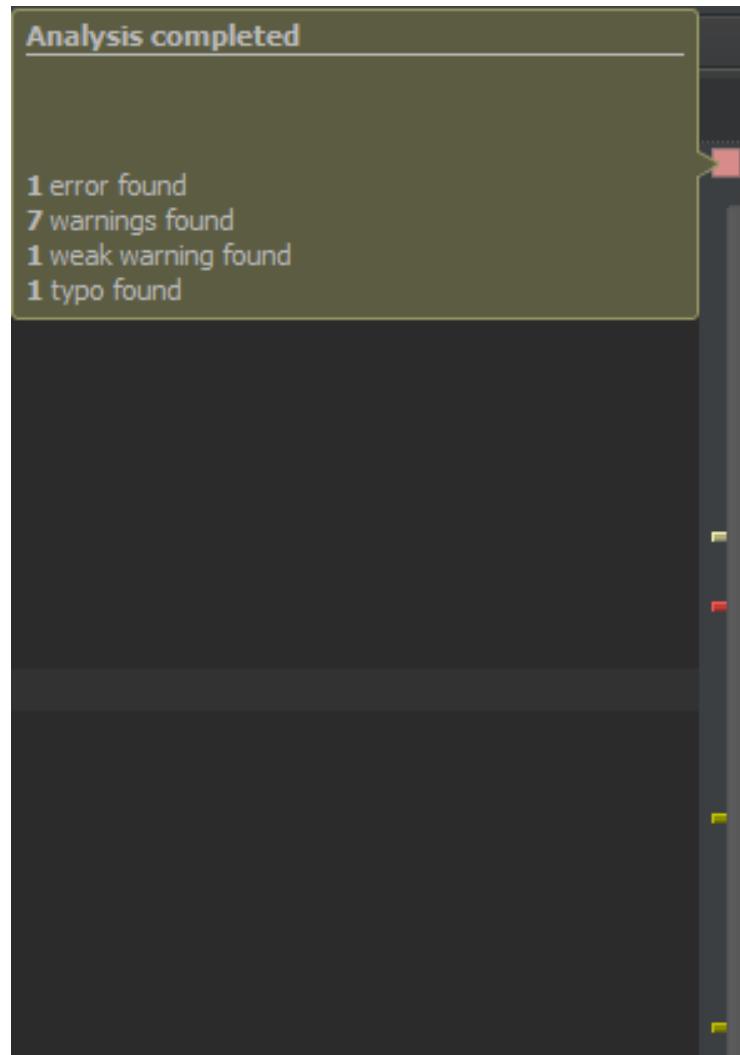
Gutter and Lens

Right gutter displays error information. Use Lens mode for preview.



A screenshot of an IDE interface showing a code editor and a tool window. The code editor contains the following PHP-like code:\$variable = 1;
}
} // Unreachable statement

// Suppress the inspection for this statement only.
public function spelling()
{
 \$misplaced = true; // Unused local variable 'misplaced'. The value of the v
}The right gutter shows inspection annotations: 'Unreachable statement' for the final brace, and 'Unused local variable 'misplaced''. The tool window shows the results of an analysis:

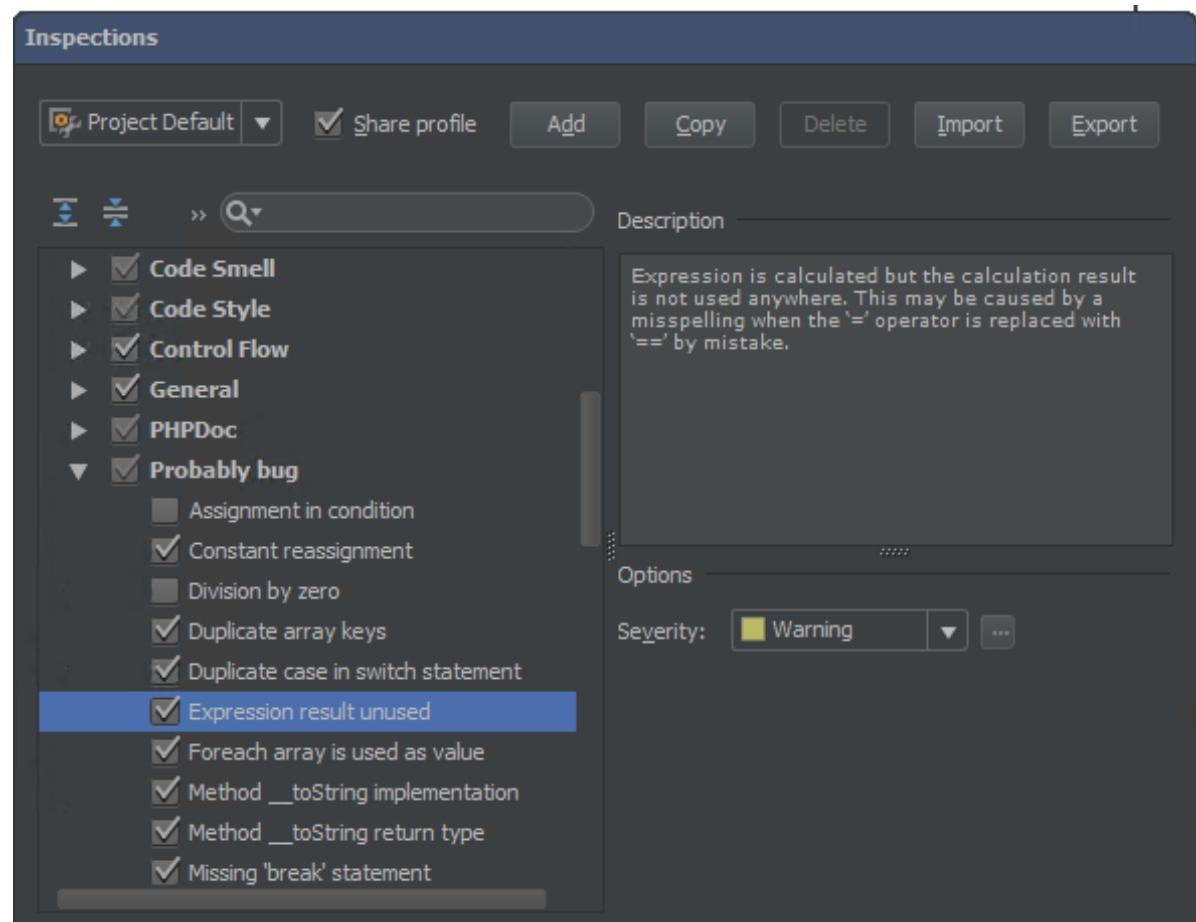


Settings

Enable/disable inspections, see examples and documentation.
Configure severity.

Windows icon Ctrl+Alt+S

Mac icon Command+,

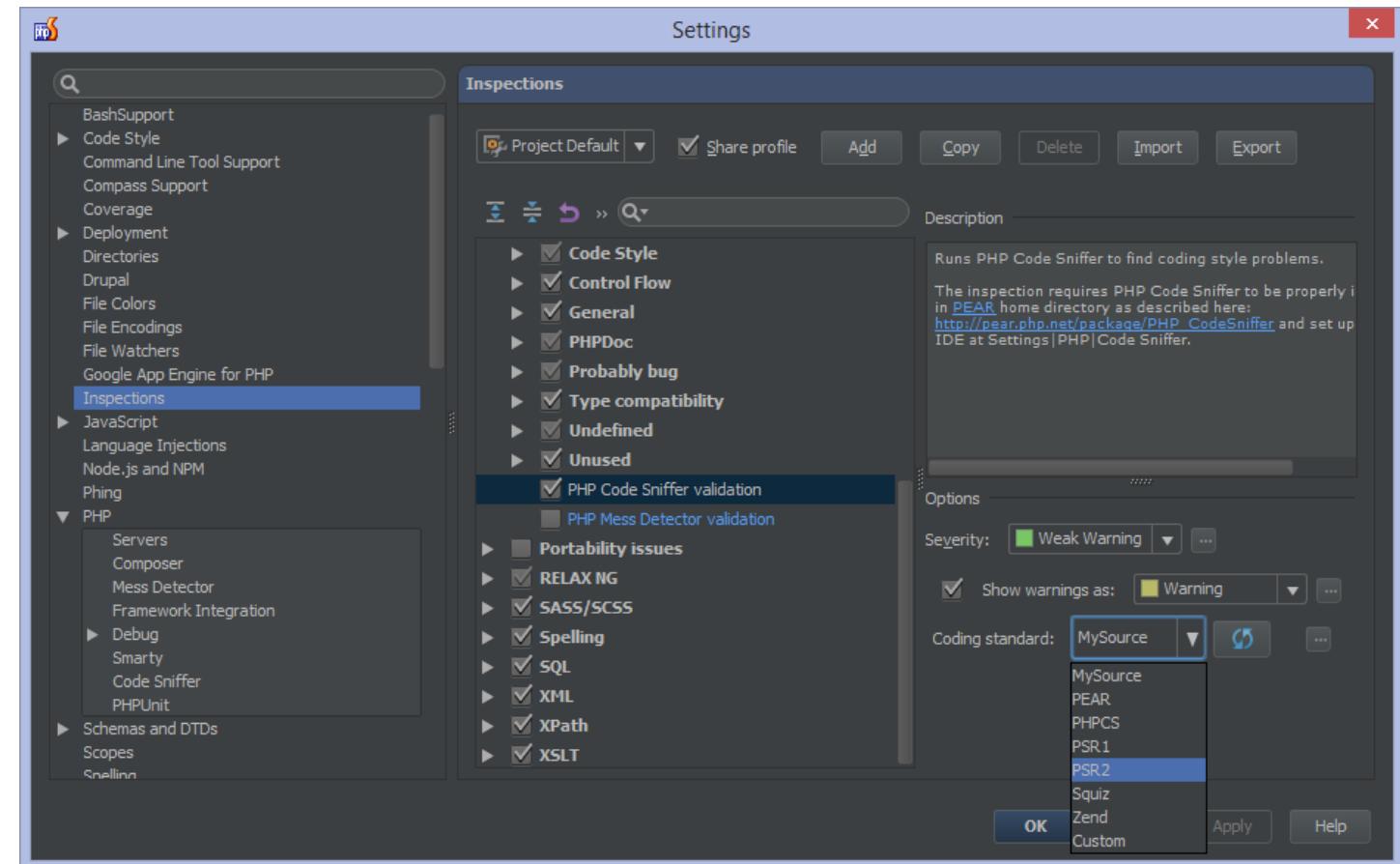


PHPMD / PHPCS

PHP Mess Detector and PHP Code Sniffer can be plugged in for inspections as well.

Windows icon Linux icon Ctrl+Alt+S

Mac icon Command+,



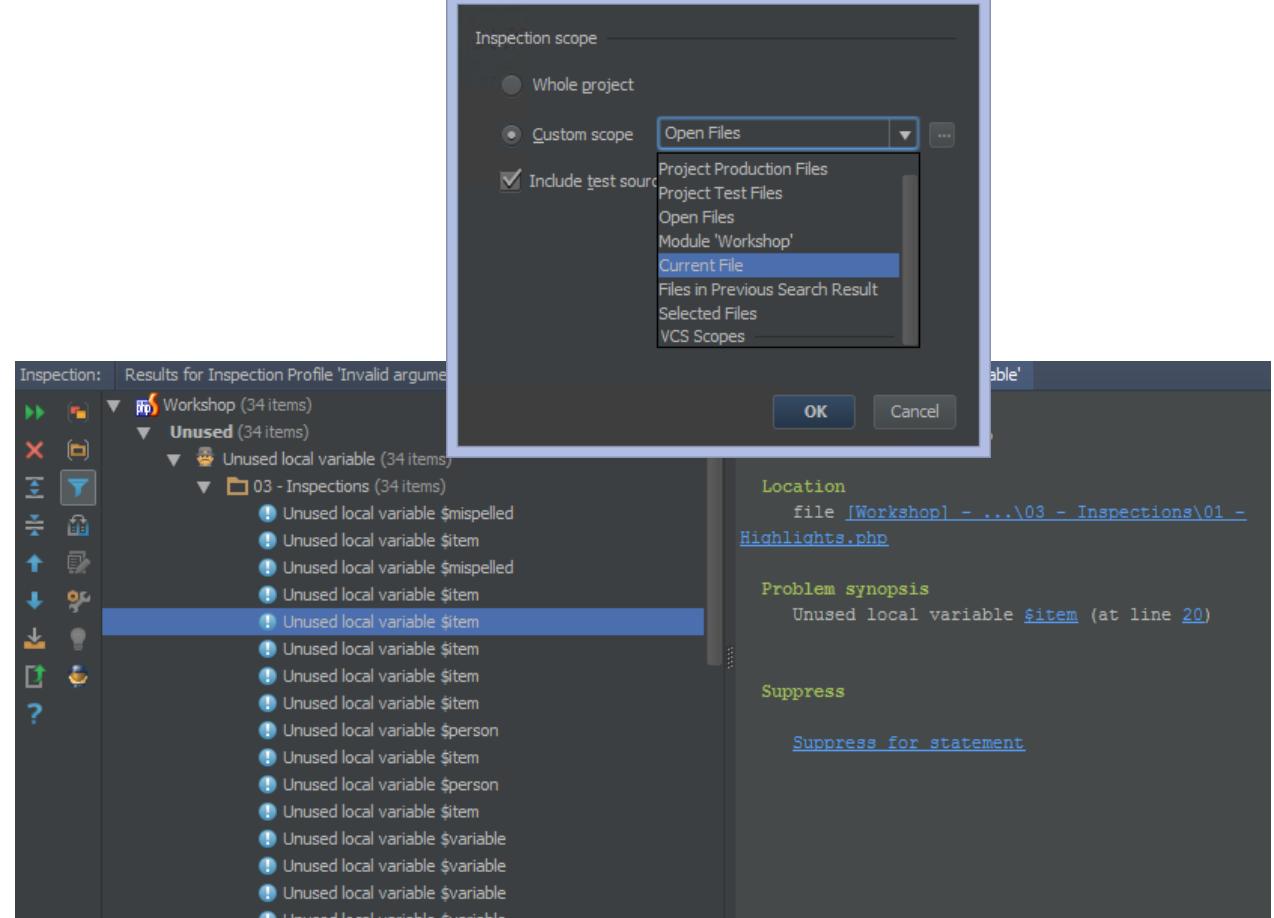
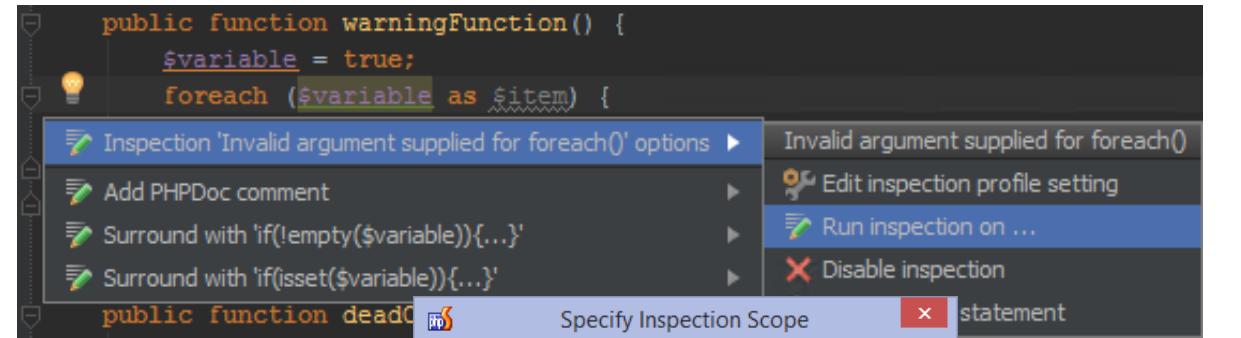
JetBrains

Find Issues

Quick fix menu allows finding all similar issues.

Windows icon Alt+Enter

Mac OS X icon Alt+Enter

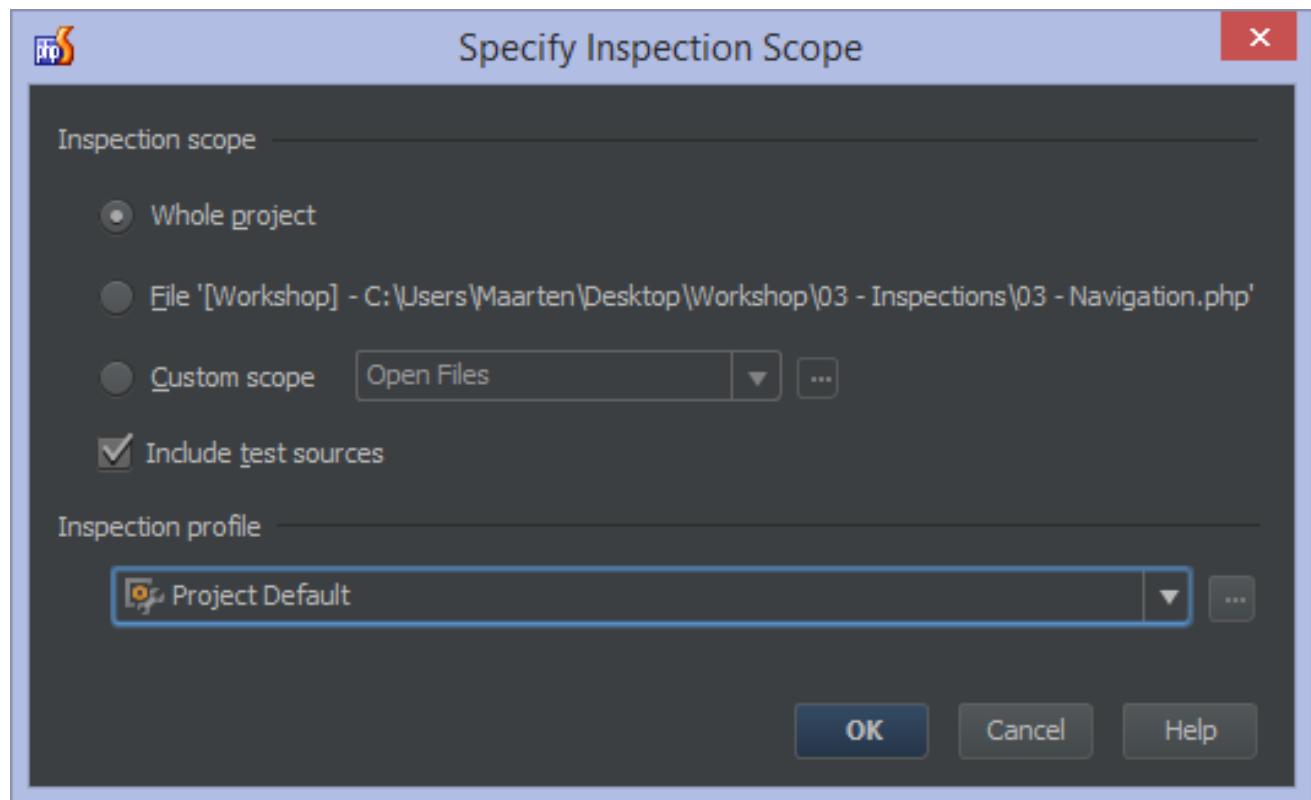


JetBrains

Run Inspections

Run an inspection profile and get results for the entire project (or scope).

Windows: Ctrl+Alt+Up / Down
Mac OS X: Command+Alt+Up / Down



Live templates (snippets)

Code Expansion

Expand text shortcut using template.

Expands into code with variable “hotspots”

Hotspot can be linked to an expression, such as “current user”, “autocomplete”, ...

Tab to move between hotspots.

Template can define end point for caret.

```
$numbers = array(1,2,3,4,5,6,7,8,9);  
  
foreach ($numbers as $item) {  
  
}
```



Creating Live Templates

Create your own live templates.

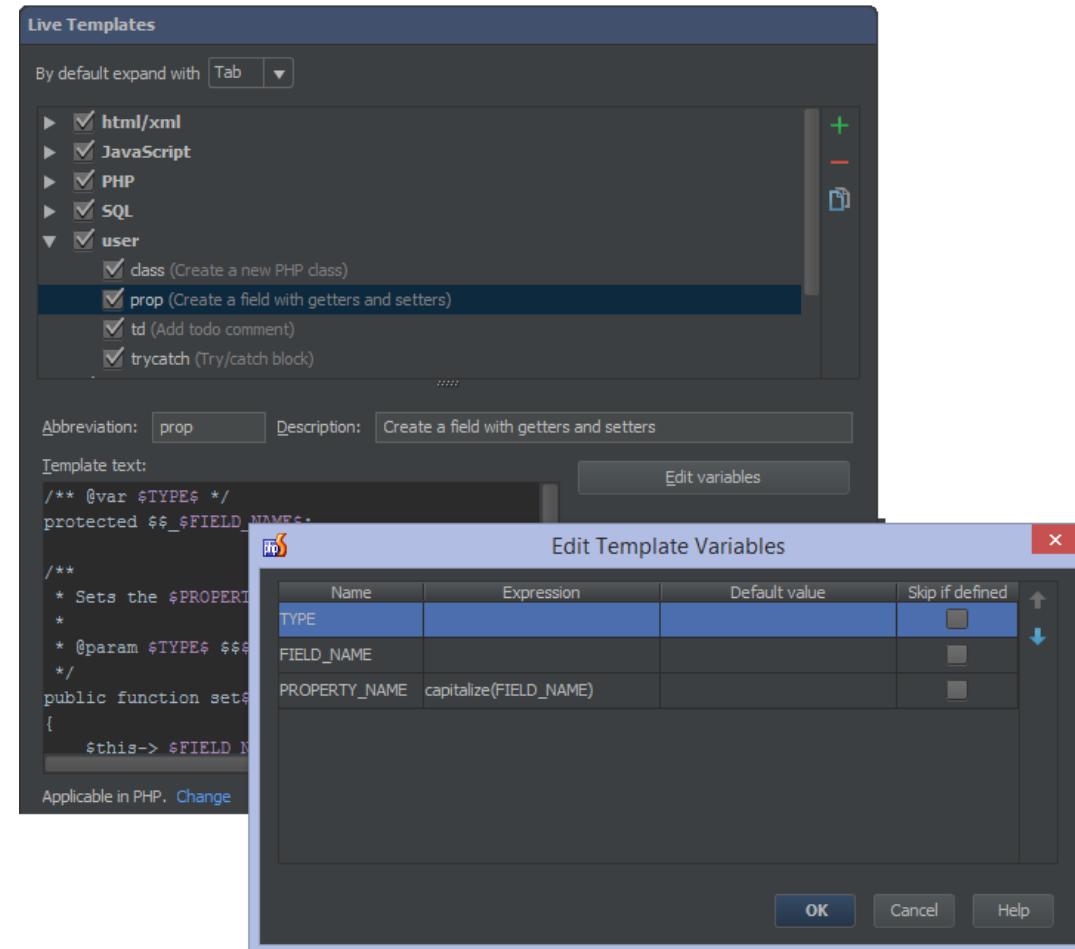
Use variables and expressions.

Can be created from settings or straight from within the editor.

`END` variable denotes where caret should be after expansion.

Expressions:

<https://www.jetbrains.com/phpstorm/webhelp/edit-template-variables-dialog.html>



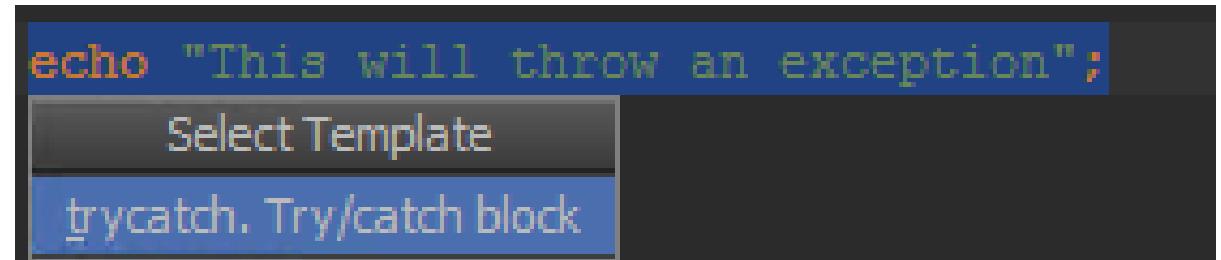
Surround Templates

Surround code with a template.

Extra predefined variable:

`$SELECTION$` is the currently selected text

Only considered a surround template when
this macro is used.



Windows: Ctrl+Alt+T or Ctrl+Alt+J

Mac OS X: Command+Alt+T or Command+Alt+J

File Templates

Applied when creating a new file or generating code.

Can contain includes.

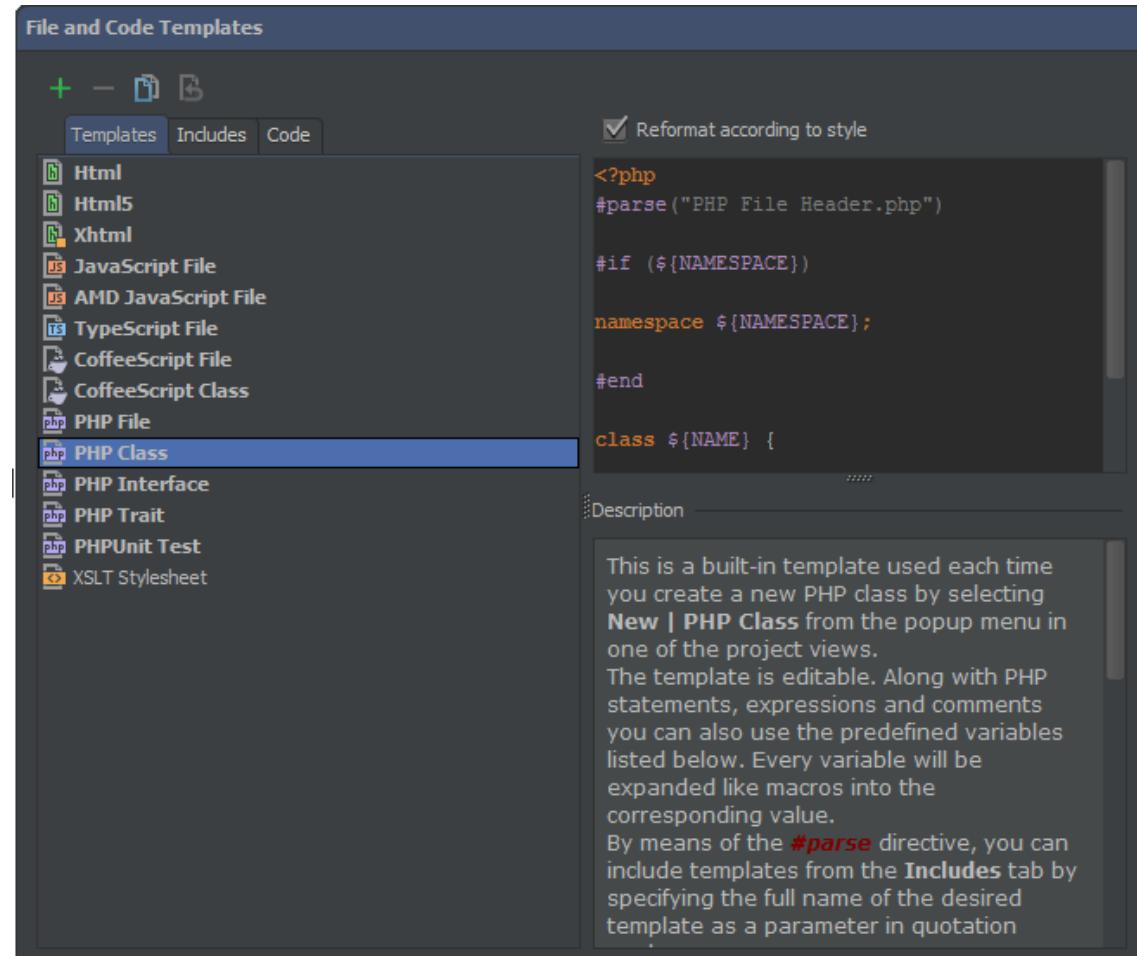
Customize code generation.

Variables:

<https://www.jetbrains.com/phpstorm/webhelp/file-template-variables.html>

  Ctrl+Alt+S

 Command+,



Refactoring

“Refactoring is a controlled technique for improving the design of an existing code base.”

Martin Fowler

<http://martinfowler.com/books/refactoring.html>



Refactoring in PhpStorm

The IDE will:

- Perform the refactoring
- Track down and correct the affected code references automatically
- Warn about occurrences it can not update automatically

Available refactorings:

- Change Signature
- Copy/Clone
- Extract Constant
- Extract Field
- Extract Interface
- Extract Method
- Extract Parameter
- Extract Variable
- Inline
- Move Refactorings
- Pull Members up
- Push Members down
- Rename Refactorings
- Safe Delete

Refactor This

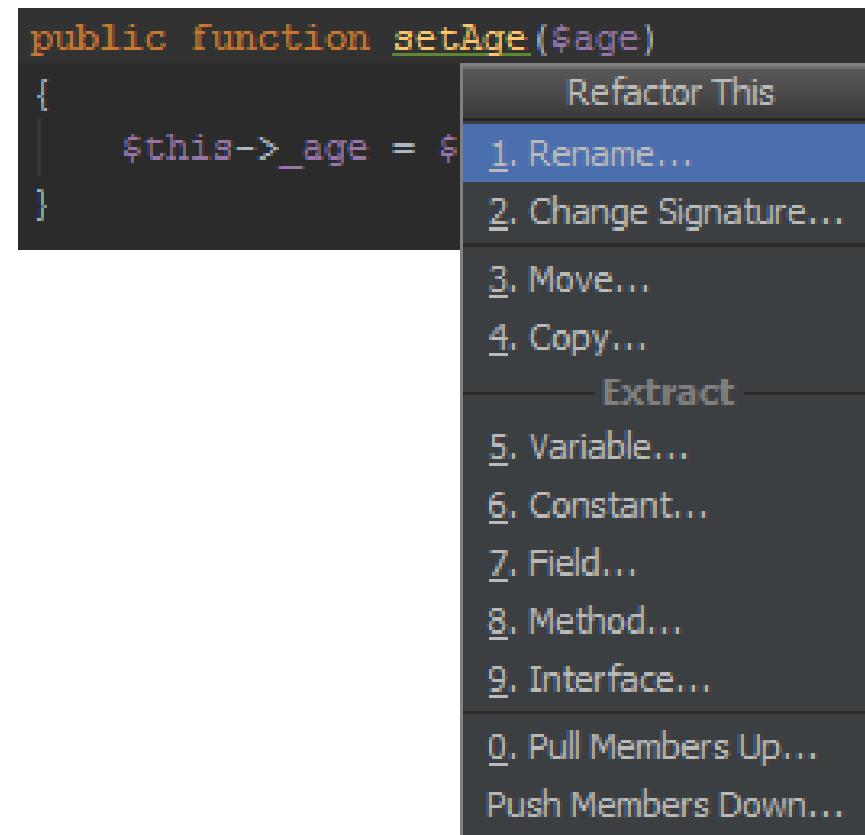
Refactor symbol or code fragment

In Project View, Structure Tool Window,
Editor or UML Class Diagram.

Use menu item Refactor | Refactor
This or keyboard shortcut.

  Ctrl+Shift+Alt+T

 Ctrl+T



Change Signature

<http://www.jetbrains.com/phpstorm/webhelp/change-signature.html>

You can use this refactoring to:

Change the function name.

Add new parameters and remove the existing ones.

Assign default values to the parameters.

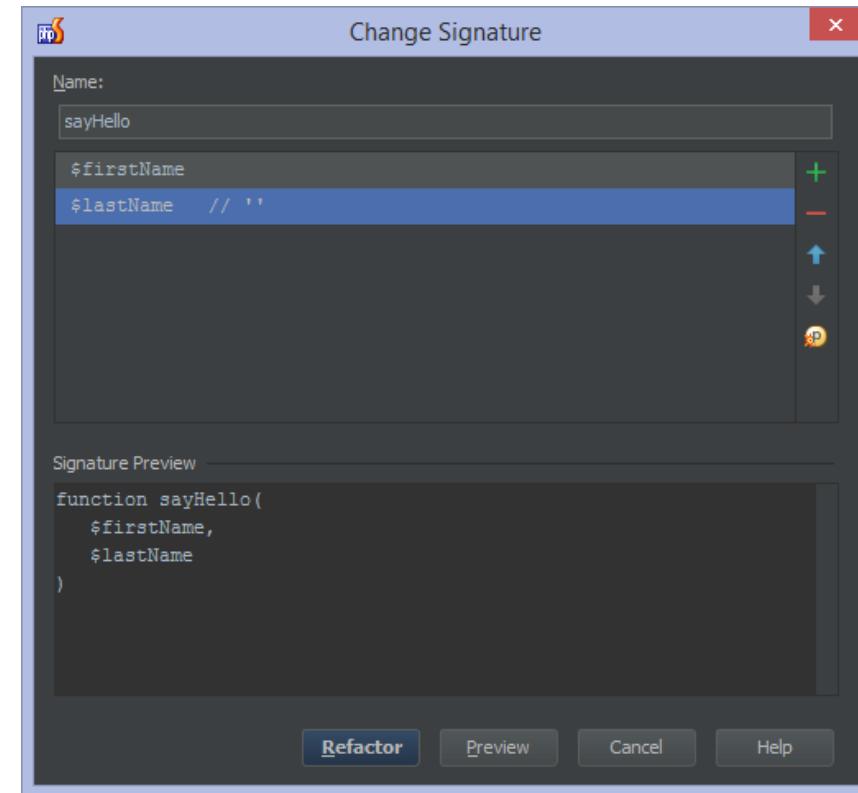
Reorder parameters.

Change parameter names.

Propagate new parameters through the function call hierarchy.

Windows icon Ctrl+F6

Mac icon Command+F6



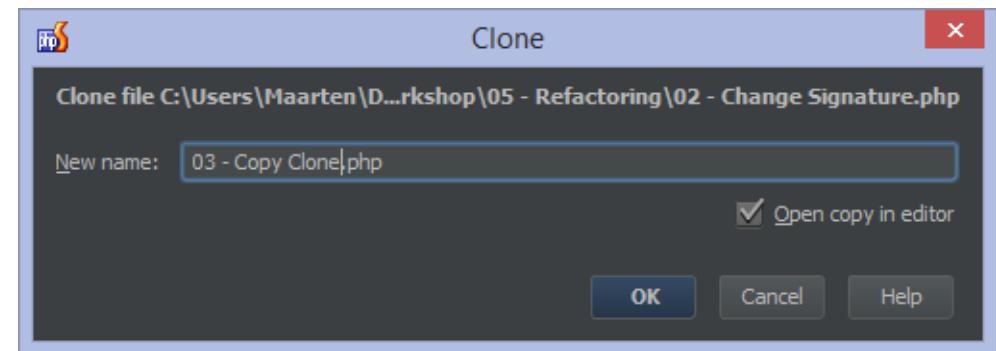
Copy/Clone

<http://www.jetbrains.com/phpstorm/webhelp/copy-clone.html>

Copy a class, file or directory to another directory or clone it within the same directory.

Using keyboard shortcut

Using drag/drop with Ctrl key pressed



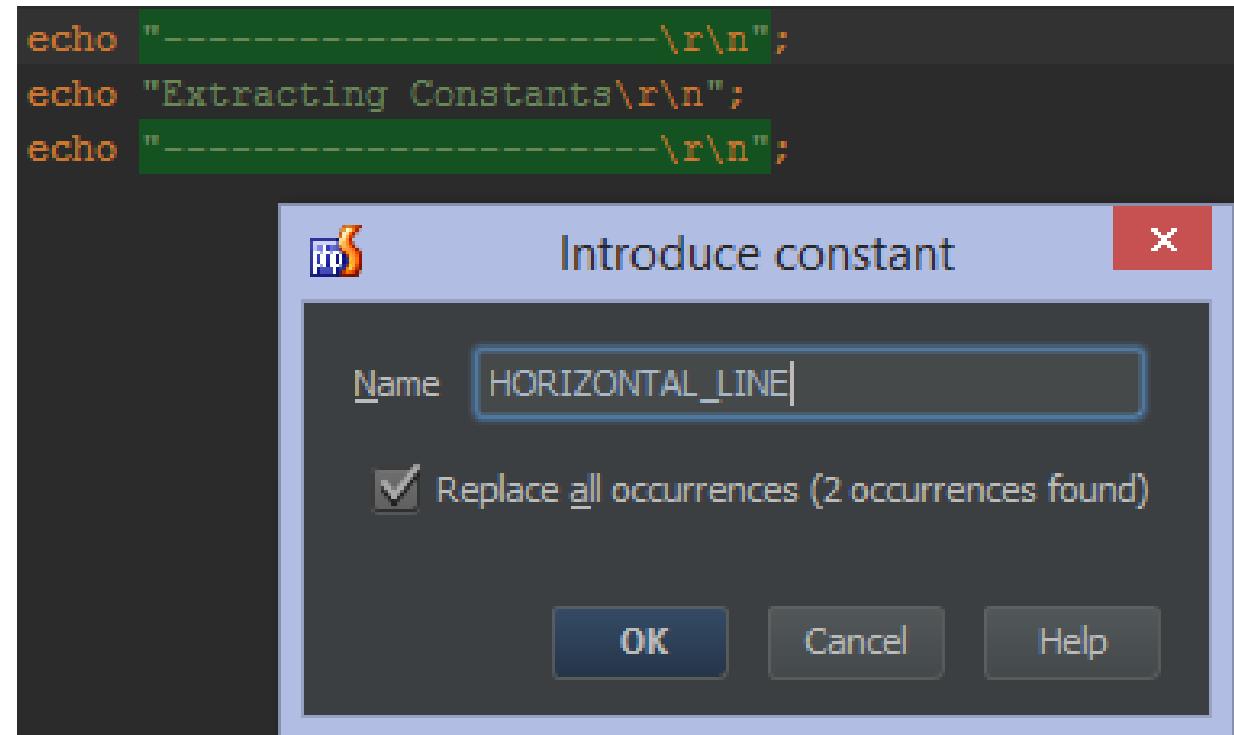
Windows icon F5 (copy), Shift+F5 (clone)

Mac OS X icon F5 (copy), Shift+F5 (clone)

Extract Constant

<http://www.jetbrains.com/phpstorm/webhelp/extract-constant.html>

Extract a constant to make code cleaner and more maintainable.



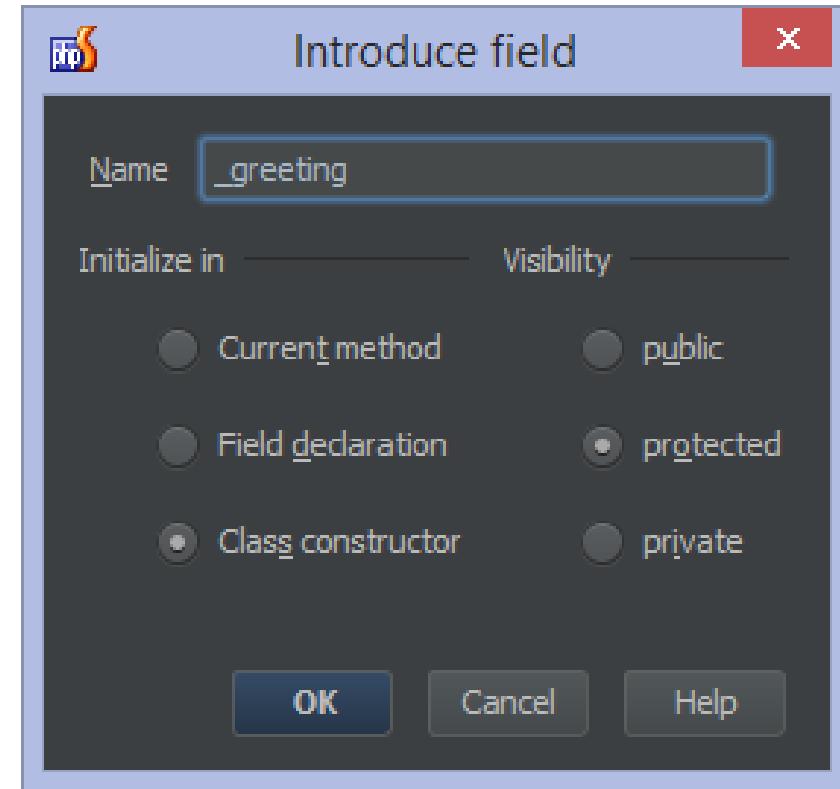
Windows icon Ctrl+Alt+C

Mac icon Command+Alt+C

Extract Field

<http://www.jetbrains.com/phpstorm/webhelp/extract-field.html>

Extract an expression into a field.



Windows icon Ctrl+Alt+F

Mac icon Command+Alt+F

Extract Interface

<http://www.jetbrains.com/phpstorm/webhelp/extract.html>

Extract an interface from a class.

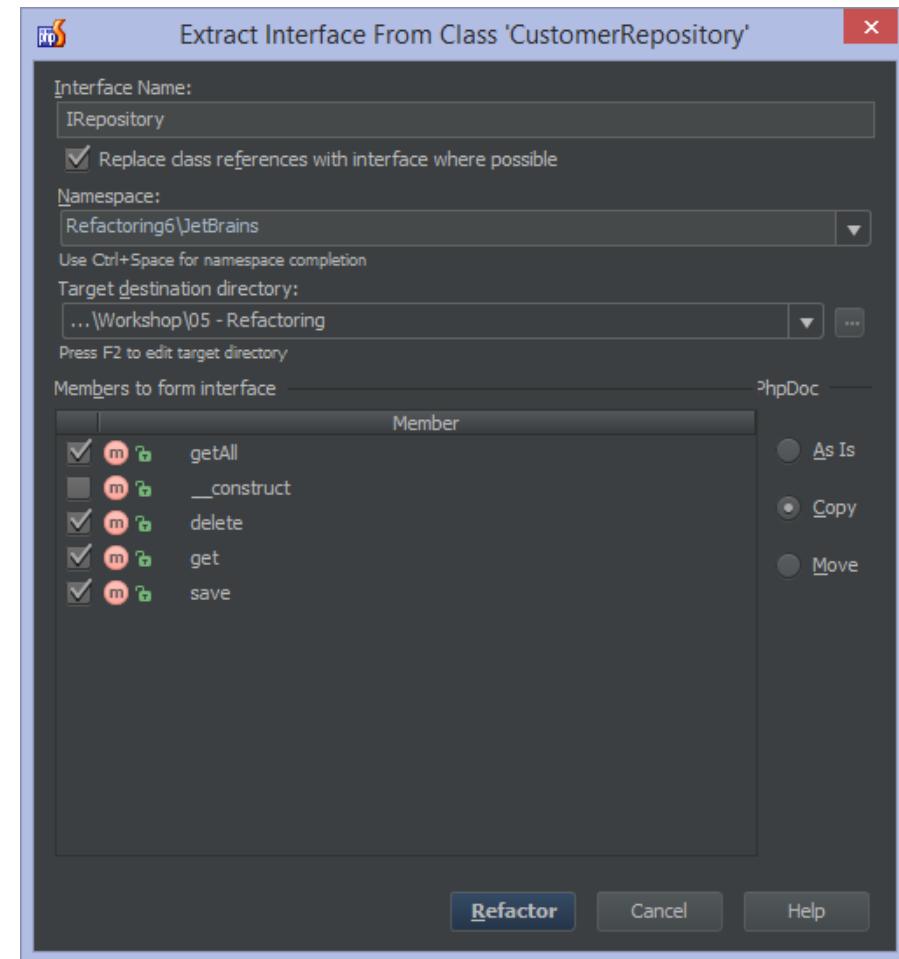
Specify a Name

Optionally chaneg namespace

Pick members to extract

Copy PHPDoc

Use the menu or Refactor This.



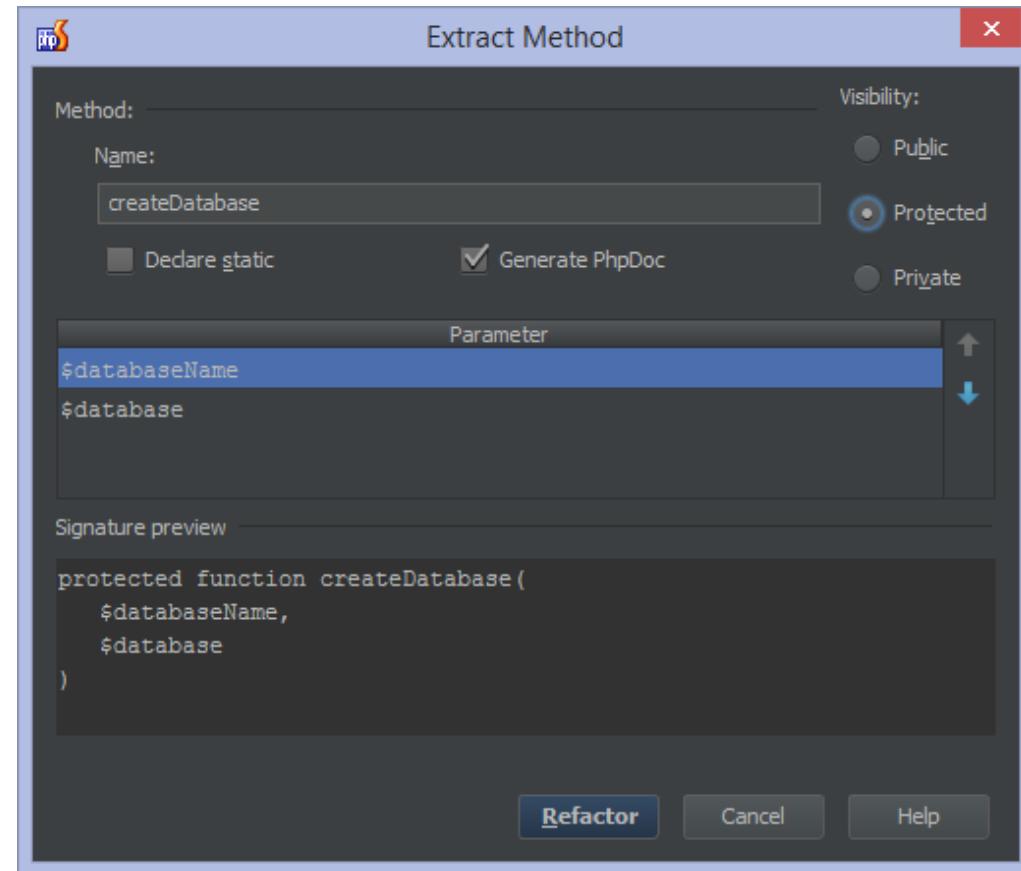
Extract Method

<http://www.jetbrains.com/phpstorm/webhelp/extract-method.html>

Extracts a block of code into a method, detecting variables.

Windows  Linux  Ctrl+Alt+M

Mac  Command+Alt+M

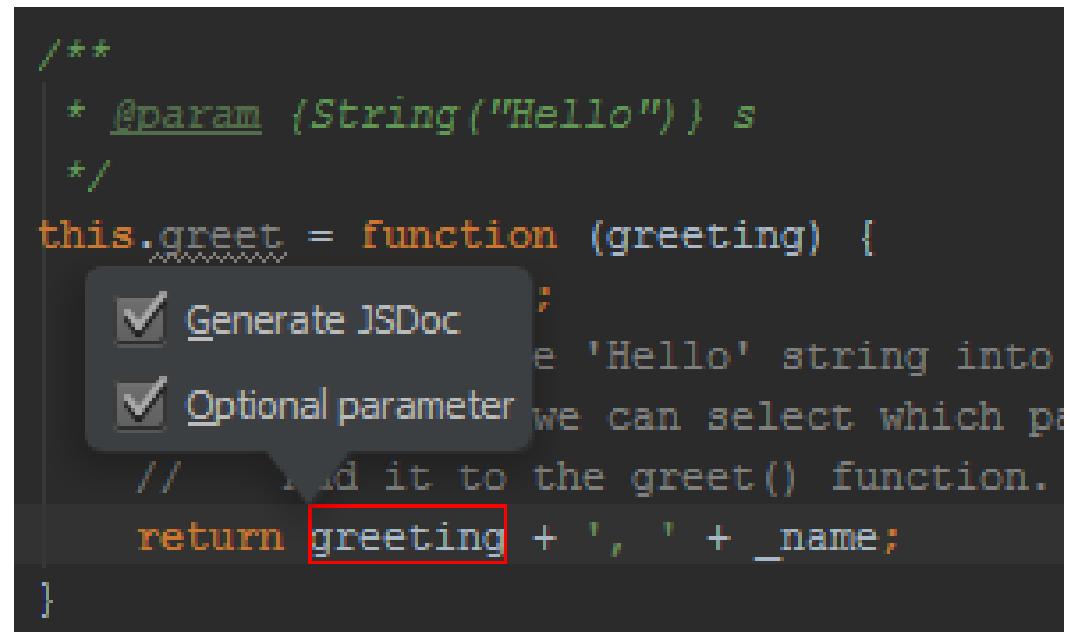




Extract Parameter

<http://www.jetbrains.com/phpstorm/webhelp/change-signature.html>

Adds a new parameter to a function declaration. Determines the default. Can generate JSDoc.



```
/**  
 * @param {String("Hello")} s  
 */  
  
this.greet = function (greeting) {  
    // ...  
    // And it to the greet() function.  
    return greeting + ', ' + _name;  
}
```

The screenshot shows a code editor with a tooltip over the 'greeting' parameter in the 'greet' function. The tooltip contains two checked checkboxes: 'Generate JSDoc' and 'Optional parameter'. The 'Optional parameter' checkbox has a descriptive text below it: 'e 'Hello' string into we can select which pa'. The 'greeting' parameter is highlighted with a red border.

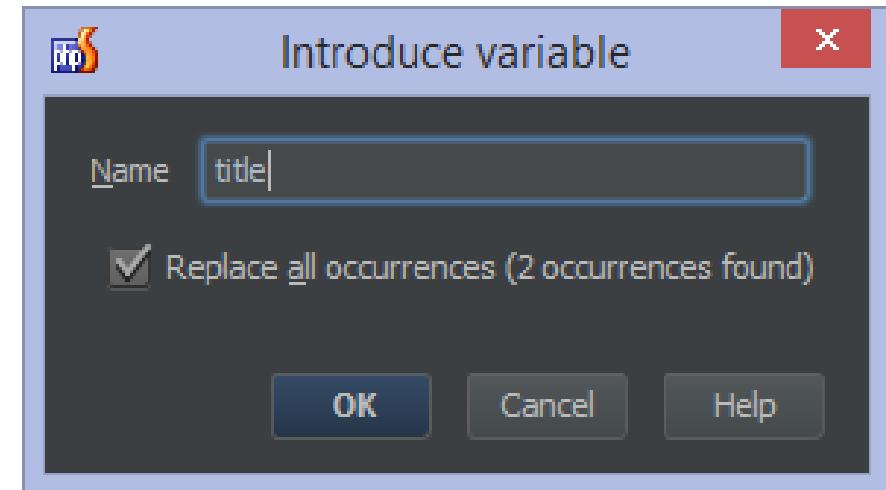
Windows: Ctrl+Alt+P

Mac: Command+Alt+P

Extract Variable

<http://www.jetbrains.com/phpstorm/webhelp/extract-variable.html>

Puts the result of a selected expression into a variable. The original expression is replaced with the new variable.



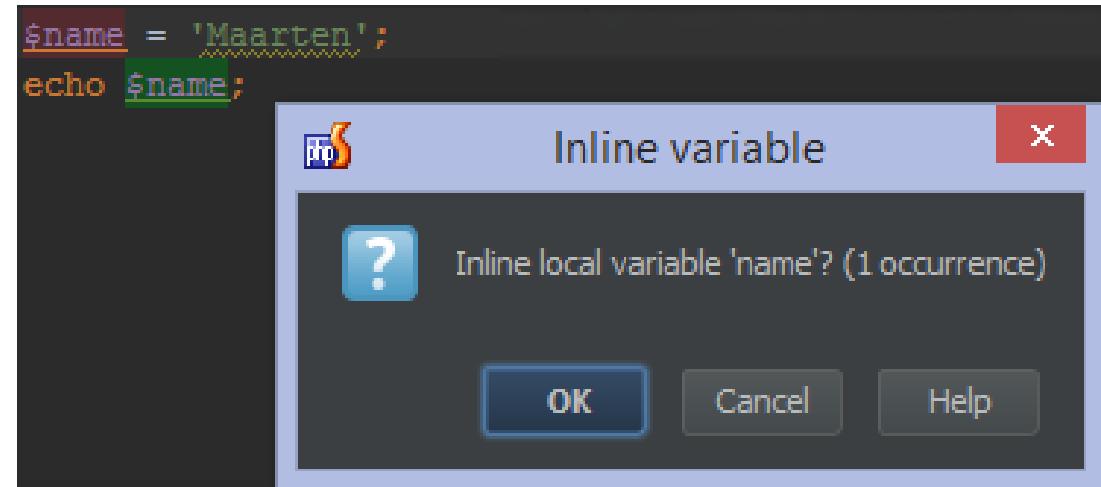
Windows: Ctrl+Alt+V

Mac OS X: Command+Alt+V

Inline

<http://www.jetbrains.com/phpstorm/webhelp/inline.html>

Replace redundant variables or functions with the full expression. It is the opposite of Extract Method.



Windows: Ctrl+Alt+N

Mac: Command+Alt+N

Move

<http://www.jetbrains.com/phpstorm/webhelp/move-refactorings.html>

Changes the location of a file,
directory, class or static member.

Move File

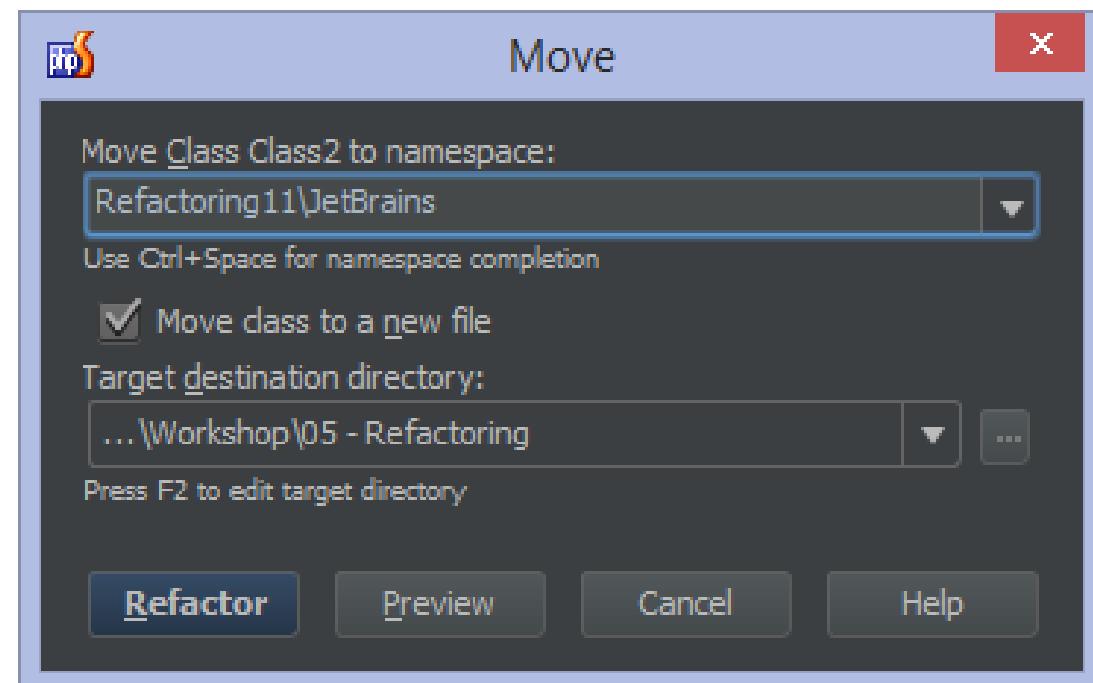
Move Directory

Move Class

Move Static Member

Windows icon F6

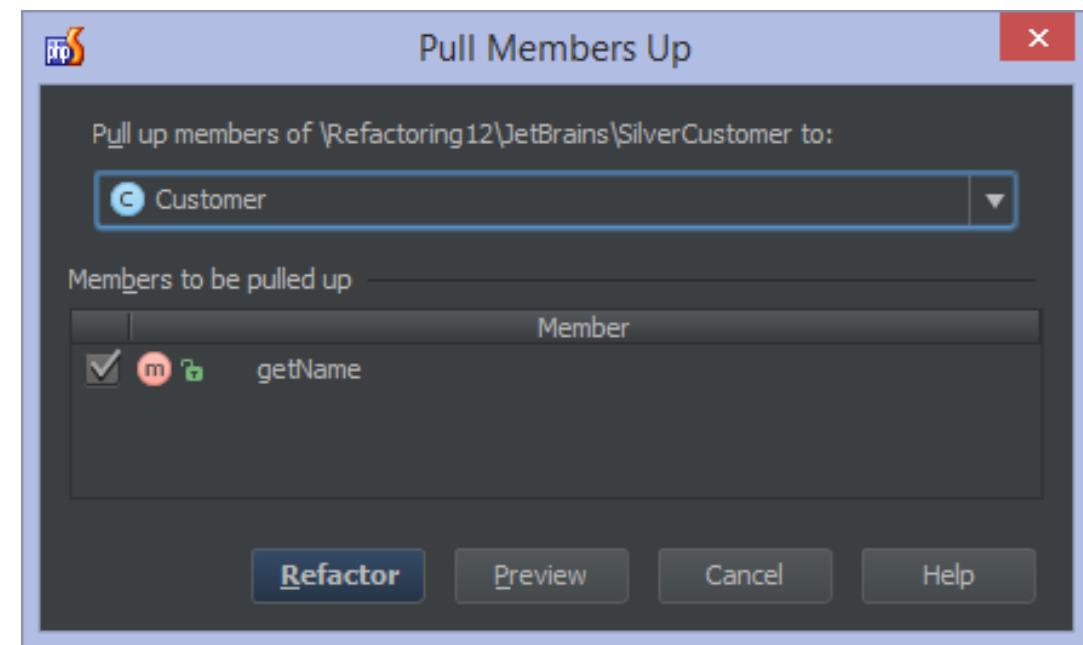
Mac OS X icon F6



Pull Members Up / Push Members Down

<http://www.jetbrains.com/phpstorm/webhelp/pull-members-up.html>

Move members from subclass to superclass or from superclass to subclass.



Rename

<http://www.jetbrains.com/phpstorm/webhelp/rename-refactorings.html>

Allow you to rename symbols,
automatically correcting all references in
the code.

Rename Class

Rename Method

Rename Field

Rename Function

Rename Variable

Rename Parameter

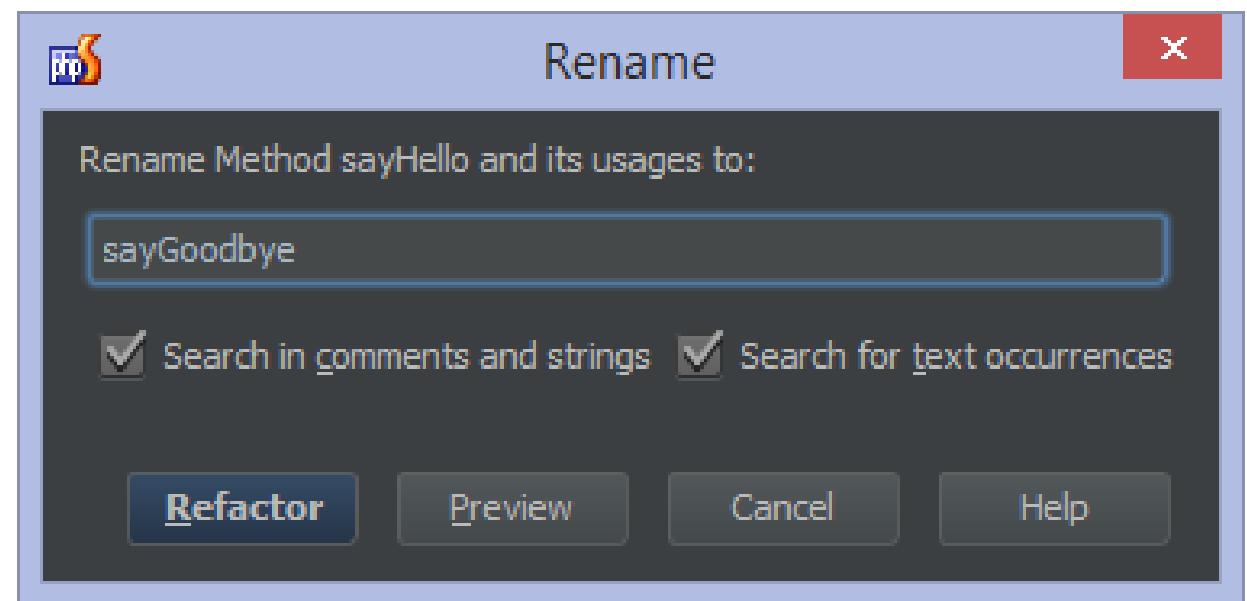
Rename CSS color value

Rename File

Rename Directory

  Shift+F6

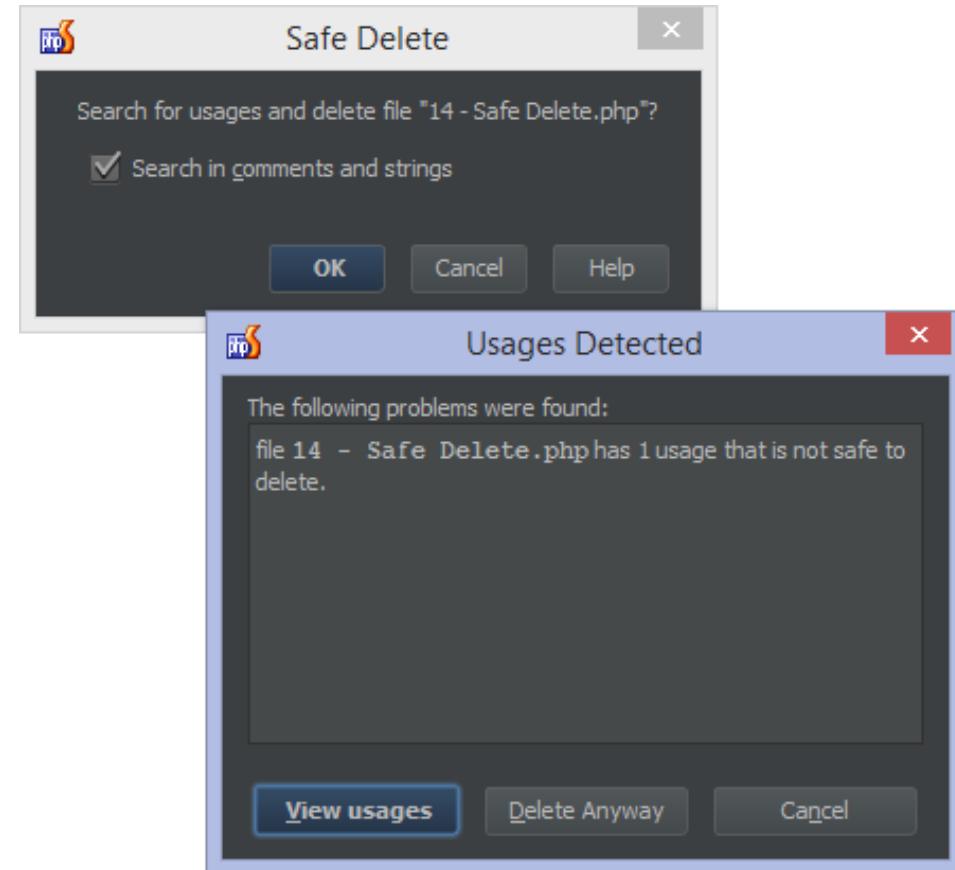
 Shift+F6



Safe Delete

<http://www.jetbrains.com/phpstorm/webhelp/safe-delete.html>

Safely remove code or symbols.



Windows icon Alt+Delete

Mac OS X icon Command+Delete

JetBRAINS

Debugging

Debugging

“Finding and reducing bugs”

In an IDE typically stepping through code & inspecting values in memory

PhpStorm needs one of these configured:

Xdebug - <http://confluence.jetbrains.com/display/PhpStorm/Xdebug+Installation+Guide>

Zend Debugger - <http://confluence.jetbrains.com/display/PhpStorm/Zend+Debugger+Installation+Guide>

Most features supported by both debuggers

Settings

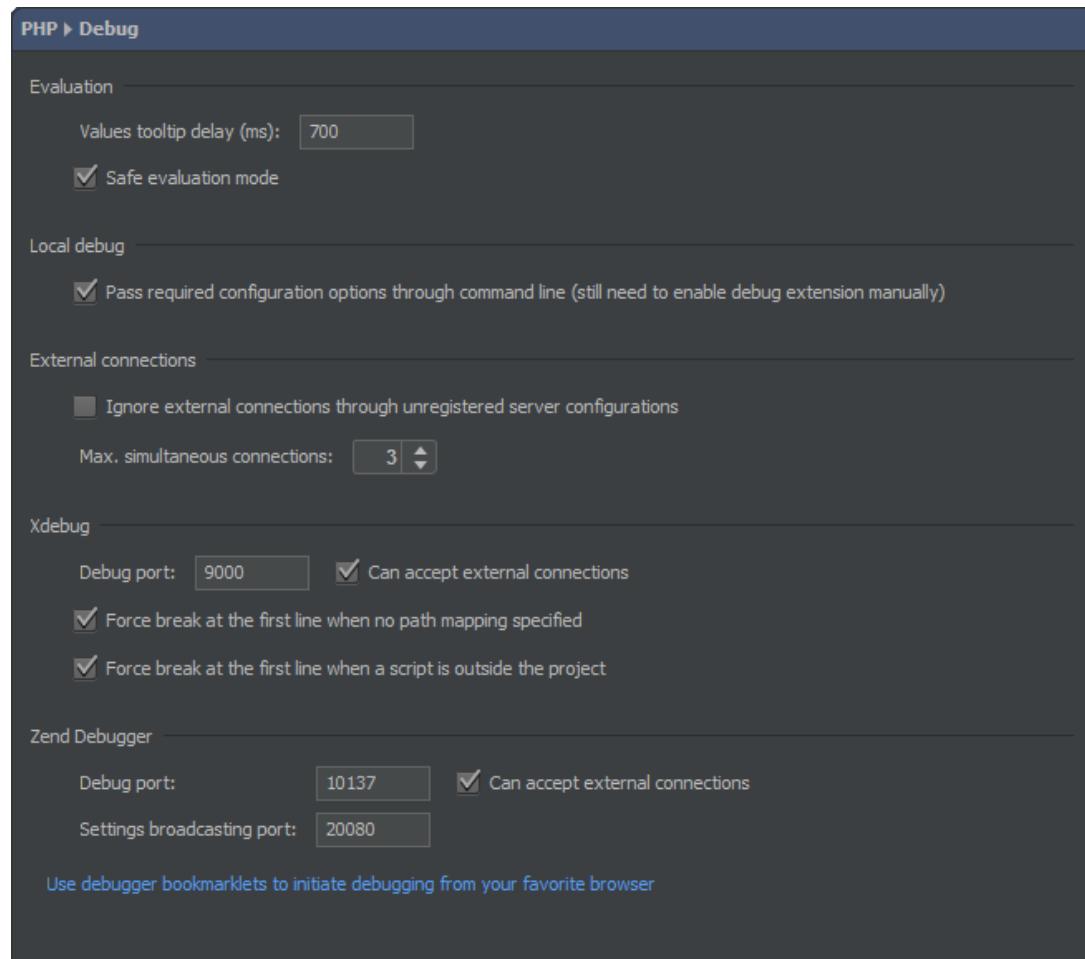
Debugger is configured through
php.ini

Project Settings | PHP | Debug

How debug info is displayed in IDE

Xdebug / Zend Debugger port number

DBGp Proxy

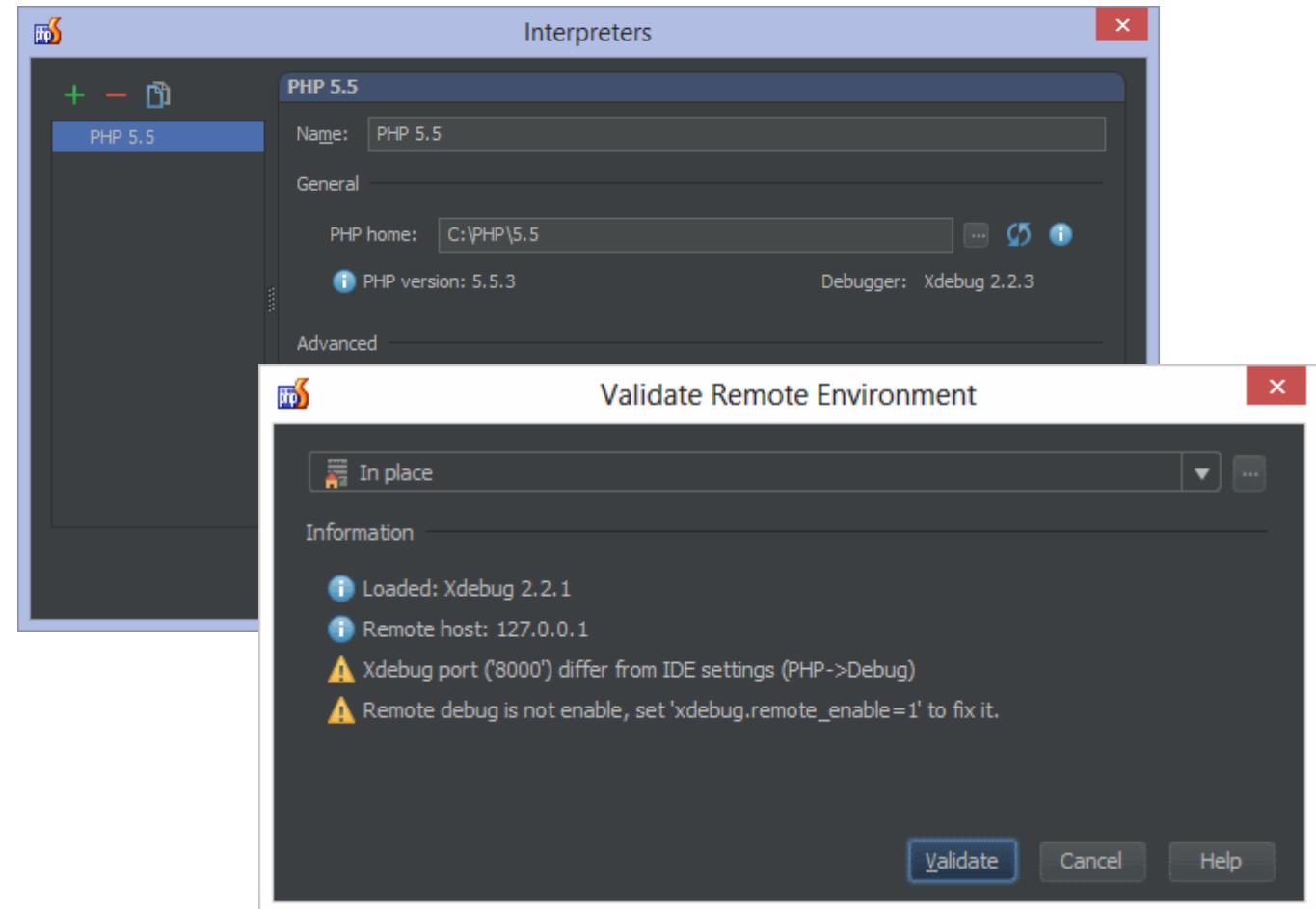


Debugger Configuration Validation

PhpStorm analyzes configuration

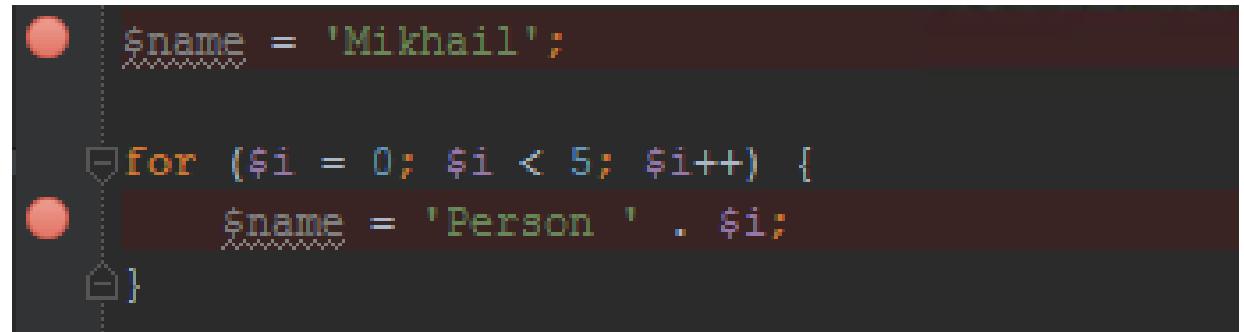
Reports active debugger

When a server is configured, reports configuration issues



Breakpoints

Tell the interpreter to pause execution and inspect variables.



A screenshot of a debugger interface showing a script with two red circular breakpoints. The script code is:

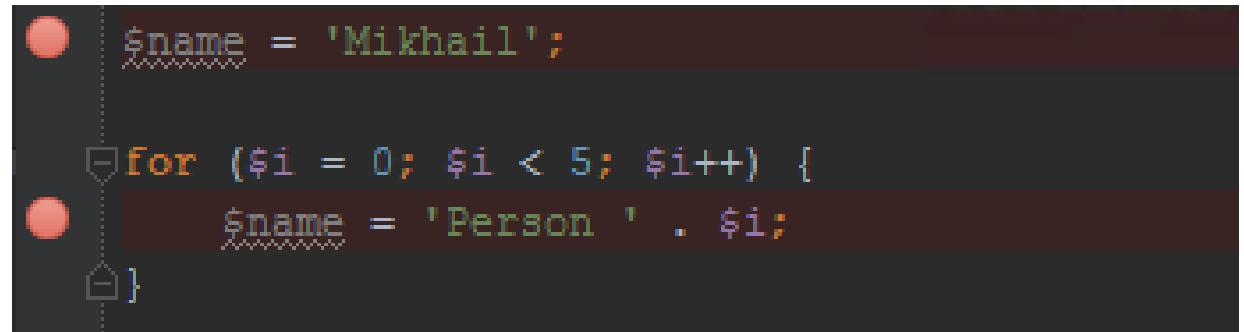
```
$name = 'Mikhail';
for ($i = 0; $i < 5; $i++) {
    $name = 'Person ' . $i;
}
```

  Ctrl+F8

 Command+F8

Conditional Breakpoints

Tell the interpreter to pause execution and inspect variables, only when a specific condition is true.



A screenshot of a debugger interface showing a code editor with PHP code. Two red circular breakpoints are set on the first two lines of code. The first line contains the assignment `$name = 'Mikhail';`. The second line contains the start of a `for` loop: `for ($i = 0; $i < 5; $i++) {`. A tooltip or status bar at the bottom indicates that the second breakpoint is a conditional breakpoint.

```
$name = 'Mikhail';
for ($i = 0; $i < 5; $i++) {
    $name = 'Person ' . $i;
}
```

Windows  Linux  Ctrl+Shift+F8

Mac  Command+Shift+F8

Breakpoints

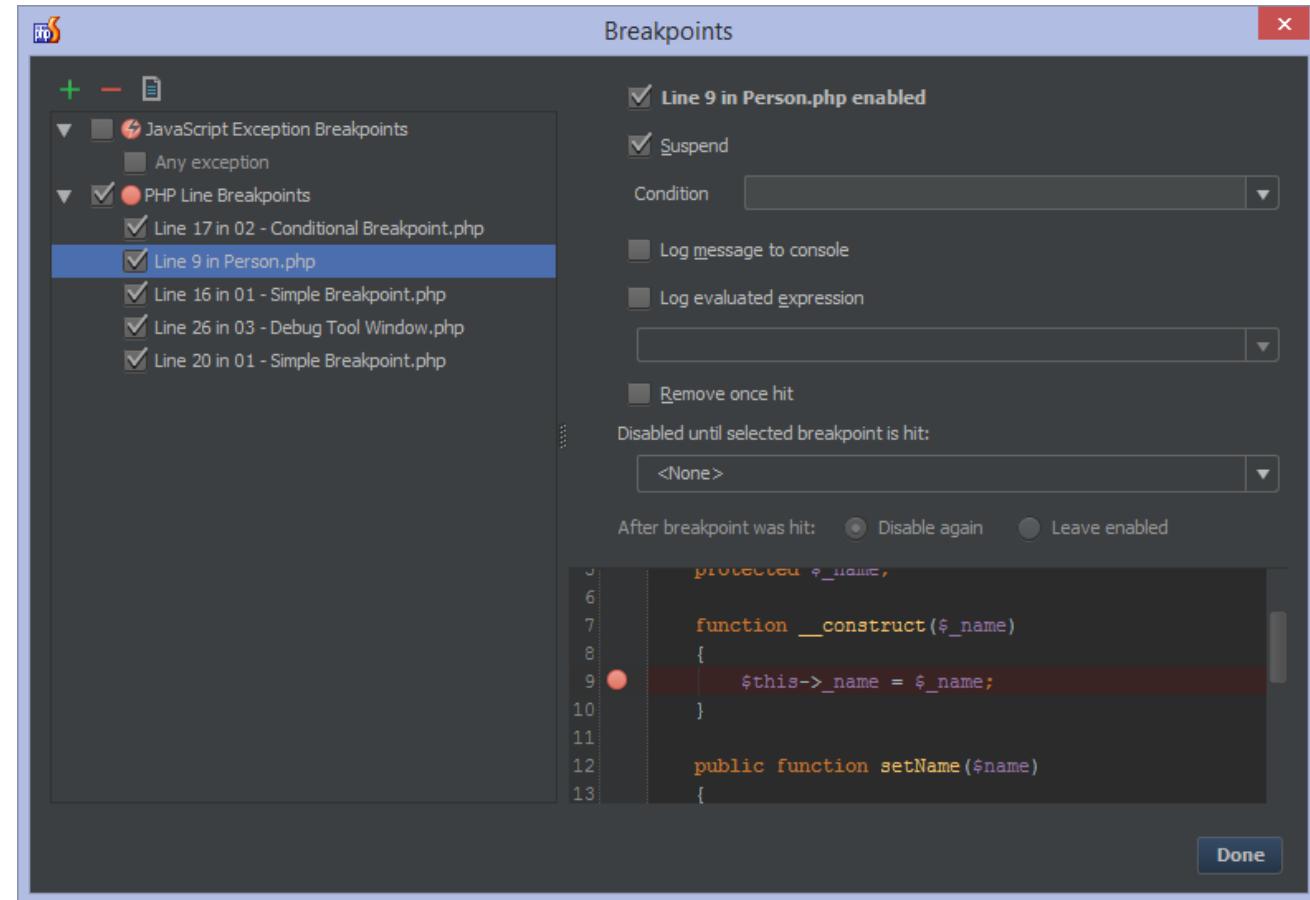
See breakpoints that are specified and configure additional options.

Log a message to console

Remove the breakpoint after it has been hit

Disable breakpoint until another breakpoint has been hit

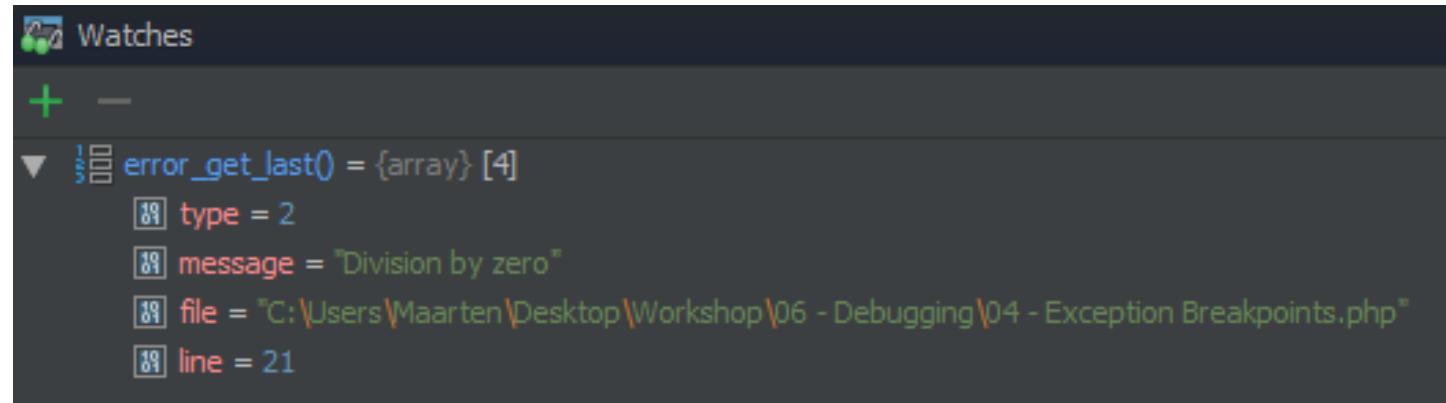
  Ctrl+Shift+F8
 Command+Shift+F8



Exception Breakpoints

Xdebug only!

Break when error, warning, notice or Exception occurs.



Windows: Ctrl+Shift+F8

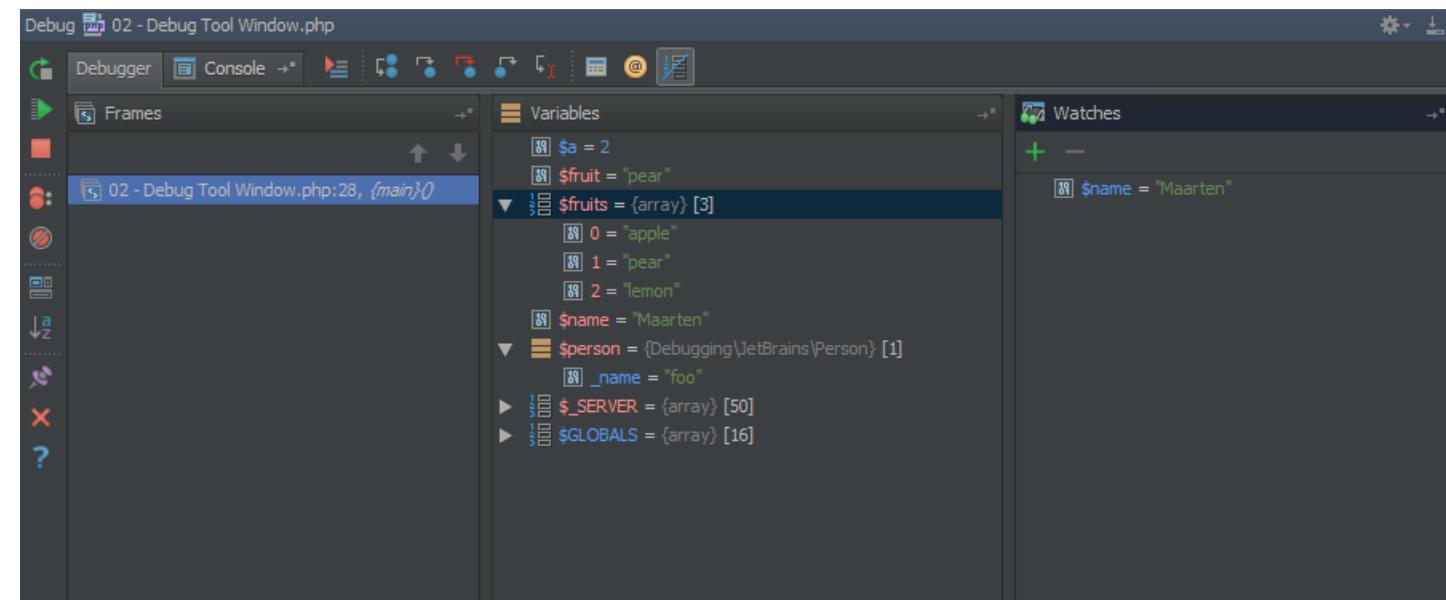
Mac: Command+Shift+F8

Debug Tool Window

Available during a debug session.

Showing execution details,
variables, watches.

Allows running code / modifying
variable value.



Windows: Alt+5, Alt+F8 evaluate expression

Mac: Command+5, Alt+F8 evaluate expression

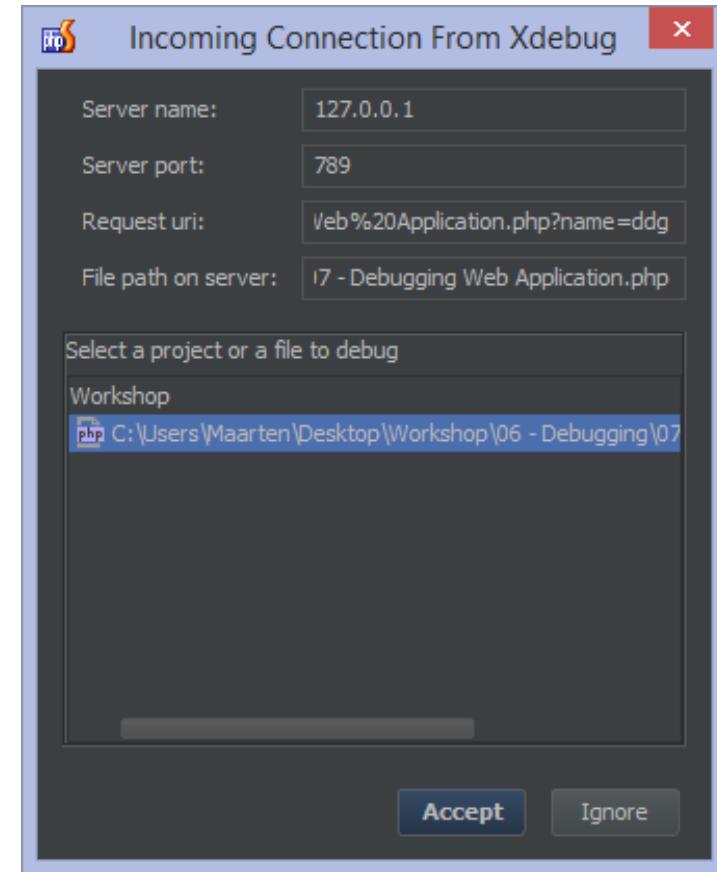
Debugging a Web Application

Use a Run Configuration

PHP HTTP Request – automate the request

PHP Web Application – start with debugging

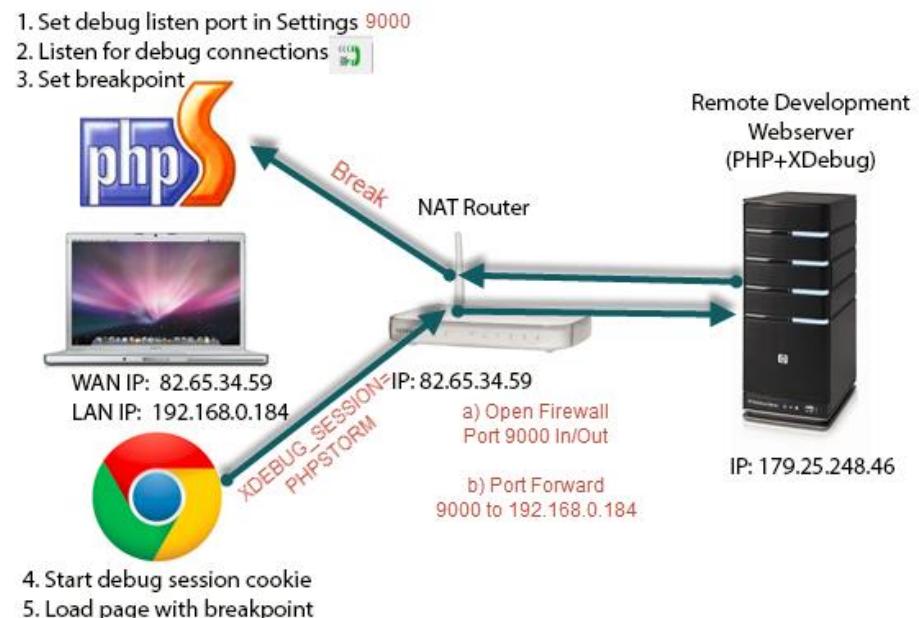
Listen for incoming connections and use bookmarklets or plugin



DBGp Proxy

Debug on remote server.

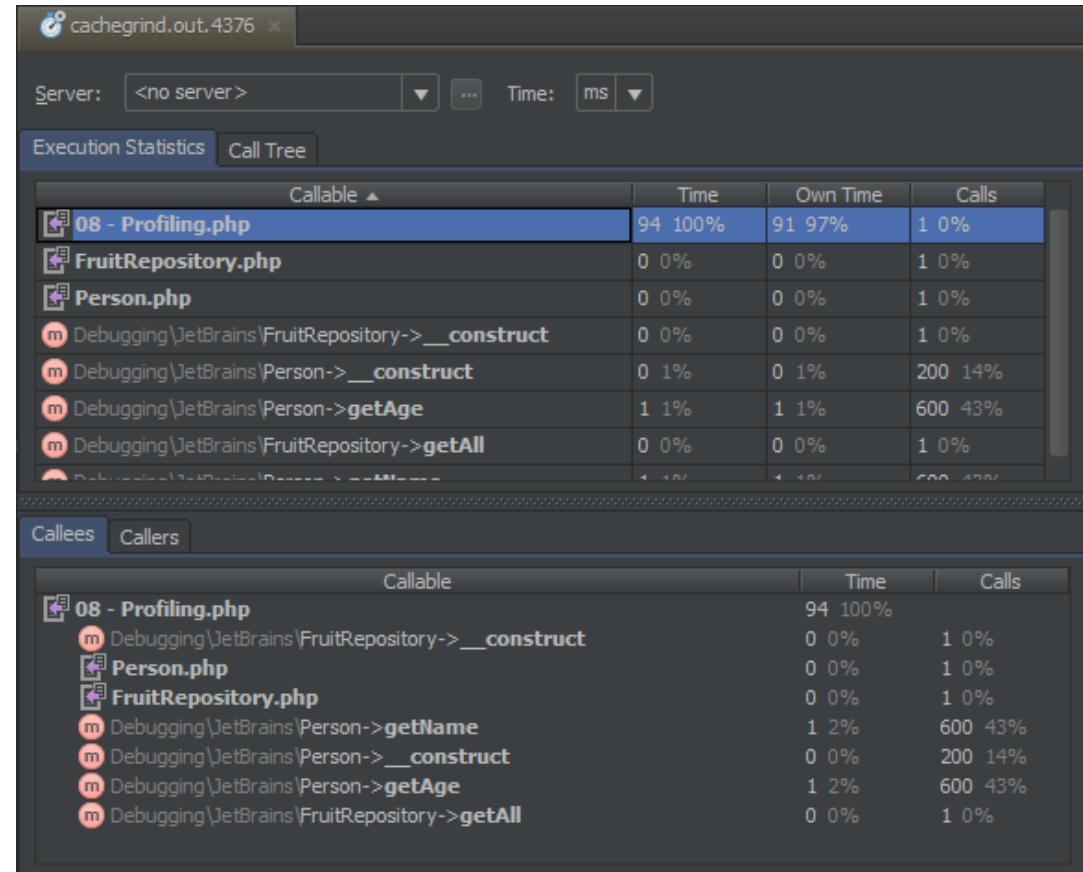
Every developer can use different IDE key and debug independent from each other.



From <http://matthardy.net/blog/configuring-phpstorm-xdebug-dbgp-proxy-settings-remote-debugging-multiple-users/>

Profiling

Insight into # of calls, execution times per function, ...



Todo explorer

Highlighting TODO

Listing tasks commented in code.

Work in every file type.

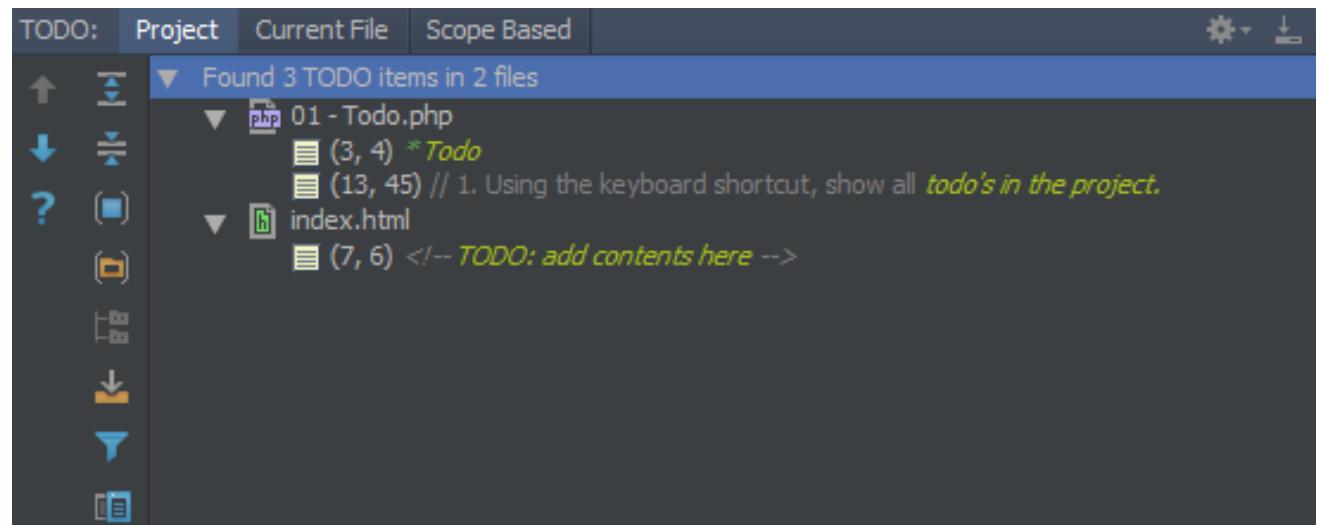
Default patterns:

// todo

// fixme

Windows icon Linux icon Alt+6

Mac icon Command+6

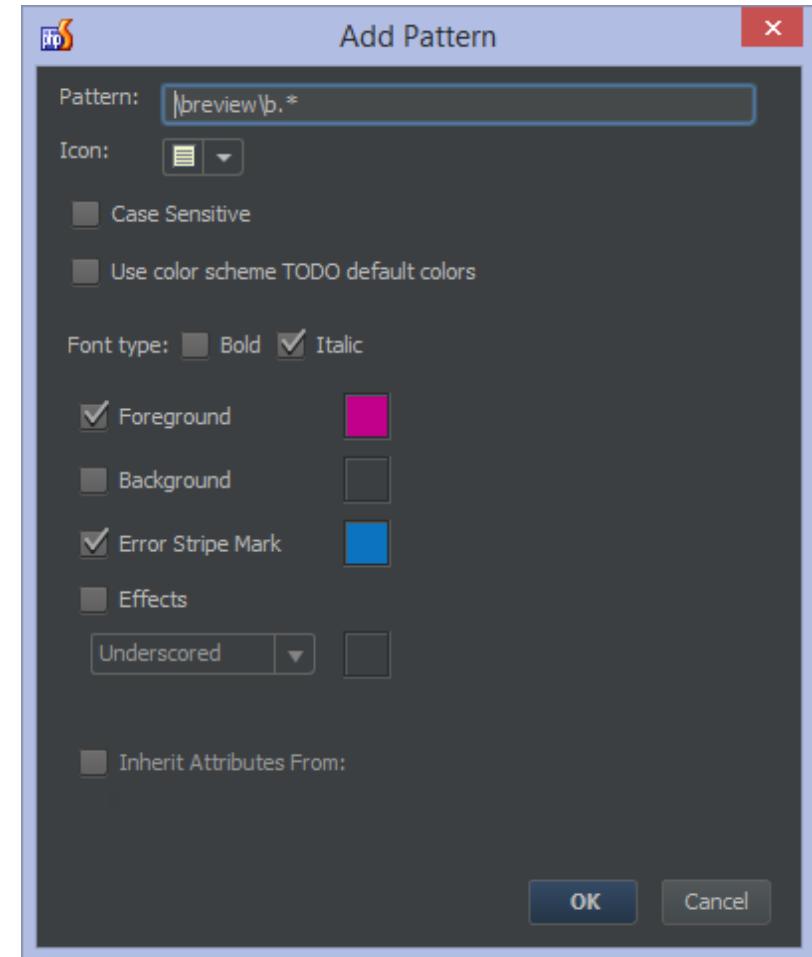


Patterns

Customize todo patterns to recognize other keywords.

Windows  Linux  Ctrl+Alt+S

Mac  Command+,

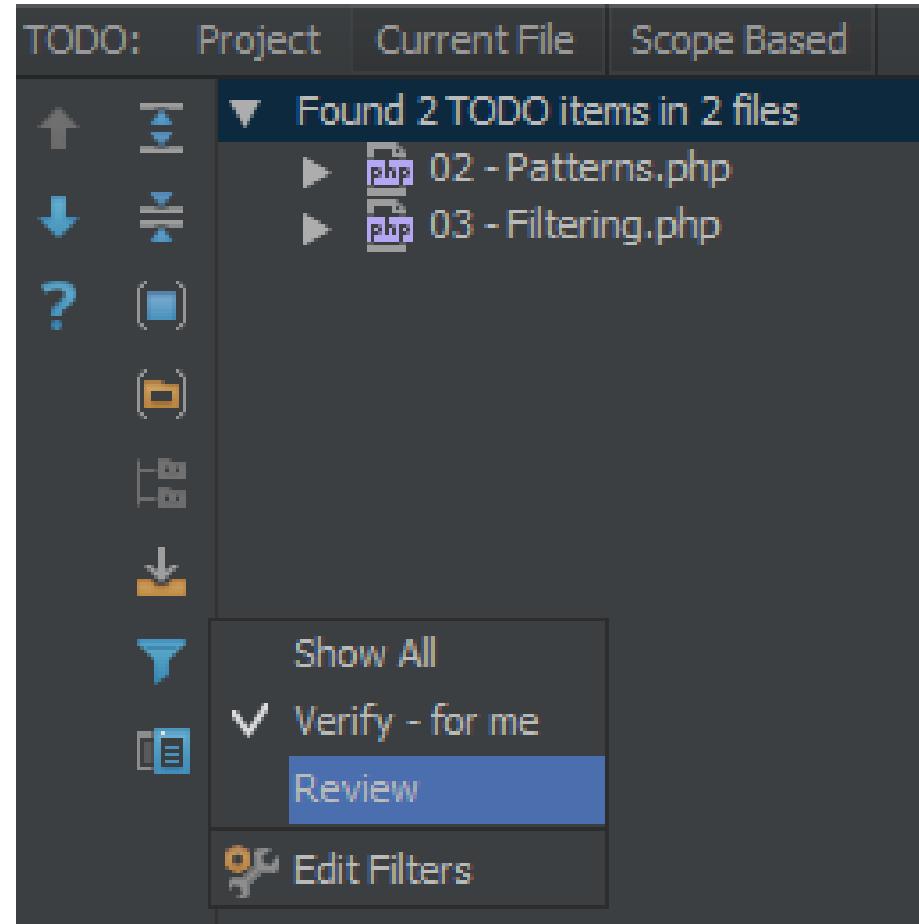


Filtering

Filter tasks in the tool window based on pattern.

Windows: Alt+6

Mac: Command+6



Unit Testing

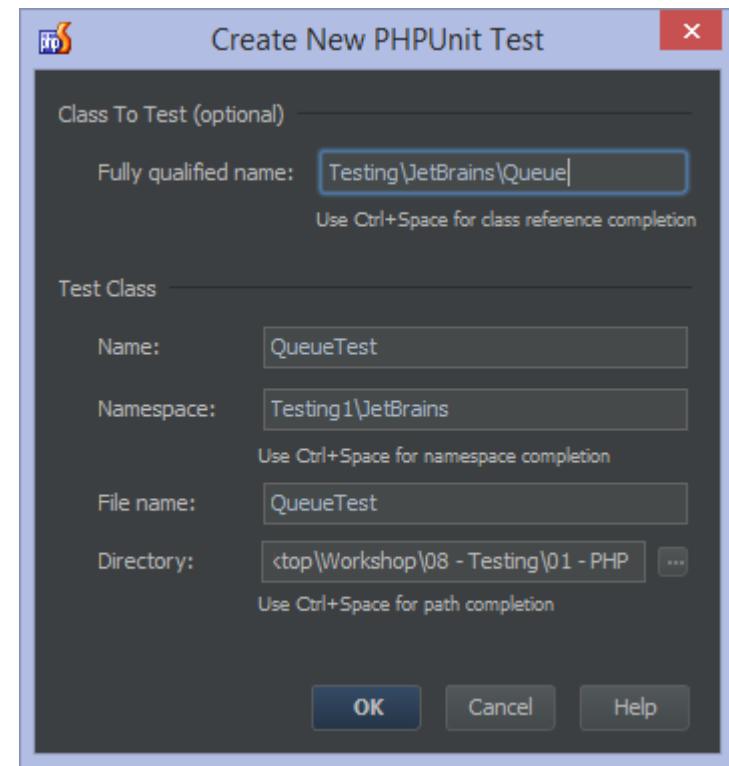
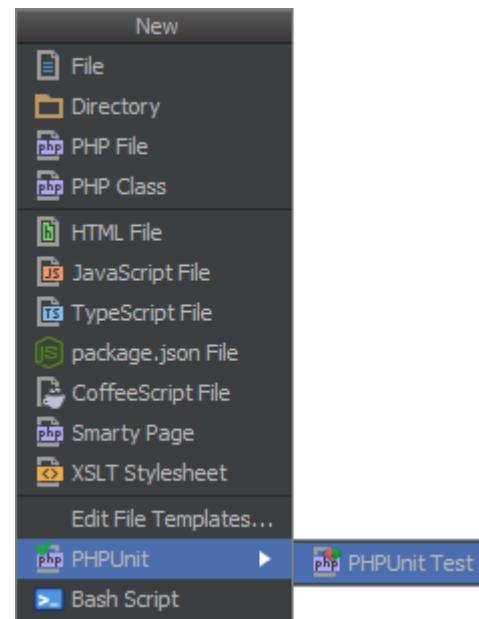
Writing PHPUnit tests

Create new file, select PHPUnit test.

Specify unit test details.

Windows icon Alt+Insert or Ctrl+Shift+T (Go to Test)

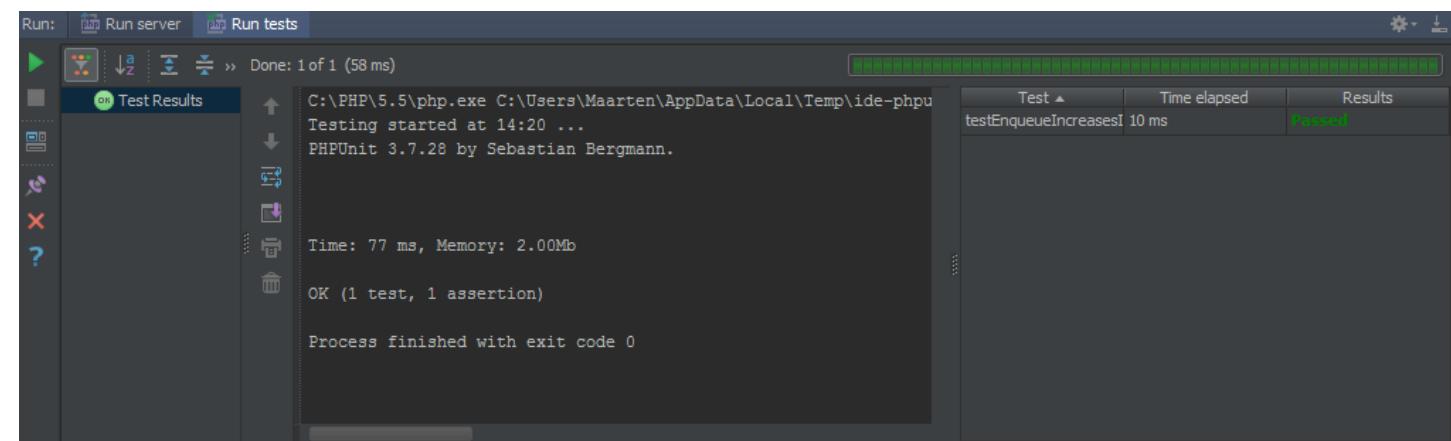
Mac icon Command+N or Command+Shift+T (Go to Test)



Running PHPUnit tests

Run Unit Tests using PHPUnit. Note that the PHPUnit framework can be acquired through PEAR or Composer.

Can be run local or on remote server
(see <http://www.jetbrains.com/phpstorm/webhelp/run-debug-configuration-phpunit-on-server.html>)



The screenshot shows the PhpStorm IDE's Test Results tool window. The window title is "Test Results". The status bar at the top says "Done: 1 of 1 (58 ms)". The main area displays the following output:

```
C:\PHP\5.5\php.exe C:\Users\Maarten\AppData\Local\Temp\ide-phpunit
Testing started at 14:20 ...
PHPUnit 3.7.28 by Sebastian Bergmann.

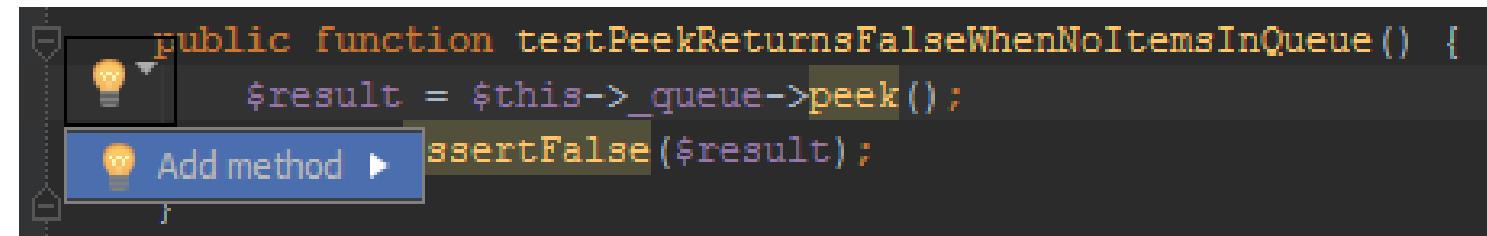
Time: 77 ms, Memory: 2.00Mb
OK (1 test, 1 assertion)

Process finished with exit code 0
```

To the right of the output, there is a table with three columns: "Test", "Time elapsed", and "Results". The single test entry is "testEnqueueIncreasesI 10 ms" with the status "Passed".

Test-Driven Development (TDD)

Writing tests first, outlining expected results, after which the method under test gets implemented.



A screenshot of an IDE interface showing a code editor with the following PHP code:

```
public function testPeekReturnsFalseWhenNoItemsInQueue() {  
    $result = $this->_queue->peek();  
    assertFalse($result);  
}
```

The cursor is positioned at the end of the line 'assertFalse(\$result);'. A code completion dropdown menu is open, showing two suggestions: 'Add method' and 'assertFalse(\$result);'. Both suggestions have a lightbulb icon next to them, indicating they are valid or recommended completions.

Windows icon Alt+Enter

Mac icon Alt+Enter

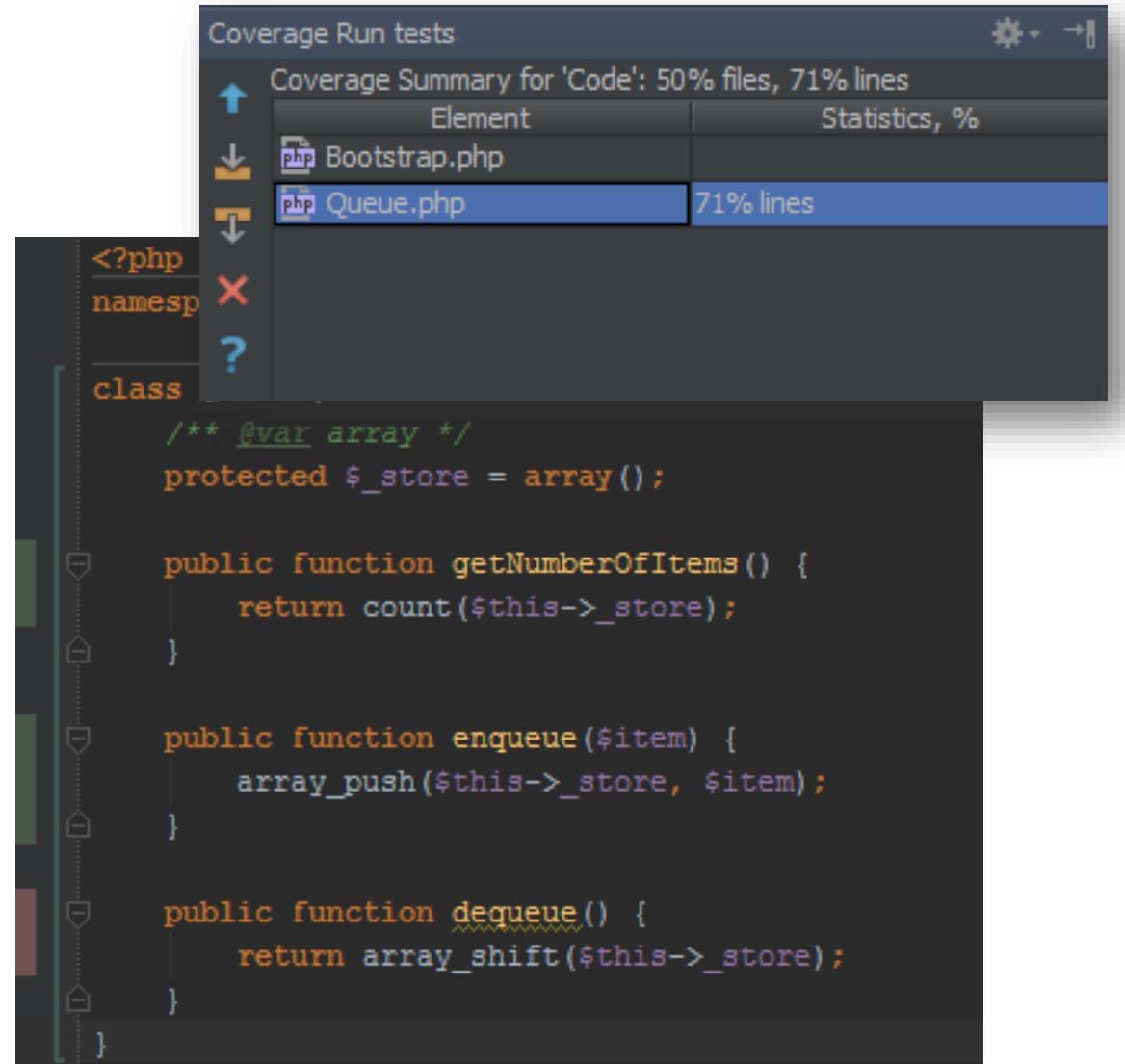
Code Coverage

See which statements have been tested and which statements have not.

Use the “Run tests with Coverage” action.

See

<https://www.jetbrains.com/phpstorm/webhelp/coverage.html>



The screenshot shows the PhpStorm Coverage Run tests tool window. At the top, it displays a summary: "Coverage Summary for 'Code': 50% files, 71% lines". Below this is a table with two rows:

Element	Statistics, %
php Bootstrap.php	
php Queue.php	71% lines

The "Queue.php" row is selected. The main area below shows the PHP code for the `Queue` class:

```
<?php  
namesp  
class  
/** @var array */  
protected $_store = array();  
  
public function getNumberOfItems() {  
    return count($this->_store);  
}  
  
public function enqueue($item) {  
    array_push($this->_store, $item);  
}  
  
public function dequeue() {  
    return array_shift($this->_store);  
}
```

On the left side of the code editor, there are vertical colored bars (green, blue, red) corresponding to the code blocks, indicating the coverage status of each section. The "enqueue" and "dequeue" methods are fully covered (green), while the constructor and the "getNumberOfItems" method are partially covered (blue).

JavaScript Unit Testing

May require a plugin to be installed,
e.g. [Karma](#)

Using Karma or JTestDriver.

Installing Karma through NPM

<https://www.jetbrains.com/phpstorm/webhelp/preparing-to-use-karma-test-runner.html>

Installing JTestDriver

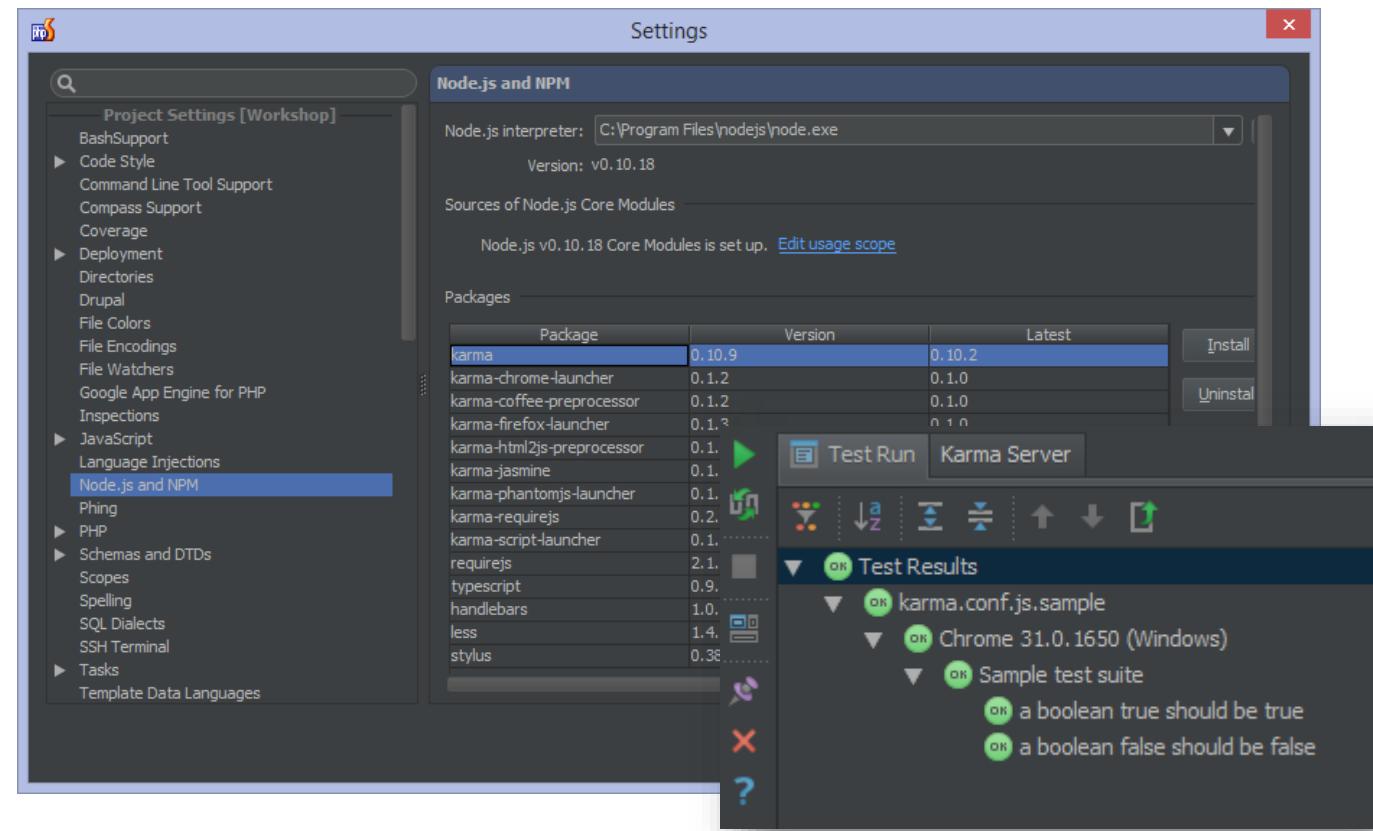
<https://www.jetbrains.com/phpstorm/webhelp/preparing-to-use-jtestdriver-test-runner.html>

Support for
JTestDriver Assertion framework

Jasmine

QUnit

Mocha



Version Control

Version Control

Keep a log of all changes

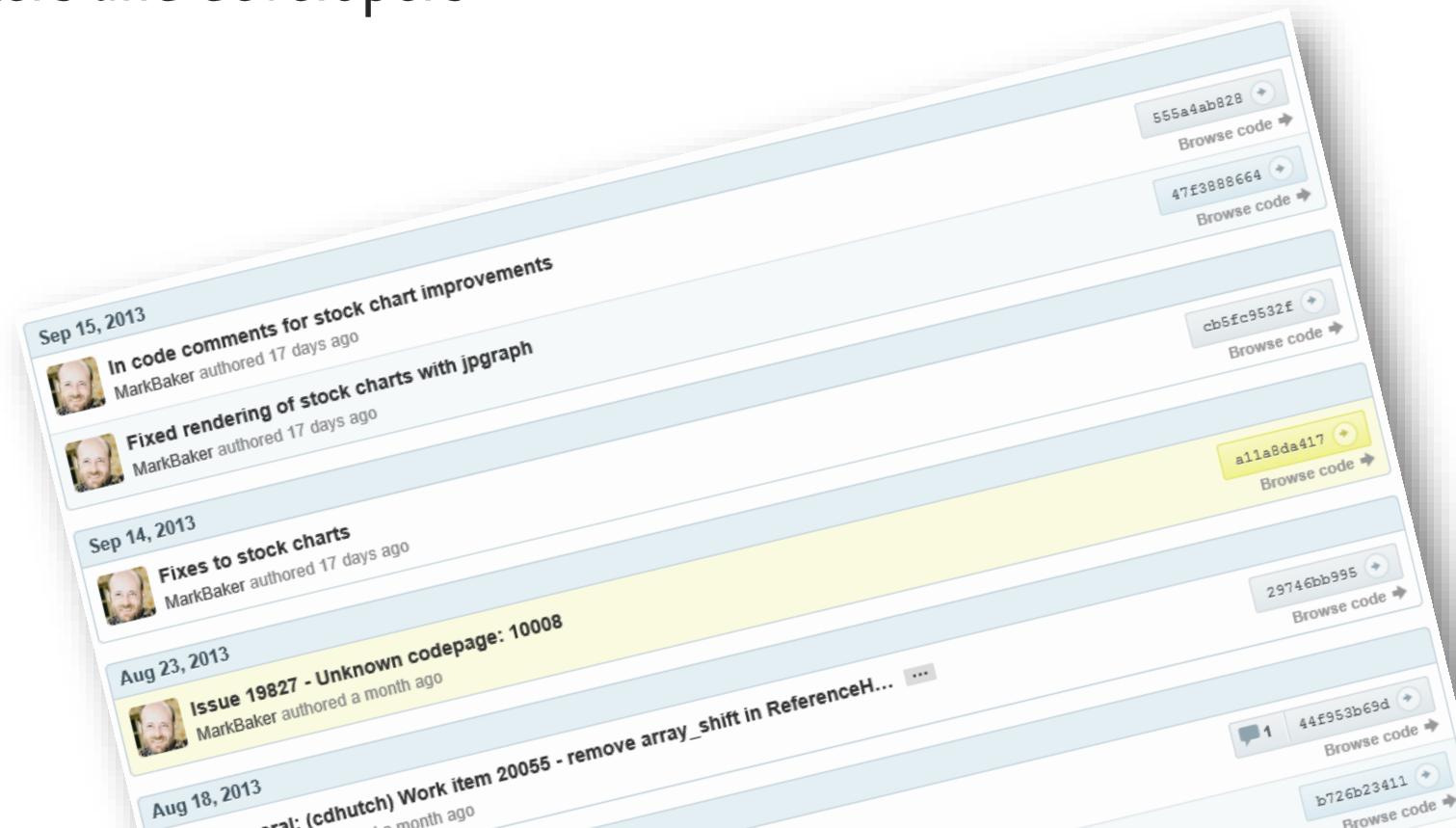
Synchronize copies across computers and developers

Go back in time

Who did what when why?

Compare versions

Rollback versions



Concepts

Repository (local / remote)

Checking out / cloning

Fetch copy from repository

Commit

Submit changes to repository

Revision / changeset

What changed? When? Who? Why?

Stored in order

Conflicts

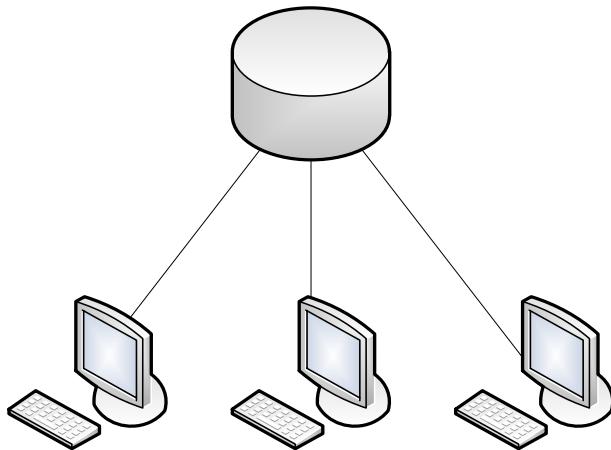
Diff

Branching / merging



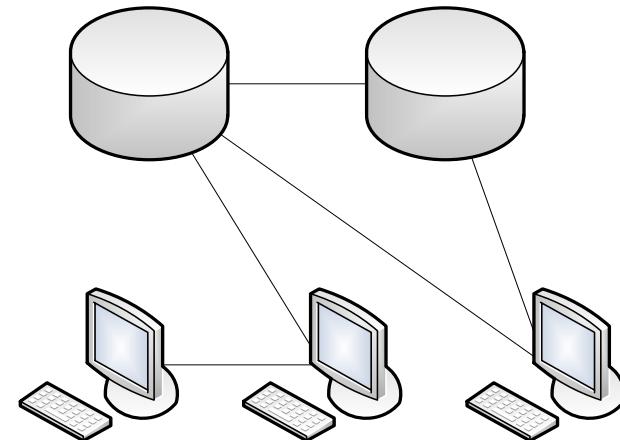
Flavours

CENTRALIZED VERSION CONTROL



Central repository = entire history
Devs work on a version and sync

DECENTRALIZED VERSION CONTROL



Devs have copy of entire history
Work on a version and sync with any

Products all use different terms...

Checkout / Clone / Export

Commit / Check In

Push / Submit

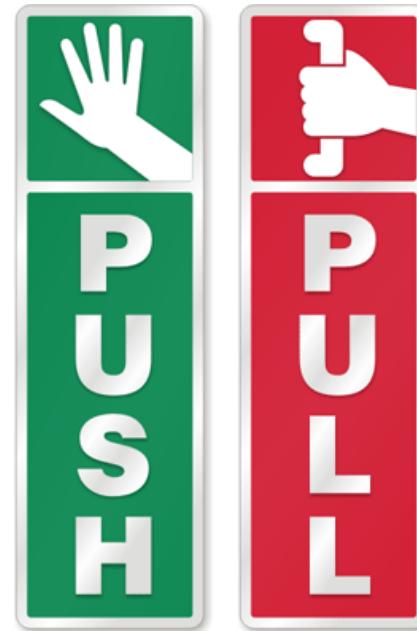
Revert / Roll back

History / Log

Annotate / Blame

Compare / Diff

...



But they are similar in nature!

Unified VCS functionality in the IDE

CVS

Git

Subversion

Mercurial

Perforce

TFS (via plugin)

(others through plugins)

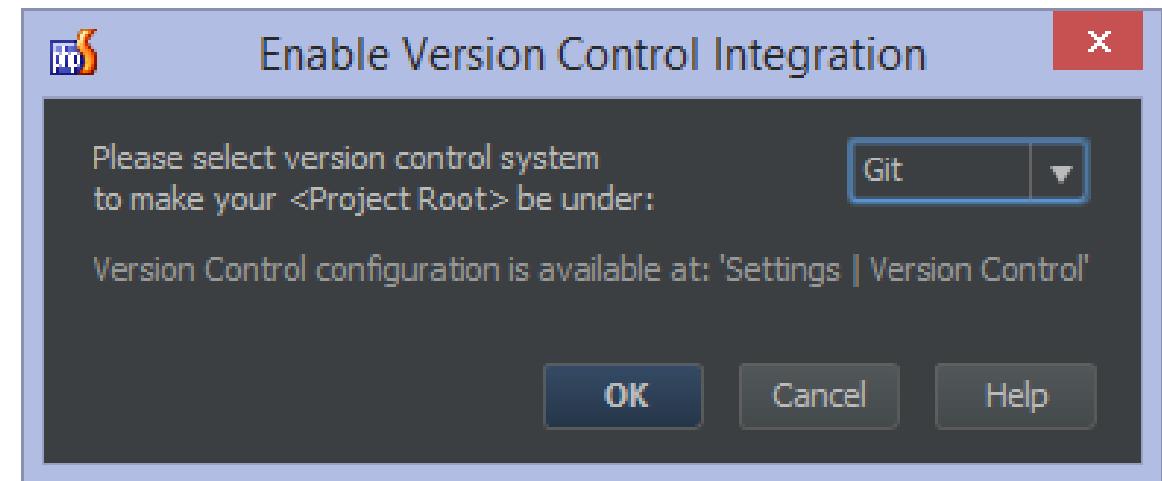
IDE enriched with specific functionality (e.g. push/pull)



Enabling VCS integration

Telling the IDE that the project is under a VCS system.

Use the VCS menu.

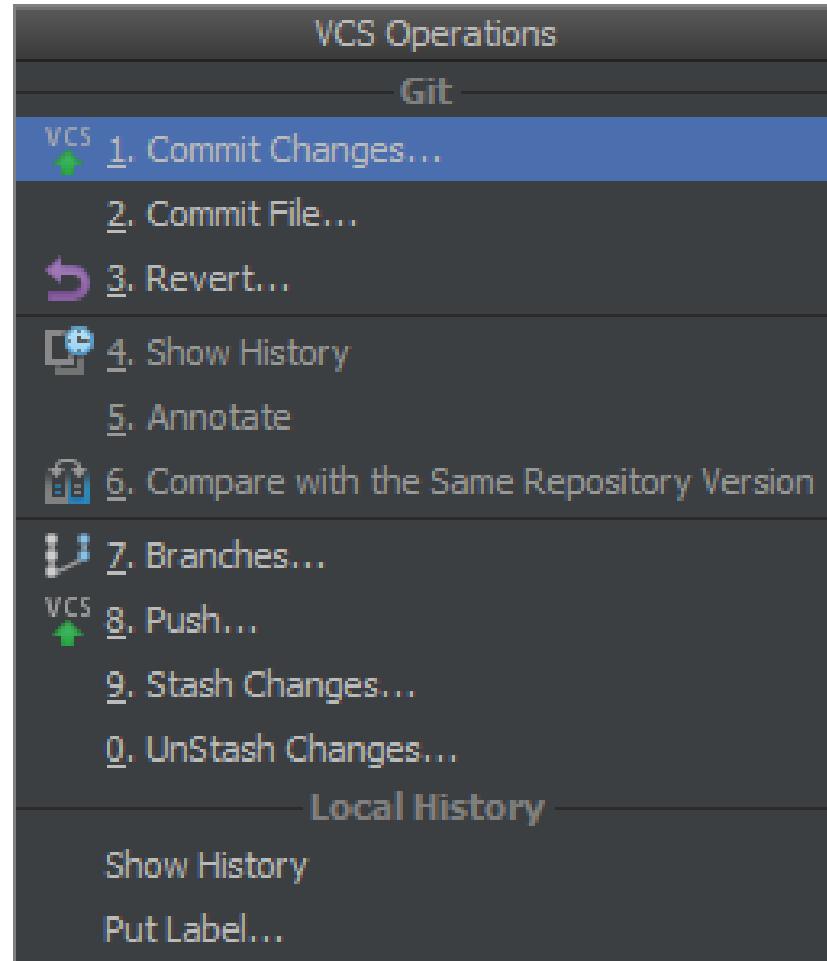


VCS operations

Perform operations on the version control system such as adding files, deleting files, committing changes and so on.

Windows icon Alt+Backquote (Alt+`)

Linux icon Ctrl+V

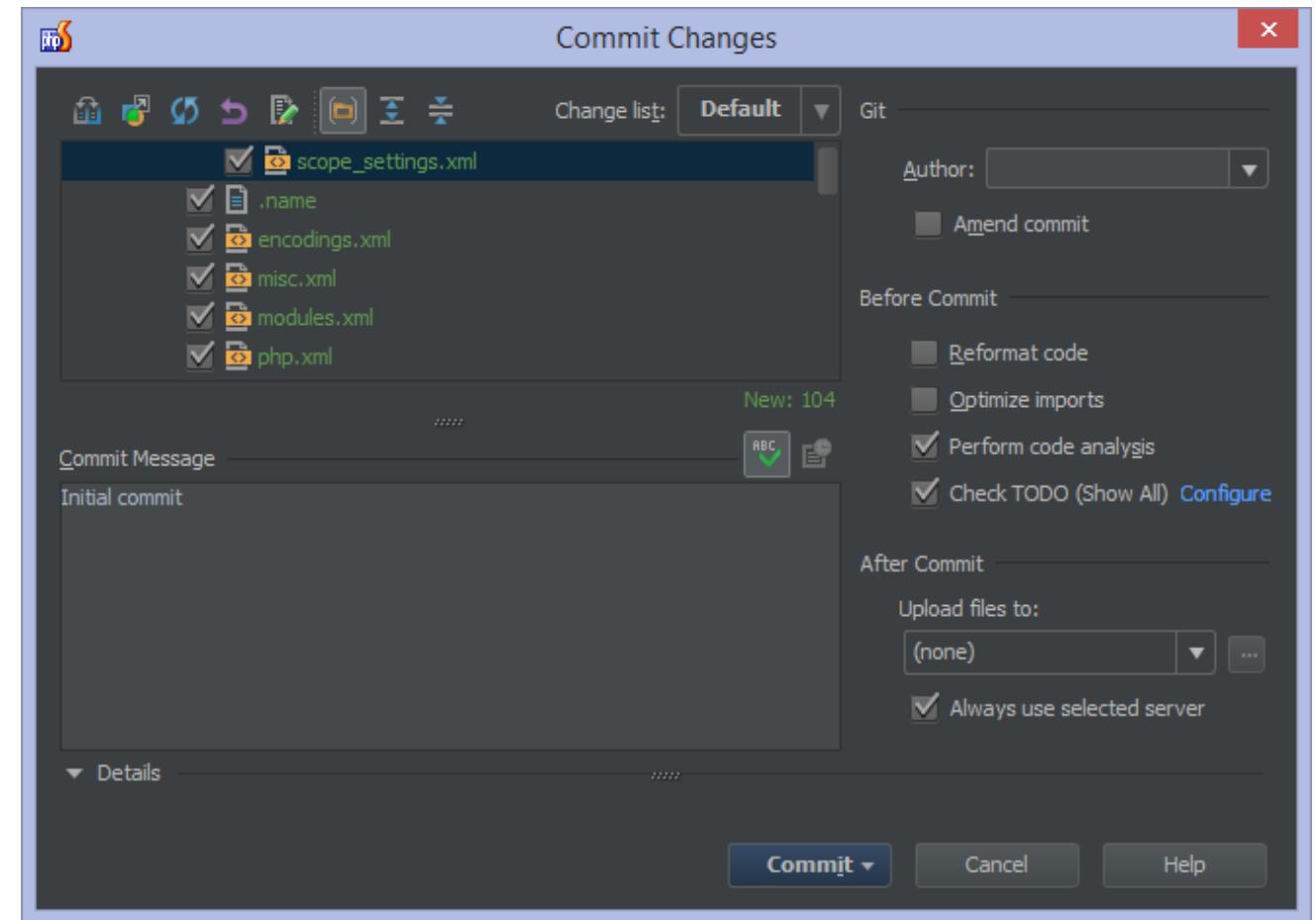


Committing Changes

Persisting the current changeset as a logical operation in the VCS, with a comment, author, ...

Windows icon Ctrl+K

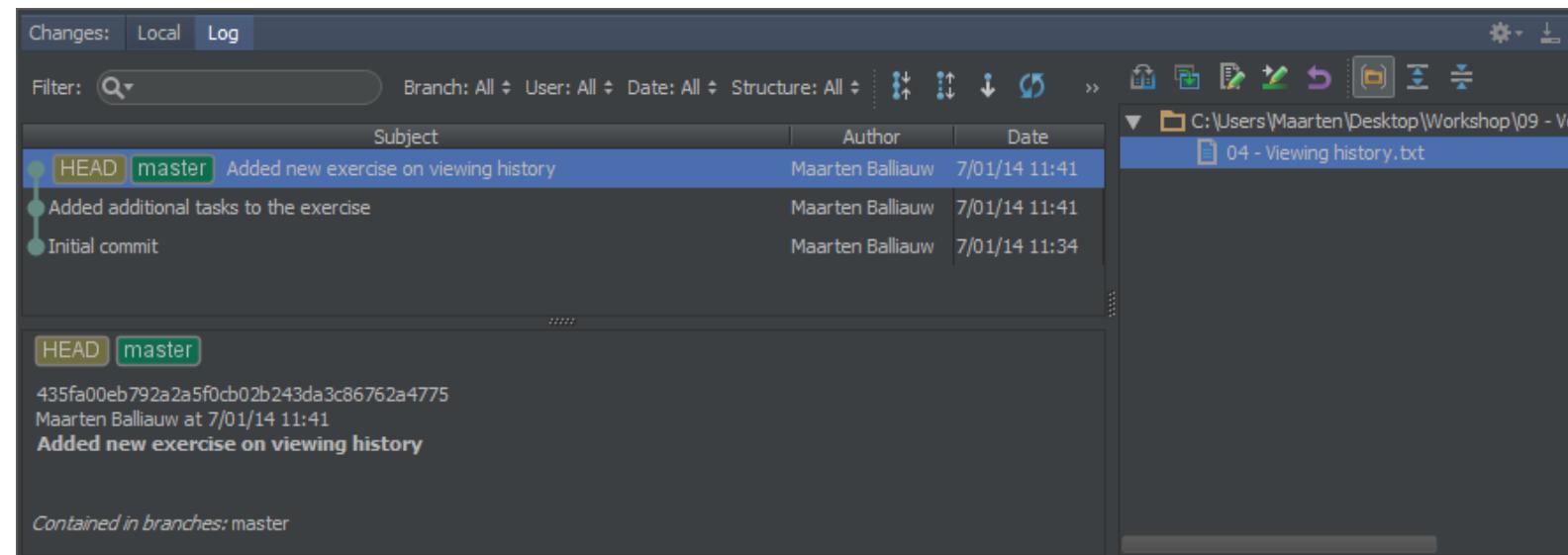
Mac icon Command+K



JetBRAINS

Viewing History

View all information related to previous commits.



Windows icon Alt+9

Mac icon Command+9

Viewing Diff

Show changes made to a particular file.

Windows icon Linux icon Ctrl+D

Mac icon Command+D

The screenshot shows a code editor window comparing two versions of a file. The left pane displays the original file content, and the right pane displays the modified file content. The interface includes line numbers and a toolbar with various icons. The status bar at the bottom shows '1 difference' and indicates the types of changes: Deleted, Changed, and Inserted.

C:\Users\Maarten\Desktop\Workshop\09 - Version Control\03 - Committing Changes.txt

18b67a2f4e3673e8b2e1e00039fe11ddc9090249 (Read-only)

c59a08aaa6bfc90ace6f05dfcc2a3caae96c0997 (Read-only)

Committing Changes

Persisting the current changeset as a logical operation

Ctrl+K (Windows/Linux)
Command+K (Mac OS X)

1. Commit the current changeset to the Git repository
2. The list on top shows changed, added and deleted files.
Set the commit message to something describing the changes.
In this case this can be "Initial commit" but for a more detailed commit message, you can add a number and a short description.
Note that the IDE can automatically reformat code, so we don't need to worry about that.
We can also upload changes to a remote server once they are committed.
3. Commit the change.

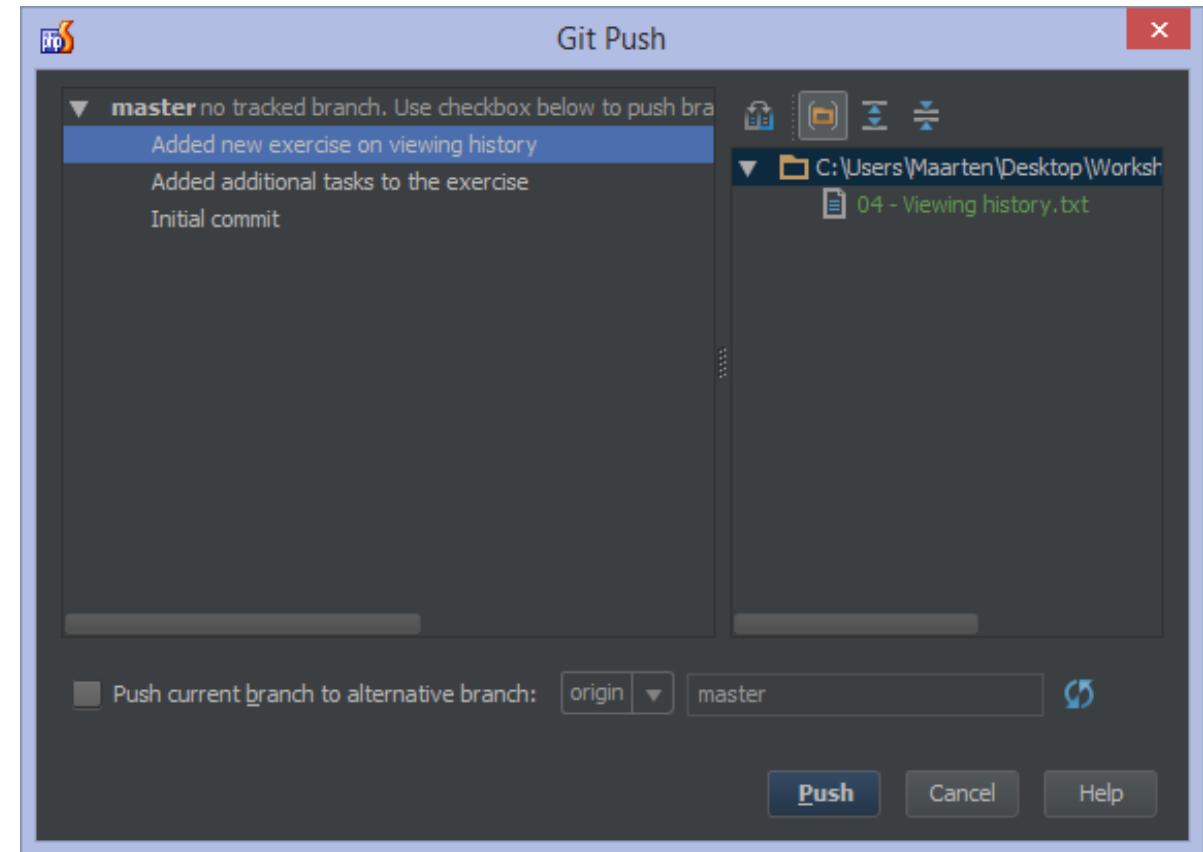
1. Commit the current changeset to the Git repository
2. The list on top shows changed, added and deleted files.
Set the commit message to something describing the changes.
In this case this can be "Initial commit" but for a more detailed commit message, you can add a number and a short description.
Note that the IDE can automatically reformat code, so we don't need to worry about that.
We can also upload changes to a remote server once they are committed.
3. Commit the change.
4. The Project Tool Window will now show files in the changeset.
5. Start typing in the current file (enter your name) in the left gutter, denoting a change in the line.
In the Project Tool Window, the file color also changes.
6. Add a line of text below. Note that the editor automatically formats it.
7. Commit all changes again and give the commit a descriptive message.

1 difference

Deleted Changed Inserted

Git/Mercurial: Push/Pull

Persist local copy of VCS repository to remote copy.



Changelists

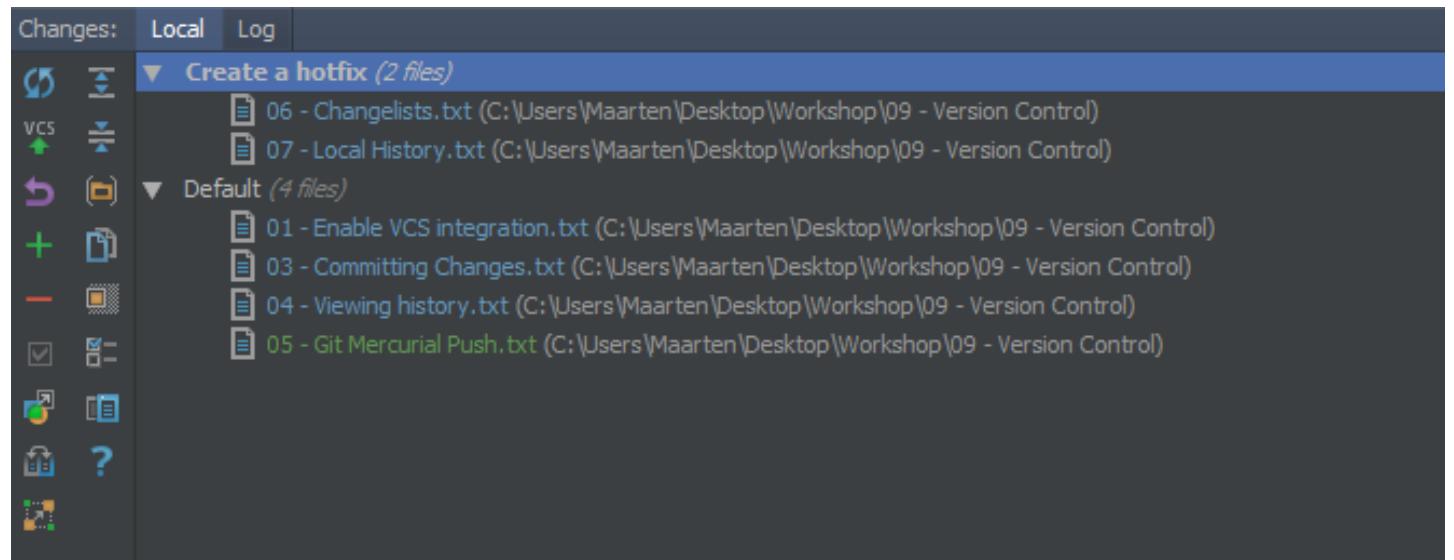
Work on multiple logical changes at the same time.

Name changelist.

Can be committed / shelved.

Windows icon Alt+9

Mac icon Command+9



Local history

Undo on steroids. Display changes made between commits, revert local changes, ...

The screenshot shows the Local History feature in an IDE. On the left, a sidebar lists file changes over the last 12 hours, including commits, deletions, and renamings. On the right, a diff viewer compares the current state of a file with its previous version. The code is PHP, demonstrating changes like variable renaming and modification. The IDE interface includes toolbars, a status bar at the bottom, and a project navigation pane on the right.

```
<?php
/**
 * Local history
 *
 * Undo on steroids. Display changes made between commits, revert local changes, ...
 */

namespace VersionControl\JetBrains;

// 1. Change the value of the variable
$name = 'Mikhail';

// 2. Make sure the $name variable has
// Code: $name = ucfirst($name);
// Save the file.

// 3. From the VCS Operations popup, S
// This opens a new window in which
// We can see we have changed a lin
// 3. From the V
// This opens
// We can see
```

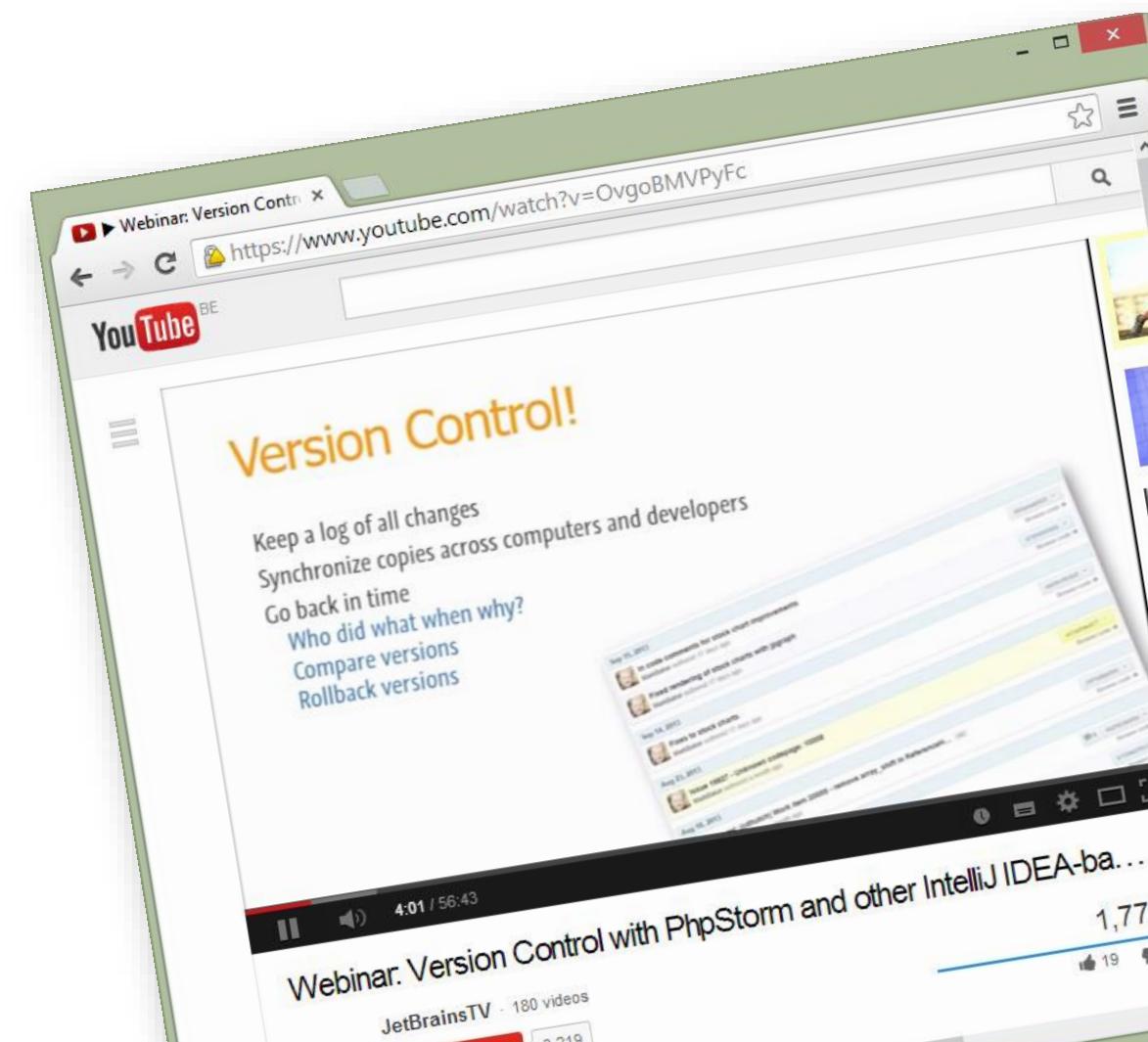
Resources

Webinar recording

<https://www.youtube.com/watch?v=OvgobMVPyFc>

Web help

<http://www.jetbrains.com/phpstorm/webhelp/version-control-with-phpstorm-2.html>



Databases

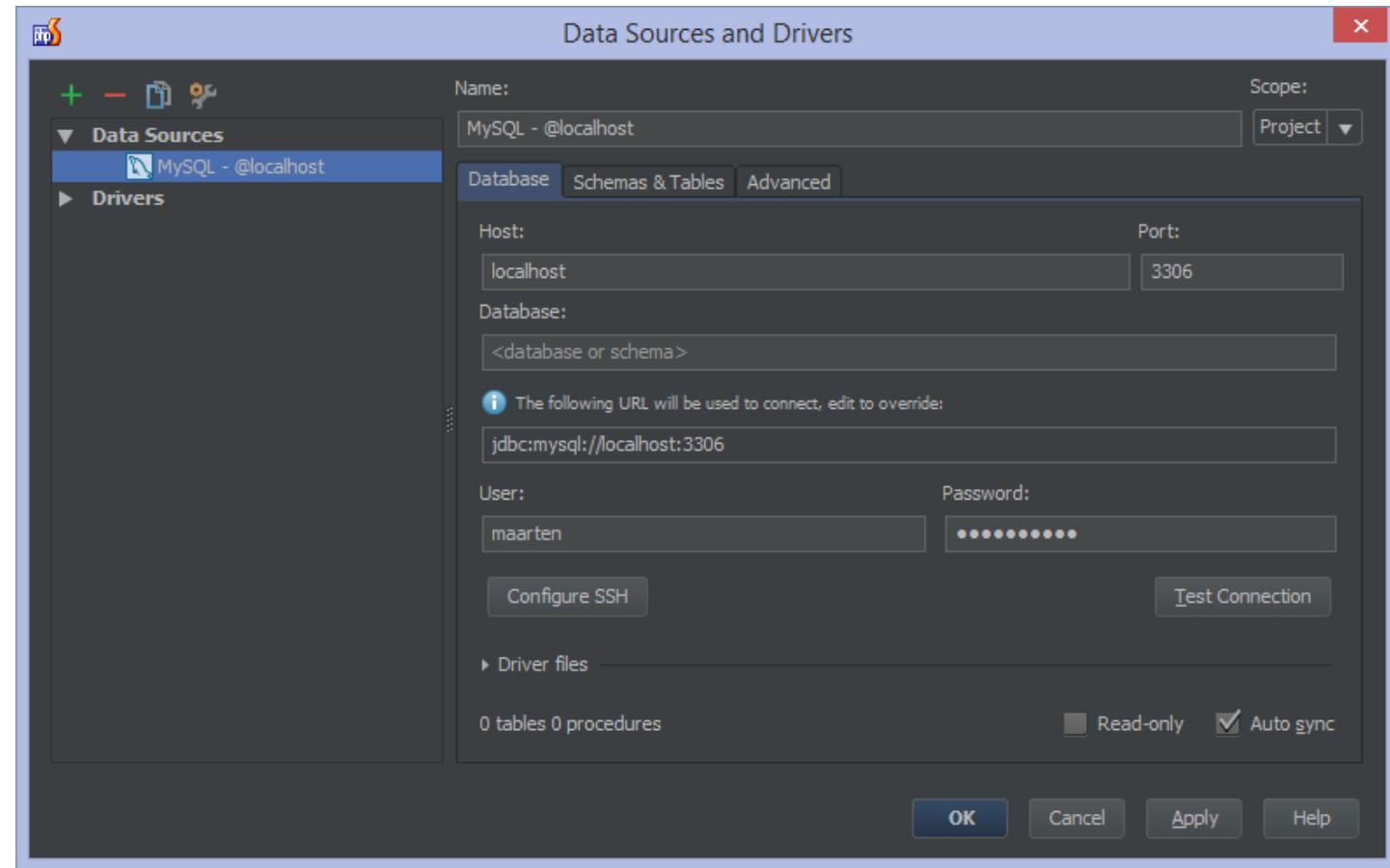
Connecting to a database (server)

Connecting to MySQL, DB2, Derby,
SQL Server, Oracle, PostgreSQL,
Sybase, H2, sqlite, Google, ...

Specify connection details

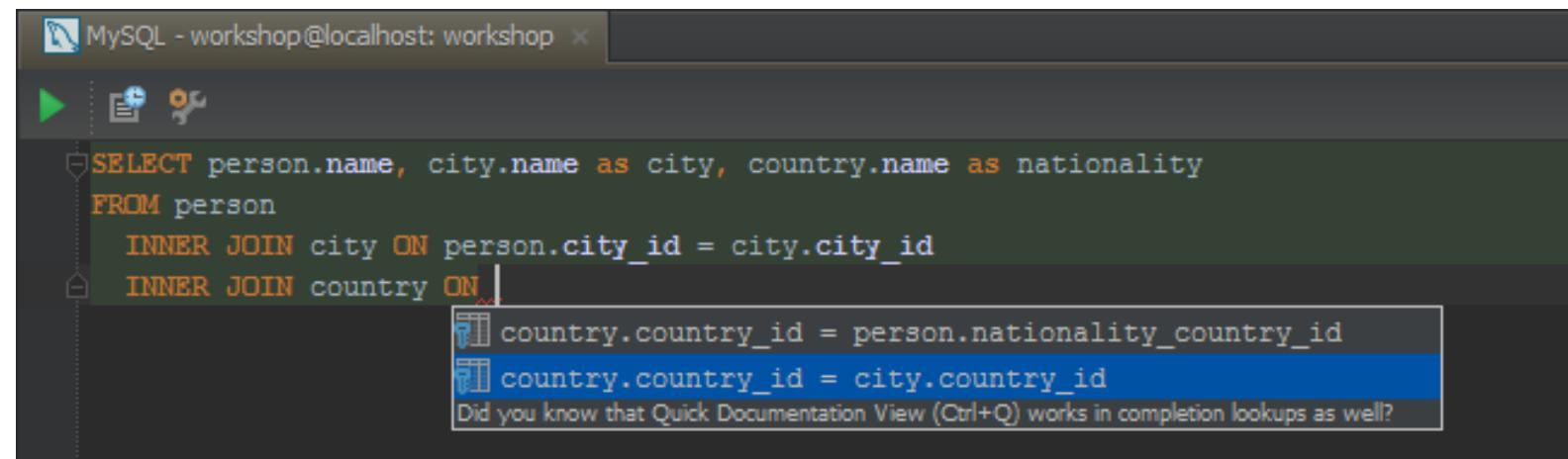
Download JDBC driver

Select schema / tables to work with



Database Console

SQL console with autocompletion,
inspections, error highlighting,
autocompletion on joins, ...



The screenshot shows a MySQL database console window titled "MySQL - workshop@localhost: workshop". The query being typed is:

```
SELECT person.name, city.name as city, country.name as nationality  
FROM person  
INNER JOIN city ON person.city_id = city.city_id  
INNER JOIN country ON |
```

A tooltip is displayed at the cursor position, listing two possible join conditions:

- country.country_id = person.nationality_country_id
- country.country_id = city.country_id

At the bottom of the tooltip, there is a note: "Did you know that Quick Documentation View (Ctrl+Q) works in completion lookups as well?"

Windows icon Ctrl+Shift+F10

Mac OS X icon Command+Shift+F10

Table Editor

Edit records in a table: insert,
update, delete, ...

The screenshot shows a database table editor for the 'workshop.person' table. The table has four columns: person_id, name, city_id, and nationality_country_id. The current view is filtered by city_id > 1. The table contains two rows:

	person_id	name	city_id	nationality_country_id
1	2	Mikhail	2	2
2	3	Hadi	4	4

Below the table, a query window displays the original query used to fetch the data:

```
'workshop.person' Table: Original Query
SELECT `workshop`.`person`.* FROM `workshop`.`person` WHERE city_id > 1
```

Windows F4

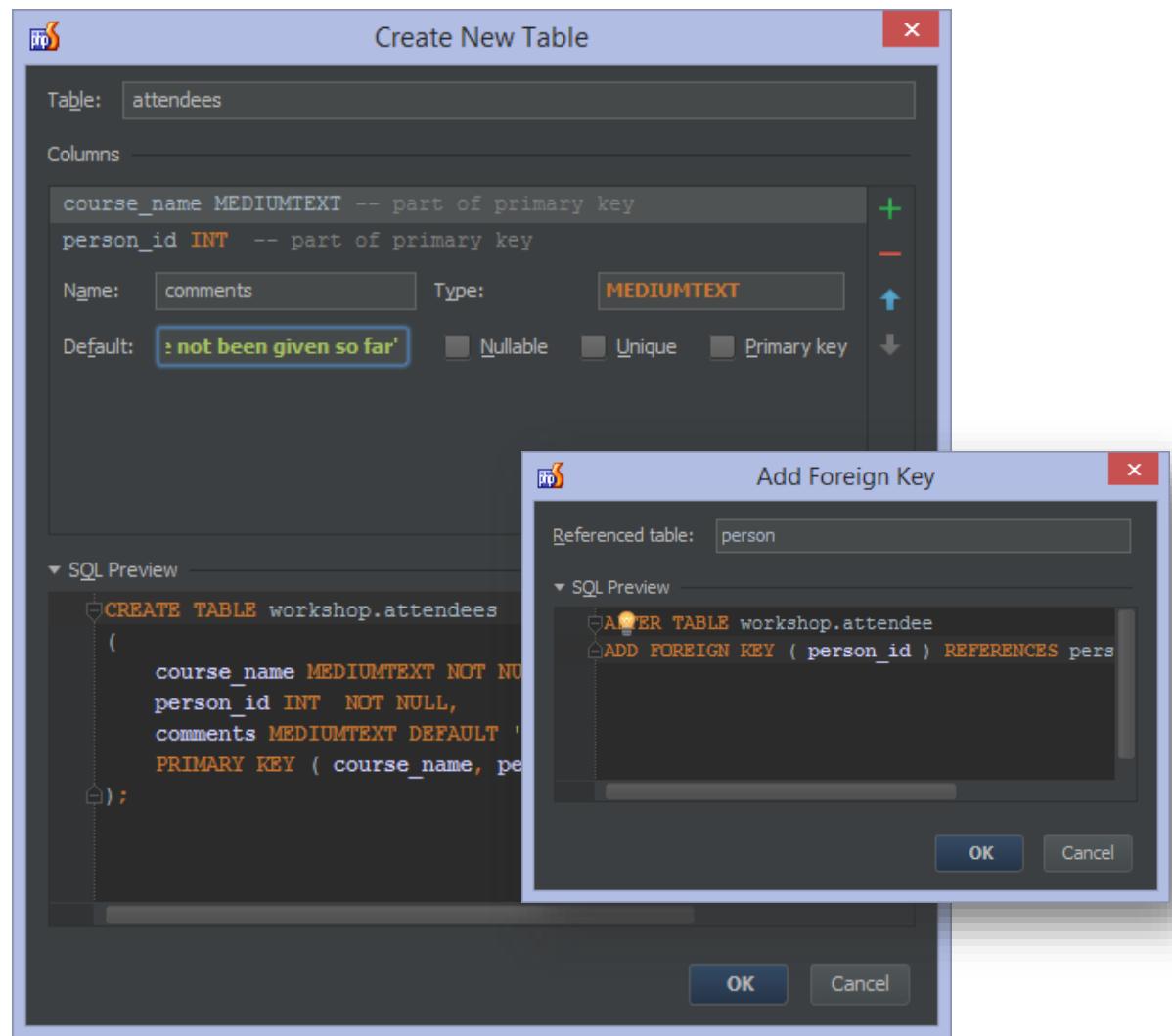
Mac F4

Create New Table

Create a new table in the database.
Specify column options, default
values, primary key, ...

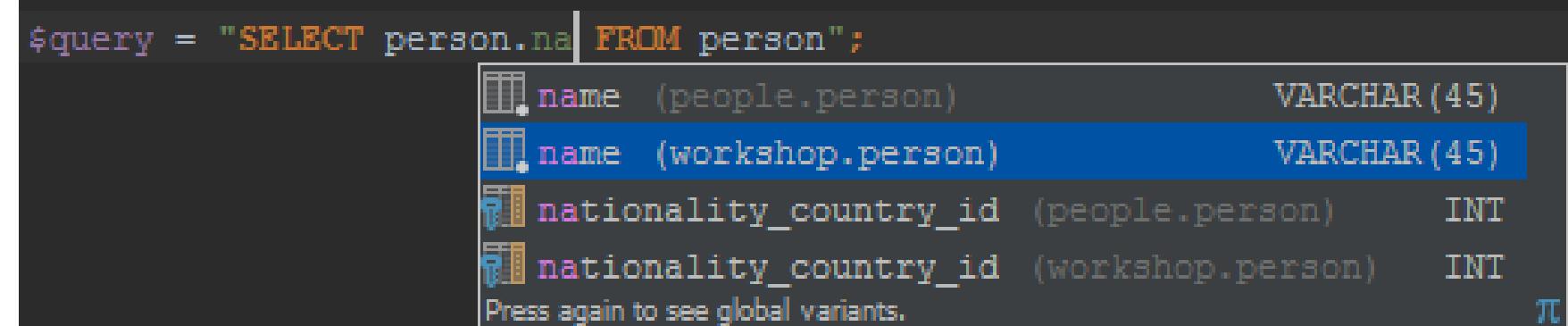
  Alt+Insert

 Command+N



Database tools in PHP

Strings can be identified as a query,
will provide autocompletion,
highlighting, ability to go to query
editor, run query right from PHP
file, ...



A screenshot of an IDE interface showing a code editor and a database completion dropdown. The code in the editor is:

```
$query = "SELECT person.name FROM person";
```

The database completion dropdown shows the following results:

Table	Column	Type
people.person	name	VARCHAR (45)
workshop.person	name	VARCHAR (45)
people.person	nationality_country_id	INT
workshop.person	nationality_country_id	INT

At the bottom of the dropdown, there is a message: "Press again to see global variants."

Windows: Ctrl+Space / Ctrl+Shift+Space

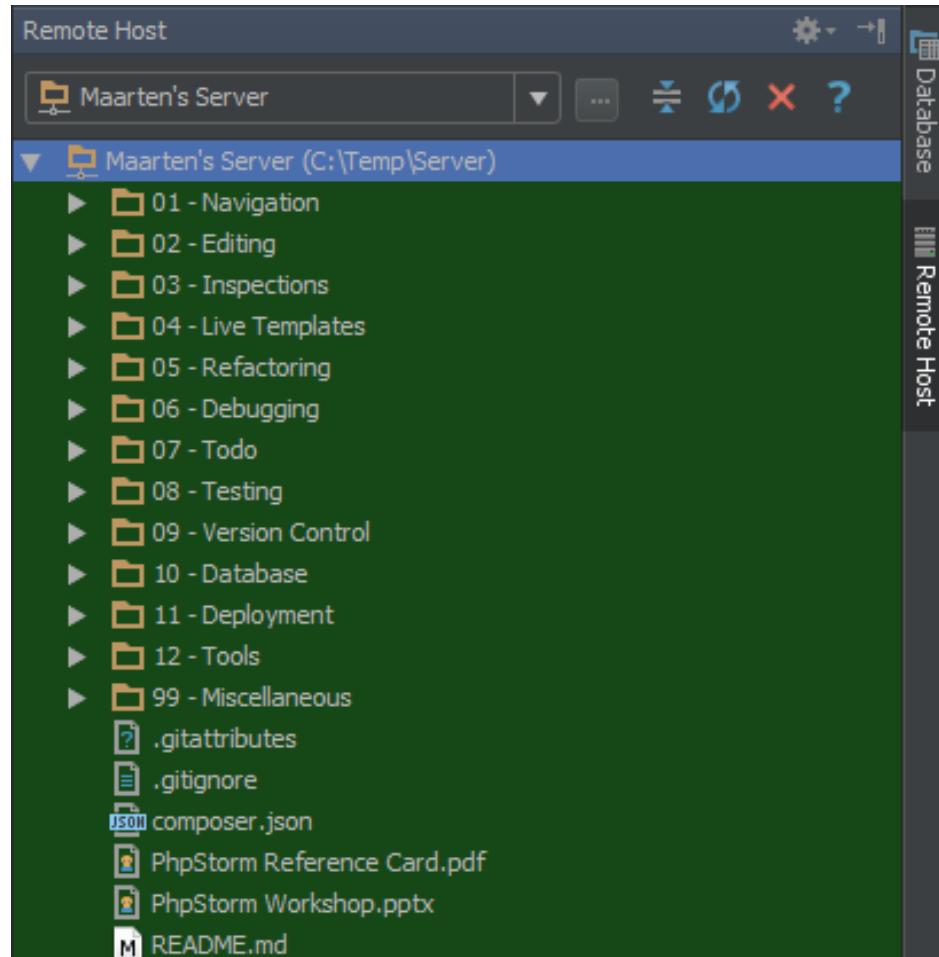
Mac: ⌘+Space / ⌘+Shift+Space

Deployment

Remote Hosts

Copy files and folders back and forth between local project and server.

Can be FTP, FTPS, SFTP, Local or Mounted Folder or In Place.

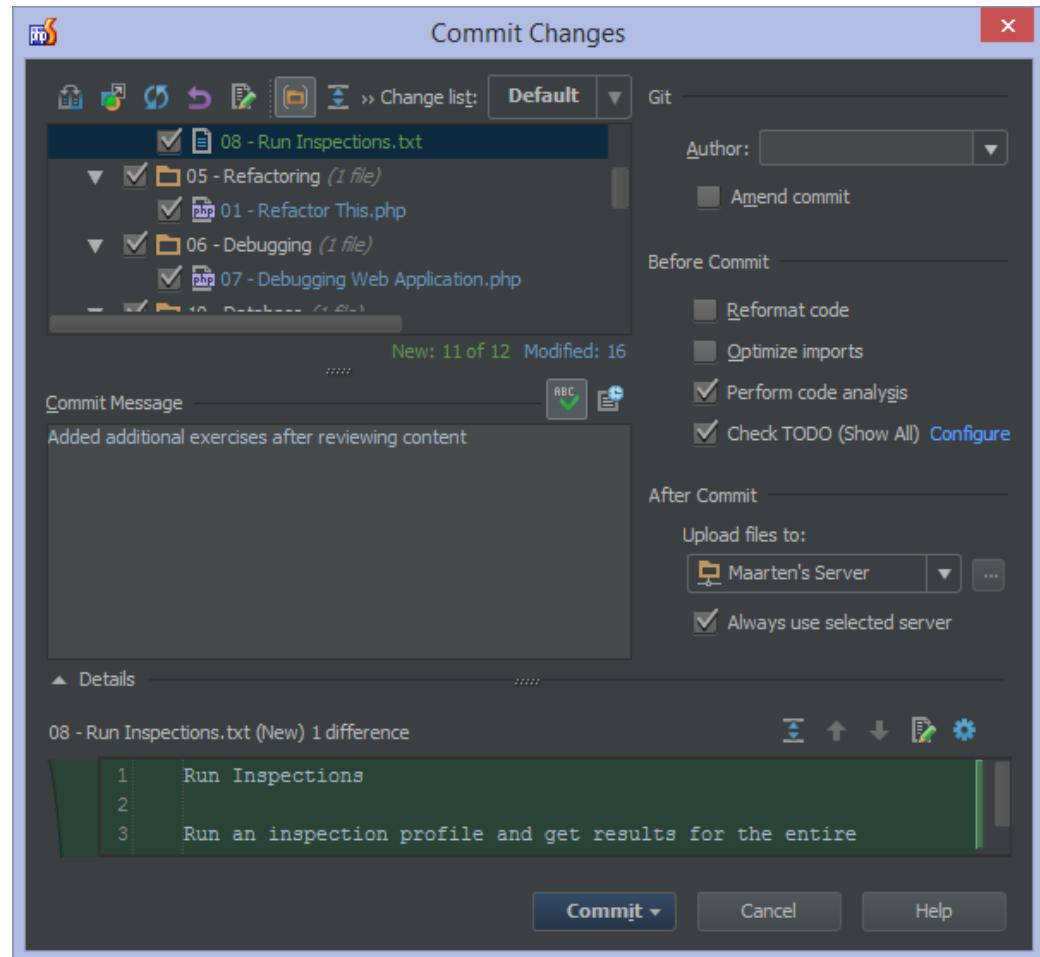


Integration with VCS

Upload files to server when committing to source control.

Useful for development servers.

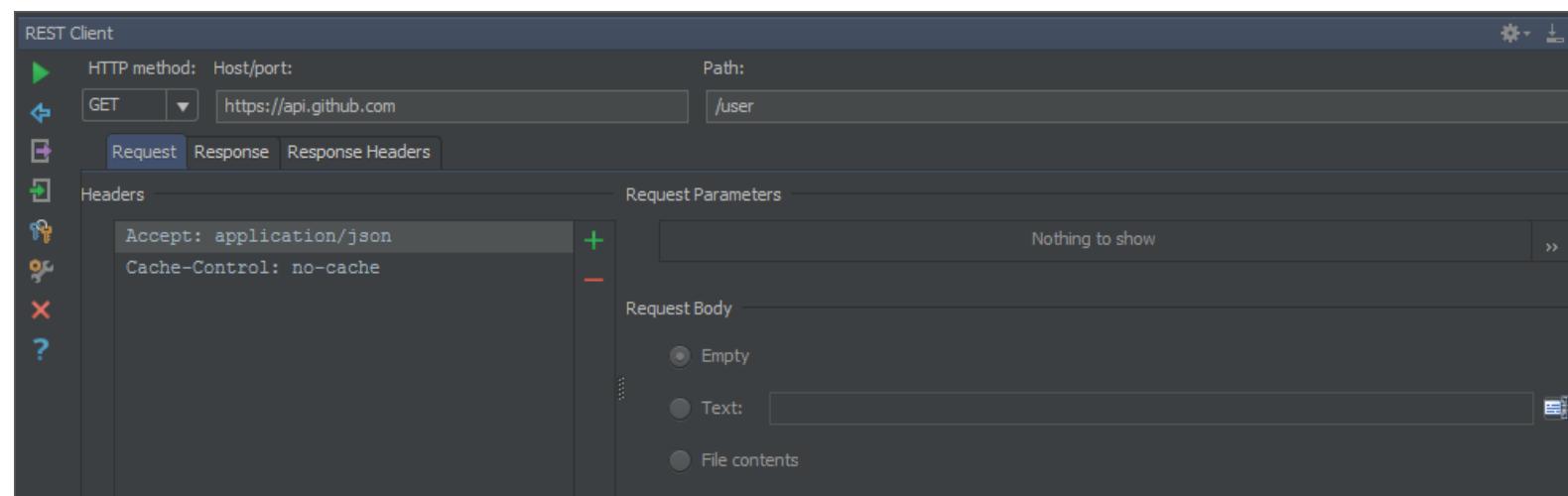
For production servers better to use continuous integration server.



Tools

REST Client

The built-in REST client allows testing web API's. Build requests, inspect responses.



Composer

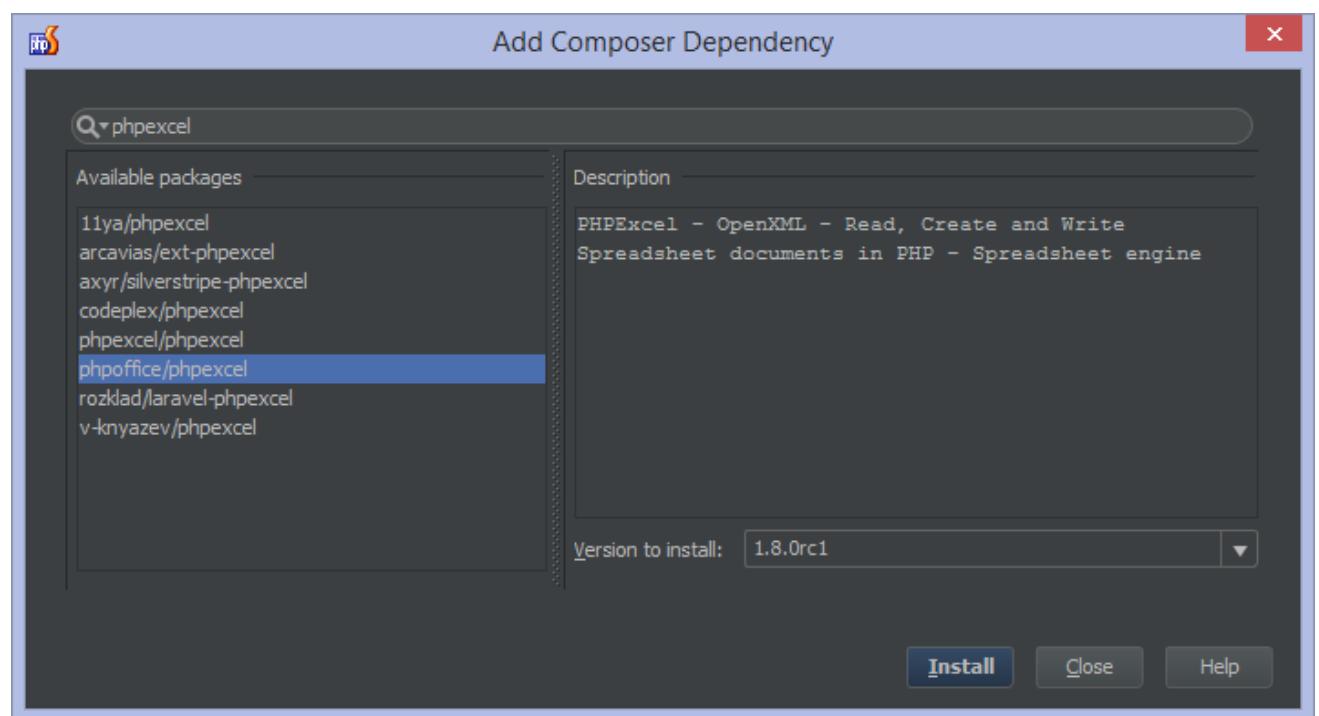
Work with [Composer](#) dependency manager in the IDE.

Initialize Composer

Search and install Composer dependencies

Create new project using Composer project type.

Command Line Tools (see further)



Command Line Tools

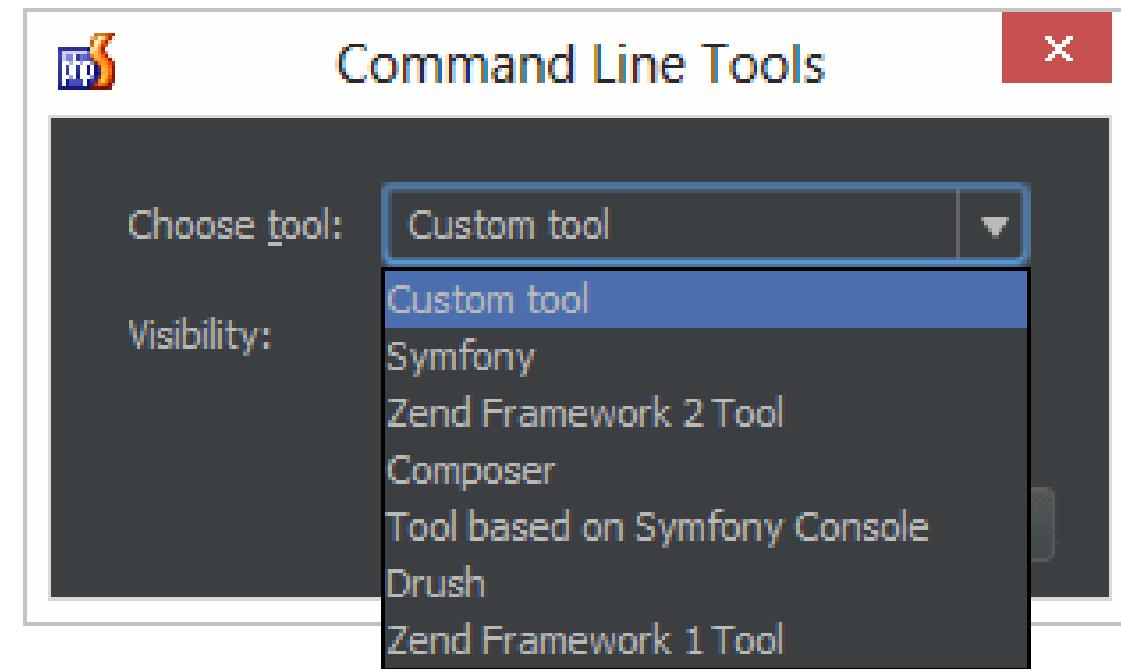
Invoke command line tools shipped with frameworks and libraries.

Composer, Zend Framework, Symfony, Symfony Console-based, Drush or roll your own

Autocompletion support

Not meant to be a full console/terminal!

-   Ctrl+Shift+X
-  Command+Shift+X

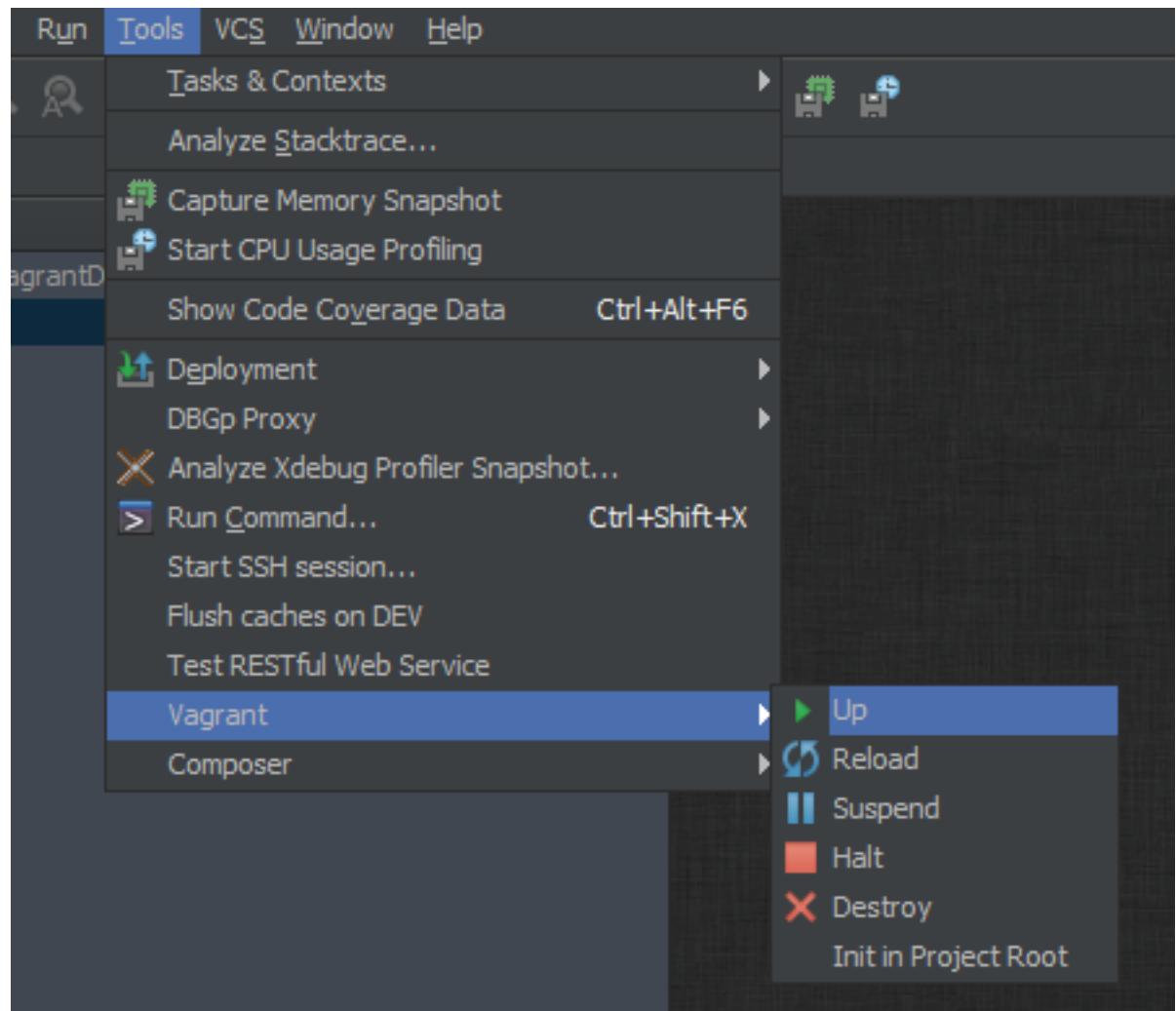


Vagrant

Vagrant is a tool which helps us create reproducible development environments by scripting a virtual machine.

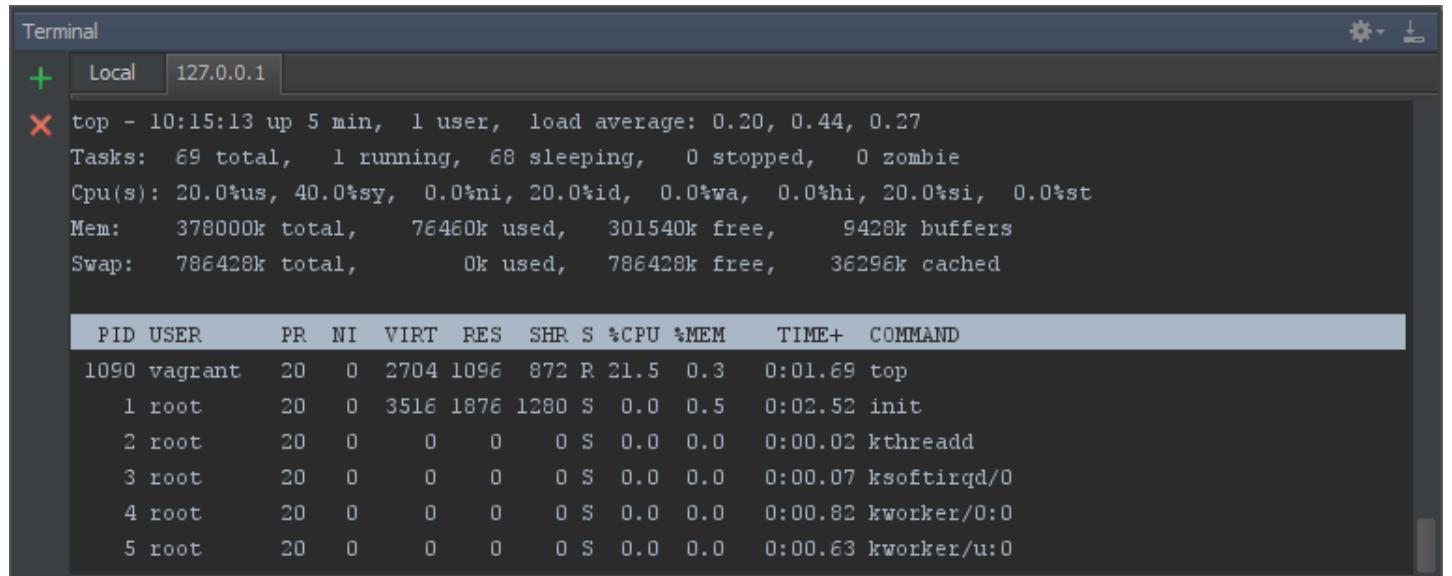
See

<http://blog.jetbrains.com/phpstorm/2013/08/vagrant-support-in-phpstorm/>



Remote SSH Terminal

Connect to a remote SSH server.
Provides a full SSH terminal in the
IDE.



The screenshot shows a terminal window titled "Terminal". The tab bar at the top has two tabs: "Local" (which is currently selected) and "127.0.0.1". Below the tabs, there is a red "X" icon. The main area of the terminal displays the output of the "top" command. The output shows system load average (0.20, 0.44, 0.27), tasks (69 total, 1 running, 68 sleeping, 0 stopped, 0 zombie), CPU usage (Cpu(s): 20.0%us, 40.0%sy, 0.0%ni, 20.0%id, 0.0%wa, 0.0%hi, 20.0%si, 0.0%st), memory usage (Mem: 378000k total, 76460k used, 301540k free, 9428k buffers, Swap: 786428k total, 0k used, 786428k free, 36296k cached), and a detailed process list.

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1090	vagrant	20	0	2704	1096	872	R	21.5	0.3	0:01.69	top
1	root	20	0	3516	1876	1280	S	0.0	0.5	0:02.52	init
2	root	20	0	0	0	0	S	0.0	0.0	0:00.02	kthreadd
3	root	20	0	0	0	0	S	0.0	0.0	0:00.07	ksoftirqd/0
4	root	20	0	0	0	0	S	0.0	0.0	0:00.82	kworker/0:0
5	root	20	0	0	0	0	S	0.0	0.0	0:00.63	kworker/u:0

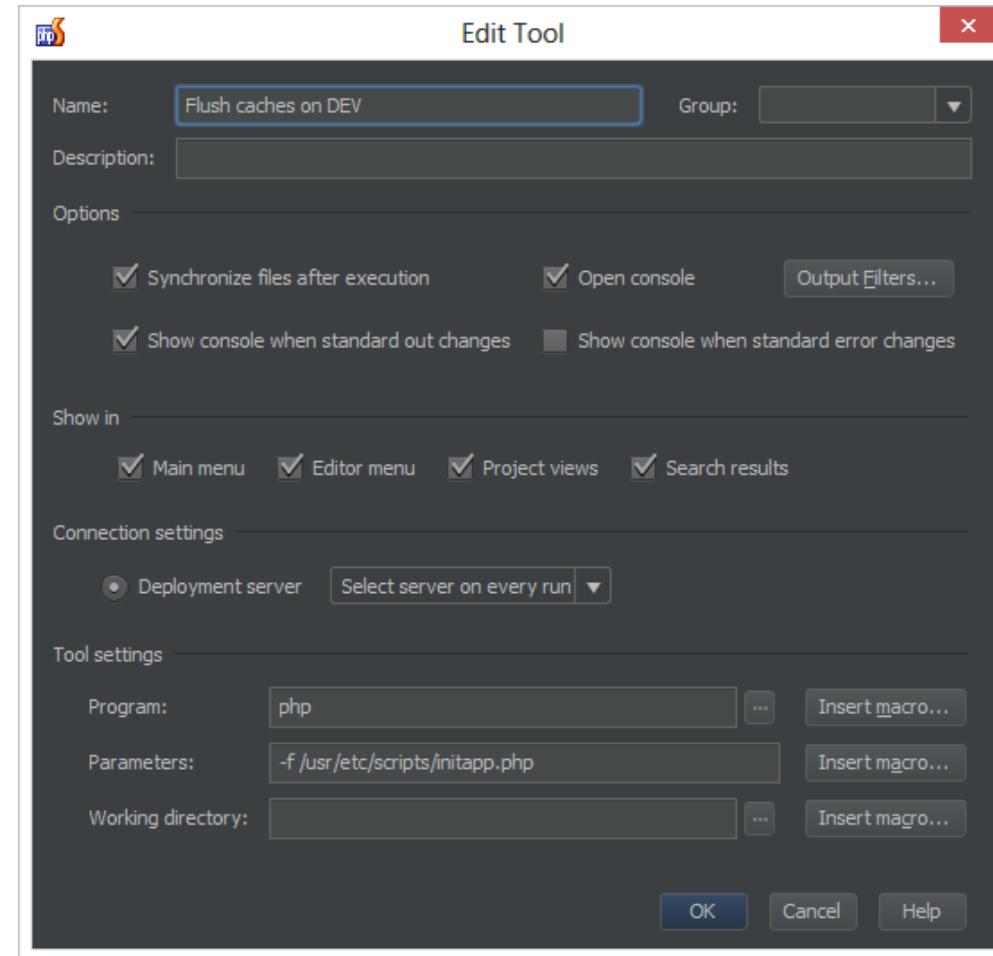
Remote SSH External Tools

Invoke remote commands over SSH without having to manually log in to the remote server.

Add them to PhpStorm menus for easy access.

Assign keyboard shortcut.

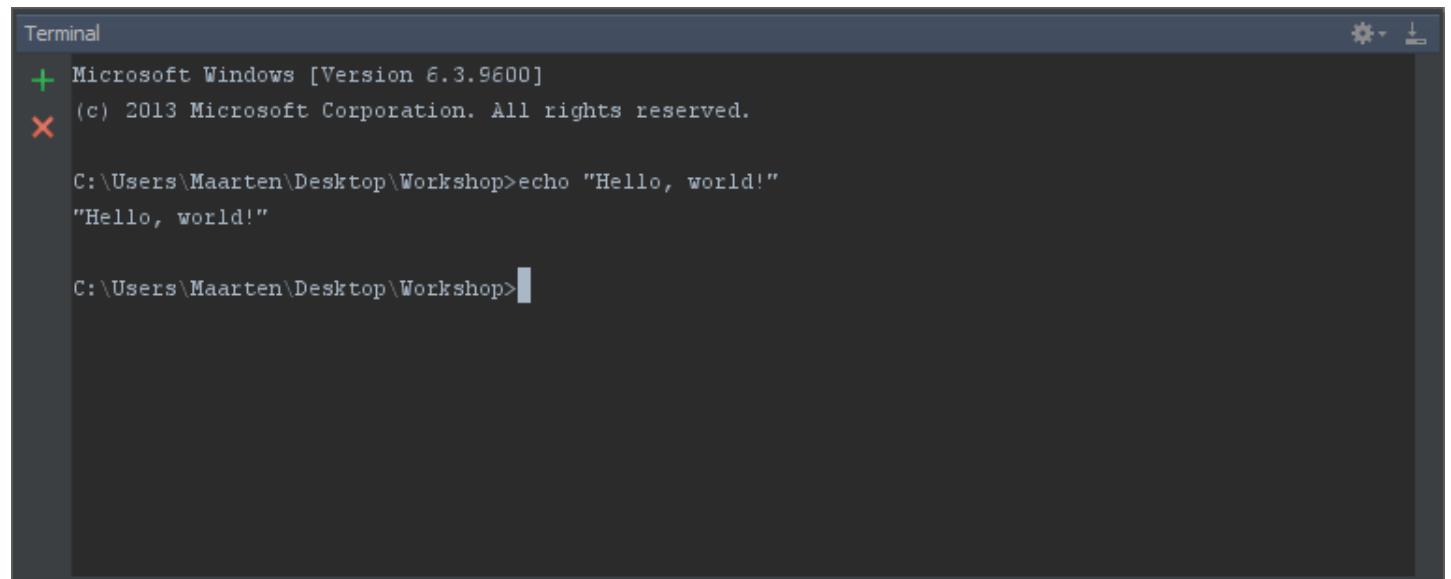
<http://confluence.jetbrains.com/display/PhpStorm/Using+the+PhpStorm+built-in+SSH+terminal+and+remote+SSH+external+tools#UsingthePhpStormbuilt-inSSHTerminalandremoteSSHexternaltools-WorkingwithremoteSSHexternaltools>



Local Terminal

Provide a local terminal. Works on any platform supported by PhpStorm.

- Windows icon | Ctrl+Space / Ctrl+Shift+Space
- Linux icon | Ctrl+Space / Ctrl+Shift+Space



The screenshot shows the PhpStorm IDE interface with a terminal window open. The terminal title bar says "Terminal". Inside the terminal, there is one session listed: "Microsoft Windows [Version 6.3.9600]". The session output shows the command "echo "Hello, world!" being run and its output "Hello, world!". The terminal has a dark theme and includes standard Windows-style icons for closing and minimizing the window.

Plugins

PhpStorm = IntelliJ IDEA – JAVA + plugins

Base IDE enriched with plugins

Some plugins are for paid IDE only

Which is cool: WebStorm features can be installed in PhpStorm

A lot of free / open source plugins at <http://plugins.jetbrains.com/?phpStorm>

Install from IDE Settings | Plugins



Some interesting JetBrains plugins

LiveEdit - view changes live in browser and vice-versa

NodeJS - install Node.js packages (NPM) in a similar way to Composer

IdeaVim - adds VIM capabilities to the IDE

Handlebars/Mustache and EJS - plugins for working with client-side template engines

JSTestDriver and Karma - run JavaScript unit tests

Puppet Support - provides editor support for puppet, very convenient with Vagrant

...

Some interesting open source plugins

'Copy' on steroids - copy rich text from the editor

Markdown - provides Markdown editor support and syntax highlighting

BashSupport - support bash files in editor

AngularJS - support for AngularJS

Symfony, Yii, CakeStorm, ... - framework-specific plugins

CodeGlance - the editor scrollbar on steroids

EditorConfig - support for EditorConfig settings

Get Gist - fetch a Gist from GitHub and insert it into code

...



There is much more...

What we did not cover...

Be sure to try these afterwards!

File Watchers – monitor a file and run command when it changes (e.g. LESS to CSS)

Google App Engine – work with GAE development environment and deploy

Drupal Support – project support, code style, hooks, ...

Live Edit – view changes live in browser and vice-versa

Issue Tracker – open and browse issues from YouTrack, JIRA, GitHub, ...

HTML, CSS, JS tools

Framework Integration – integrate with Symfony, Yii, ...

GitHub support – pull requests, gists, ...



Resources

Resources

Keyboard reference - <http://bit.ly/phpstorm-shortcuts>

Blog - <http://blog.jetbrains.com/phpstorm/>

Web Help - <http://www.jetbrains.com/phpstorm/webhelp/>

Tutorials - <http://confluence.jetbrains.com/display/PhpStorm/Tutorials>

Courseware (videos) - <http://bit.ly/phpstorm-videos>

Webinars - <http://blog.jetbrains.com/phpstorm/tag/webinar/>

Twitter - <https://twitter.com/phpstorm>

