Debugging a PHP HTTP Request

Besides debugging the entire application, you can debug separate HTTP Requests. This is 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 IntelliJ IDEA is supported through the PHP HTTP Request run configuration. Based on the configuration settings, IntelliJ IDEA composes the request to run.

On this page:

Preparing the debugging engine

Before you start debugging, make sure that you have a debugging engine installed and configured properly. IntelliJ IDEA supports debugging with two most popular tools: XDebug and Zend Debugger. 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. To find out which `php.ini` file is active, create and run a test file with `phpinfo()`, then search for the *Loaded Configuration File*.

For more information on configuring debugging engines, see Configuring XDebug, Configuring Zend Debugger, http://confluence.jetbrains.com/display/PhpStorm/Xdebug+Installation+Guide , and http://confluence.jetbrains.com/display/PhpStorm/Zend+Debugger+Installation+Guide.

Setting the breakpoints

Breakpoints are source code markers used to trigger actions during a debugging session. Typically, the purpose behind setting a breakpoint is to suspend program execution to allow you to examine program data. However, IntelliJ IDEA 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 IntelliJ IDEA project so you can reuse them across debugging sessions.

1. Place the caret on the desired line of the source code.

   Breakpoints can be set in the PHP context inside `*.php`, `*.html`, and files of other types. Only executable lines are valid locations for line breakpoints. Comments, declarations, and empty lines are not valid locations for the

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`.

Creating a PHP HTTP Request debug configuration

IntelliJ IDEA 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 `Shift+Alt+F10`, then press `0` to display the **Edit Configuration** dialog box or select the configuration from the pop-up window and press `F4`.
2. Click + on the toolbar or press `Insert`. 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, IntelliJ IDEA 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.

Initiating a debugging session and examining the suspended program

1. To start debugging, click the **Debug** button on the toolbar.
2. As soon as the debugger suspends on reaching the first breakpoint, examine the application by analyzing *frames*. A *frame* corresponds to an active method or function call and 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 in the **Variables** and **Watches** panes. For more details, see the section Examining Suspended Program.
3. Continue running the program and examine its frames as soon as it is suspended.
   - To control the program execution manually, step through the code using the commands under the **Run** menu or toolbar buttons: **Step Into** (`F7`), **Step Out** (`Shift+F8`), **Step Over** (`F8`), and others. For more details, see Stepping Through the Program.
   - To have the program run automatically up to the next breakpoint, resume the session by choosing **Run | Resume Program** or pressing `F9`

See Also

Procedures:

- Debugging

Reference:

- Run/Debug Configuration: PHP HTTP Request
- Debug Tool Window

PHP Support:

- PHP Debugging Session

Web Resources:

- Developer Community