Debugging PHP CLI scripts with PhpStorm

Not all PHP applications are web applications. Many of us write scripts that run in the PHP CLI. Our own command line tools, our own daemons, a message queue processing application and other types of applications typically run from the command line. In this tutorial, we'll see how we can debug PHP CLI scripts with PhpStorm.

There are various ways to start a debugging session. We can start a session from within PhpStorm and make it start our script and attach the debugger to it. Another way is to let PhpStorm listen for incoming debugger connections and start the script outside the IDE. We'll have a look at both options.

Be sure to check the other tutorials about debugging PHP code with PhpStorm to learn more, including troubleshooting tips and advanced debugging techniques.

Prerequisites

To debug PHP code with PhpStorm, we will need Xdebug or Zend Debugger. Make sure either Xdebug or Zend Debugger are installed and configured with PhpStorm.

Starting a Debugging Session from PhpStorm

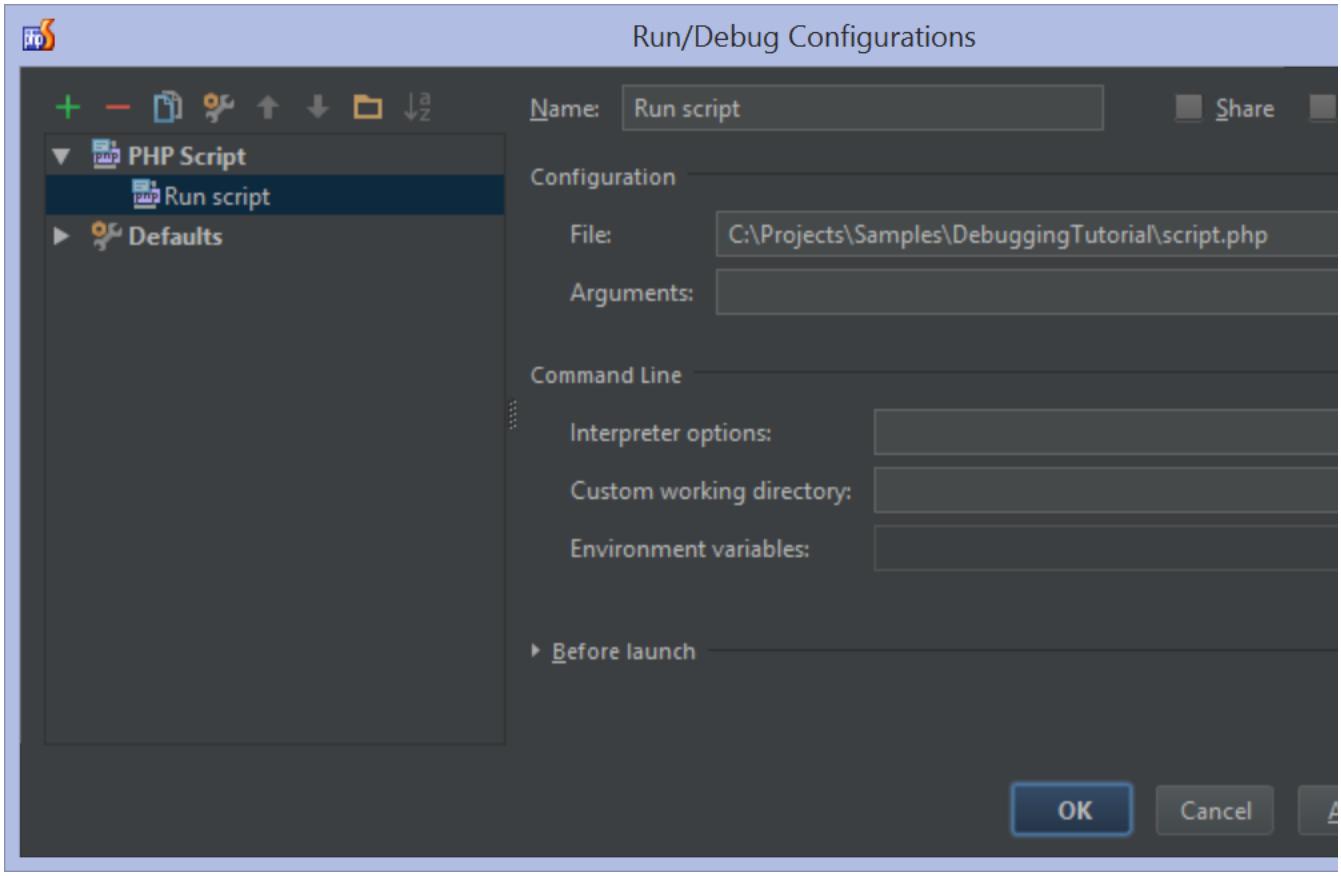
Let's see how we can start debugging a PHP CLI script from within PhpStorm.

1. Create a Run/Debug Configuration

PhpStorm uses Run/Debug configurations to execute a script from within the IDE. A configuration can define additional arguments for the PHP interpreter, as well as launch other commands prior to starting our script. We will need a Run/Debug configuration to start the debugger from within PhpStorm.

Manually creating a Run/Debug configuration for a PHP script

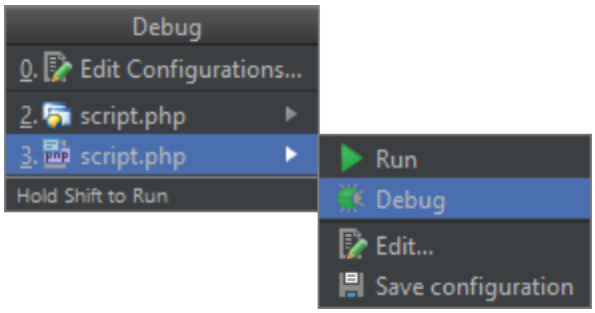
Create a new Run/Debug configuration using the Run | Edit Configurations menu. Next, add a new configuration of the type PHP Script and provide the required parameters, such as the script to be executed.



Click Save to store the Run/Debug configuration.

Generating a Run/Debug configuration for a PHP script

We can let the IDE generate a Run/Debug configuration for a script for us. From the Project tool window, use the Debug | <script name>.php context menu (make sure to pick the one which has a PHP icon). We can also open our script in the editor and press Alt+Shift+F9 (Ctrl+Alt+D on Mac OS X) and select the script to be debugged.



The IDE will launch the script with the debugger enabled, and open the debugger tool window.

2. Launch the Debugger

Before launching the debugger, make sure a breakpoint is set (or the Run | Break at first line in PHP scripts option is enabled).

From the toolbar, click the "bug" icon to start the debugger. We can also use the Run | Debug menu or the Shift+F9 keyboard shortcut (CMD+F9 on Mac OS X). This will launch our script with the debugger attached, and pauses execution at the first breakpoint.

Continue debugging using the tools and techniques outlined in Using the PhpStorm Debugger.

Starting a Debugging Session from the Command Line

When running CLI scripts one of the requirements below should be met to start a debugging session with IDE:

- Xdebug's remote_autostart setting is enabled
- XDEBUG_CONFIG environment variable exists

Let's go ahead and configure the environment, starting with IDE.

1. Listen for Incoming Debugger Connections

From the toolbar, toggle the "Listen debugger connections" button. We can also use the Run | Start Listening for PHP Debug Connections menu. Enabling this option will ensure PhpStorm reacts when a debugging session is started and opens the debug tool window for us.

Before launching the script, make sure a breakpoint is set (or the Run | Break at first line in PHP scripts menu is enabled).

2. Start the Script with Debugger Options

Since we'll be starting the script from the command line, we will have to make sure it is started with the required settings to enable the debugger. Let's see how we can start our PHP CLI script with debugging enabled using either Xdebug or Zend Debugger.

Starting the script with Xdebug

Xdebug has various configuration options which we can use to let the PHP interpreter reach out to PhpStorm. These parameters have to be passed to the PHP interpreter using the `-d` command line switch. A more convenient way will be to set an environment variable so we don't have to provide the `-d` switches all the time.

Using PHP command line switches

To start our script with debugging we have to:

1. Set an environment variable that would tell XDebug to connect to IDE:

```
Windows:
set XDEBUG_CONFIG="idekey=123"

Linux / Mac OS X:
export XDEBUG_CONFIG="idekey=123"
```

   Here idekey should have a random value.
2. Launch PHP with several switches:

```
php -dxdebug.remote_enable=1 -dxdebug.remote_mode=req -dxdebug.remote_port=9000
-dxdebug.remote_host=127.0.0.1 -dxdebug.remote_connect_back=0 path/to/script.php
```

   You might use 10.0.2.2 instead of 127.0.0.1 with Vagrant (see related SO question).

Using an environment variable

To start our script with debugging, we can set an environment variable that configures Xdebug:

```
Windows:
set XDEBUG_CONFIG="remote_enable=1 remote_mode=req remote_port=9000
remote_host=127.0.0.1 remote_connect_back=0"

Linux / Mac OS X:
export XDEBUG_CONFIG="remote_enable=1 remote_mode=req remote_port=9000
remote_host=127.0.0.1 remote_connect_back=0"
```

Next, we can start our script like we would normally do:

```
php path/to/script.php
```

Optionally, we can use Xdebug's remote_autostart setting to always start a debugging session for every script that is run.

Starting the script with Zend Debugger

Zend Debugger has various configuration options which we can use to let the PHP interpreter reach out to PhpStorm. These parameters have to be passed to the PHP interpreter using an environment variable.

First, we have to set the `QUERY_STRING` environment variable:

```
Windows:
set
QUERY_STRING="start_debug=1&debug_host=127.0.0.1&no_remote=1&debug_port=10137&deb
ug_stop=1"

Linux / Mac OS X:
export
QUERY_STRING="start_debug=1&debug_host=127.0.0.1&no_remote=1&debug_port=10137&deb
ug_stop=1"
```

Next, we can start our script like we would normally do:

```
php path/to/script.php
```

Optionally: to tell the PhpStorm which path mapping configuration should be used for a connection from certain machine, the value of the PHP_IDE_CONFIG environment variable should be set to serverName=SomeName, where SomeName is the name of the server configured in Settings / Preferences | Languages & Frameworks | PHP | Servers:

```
Windows:
set PHP_IDE_CONFIG="serverName=SomeName"

Linux / Mac OS X:
export PHP_IDE_CONFIG="serverName=SomeName"
```

If this environment variable is not set - you'll be prompted to specify path mappings manually once IDE detects an incoming XDebug connection.

3. Debug!

Once the script is started, PhpStorm will open the debug tool window and break at the first breakpoint that was set in the script. Next, we can continue debugging our PHP CLI script using PhpStorm.

If the script that is being debugged is not a part of the project that's open in PhpStorm, the IDE will still open the script in the editor and pause execution at the first statement. This makes it possible to quickly debug any PHP CLI script, even if we don't have a PhpStorm project for it yet.