Command Line Debug Client

The command line debug client allows you to debug PHP scripts without having to set up an IDE.

Installation

A binary for Linux, macOS, and Windows is available on the **downloads** page. You only have to download the binary, which you can then run from a command line.

Command Line Options

The following command line options are available:

`-1`	Debug once and then exit
`-f`	Whether act as fully featured DBGp client. If this is enabled, the client will perform certain tasks for you, such as enabling *async* support. In the future, this mode will turn the dbgpClient in a fully fledged command line debugging client without the need for you to remember all DBGp commands.
`-h`	Show this help
`-p value`	Specify the port to listen on [`9003`]. This is the same port that Xdebug should initiate a connection on. On the Xdebug side, this port can be configured with the xdebug.remote_port setting.
`-r idekey`	If the `-r` option is given, the client will register itself with a debugging proxy (selected with `-y`), and then wait for incoming debugging connections as usual on the port configured with `-p`.
`-s`	Enables SSL. With this option on, the client expects incoming connections on the configured port to be in SSL. This is an *experimental feature* that is not fully

	finished. Right now, Xdebug does not support SSL yet, but the dbgpProxy does.
`-u idekey`	If the `-u` option is given, the client will unregister itself with a debugging proxy (selected with `-y`), and then quit.
`-v`	Show version number and exit
`-x`	Show protocol XML. With this option enabled, the client will also show the raw XML data that the debugging engine sends back. This can be useful for debugging issues with the interaction between the debugger engine, for example Xdebug, and this client.
`-y host:port`	Configures the host and port of an DBGp proxy. This option is used with `-r` and `-u` only.

Usage

To start the client on the command line on Linux, open a shell, and then run:

```
./dbgpClient
```

If the binary doesn't start or you get a `not found` message, please refer to this **FAQ entry**.

To start the client on the command line on MacOS, open a shell, and then run:

```
./dbgpClient-macos
```

On Windows, open a command a *Command Prompt* and run:

```
dbgpClient.exe
```

In all cases, you can add the **command line options** as described above.

When the debug client starts, it shows some version information. It will then wait until a connection is made to it by a debugging engine (such as Xdebug):

```
Xdebug Simple DBGp client (0.4.2)
Copyright 2019-2020 by Derick Rethans

Waiting for debug server to connect on port 9003.
```

After a connection is received, the debug client shows basic information, and then shows the `(cmd)` prompt and waits for input:

```
Connect from [::1]:51884
DBGp/1.0: Xdebug 3.0.0 — For PHP 7.4.3-dev
Debugging file:///tmp/xdebug-test-2.php (ID: 5242/dr)
(cmd)
```

On this command prompt you can then enter the available DBGp commands. With `{tab}` you can auto complete your command. You can use arrow up to a previous line, and use `ctrl-R` to search through your previous issued commands.

Here we step into the next line (the first line, in our case), and see which variables are available:

```
(cmd) step_into
1 | step_into > break/ok
1 | file:///tmp/xdebug-test-2.php:14

(cmd) context_get
2 | context_get
2 | uninitialized $a
```

The following commands and options are common:

Command	Description
`breakpoint_set -t line -f file:///xdebug-test-2.php -n 5`	Sets a breakpoint on line 5 of `file:///xdebug-test-2.php`
`step_into`	Steps to the next executable line in the code
`run`	Runs the script until the next breakpoint, or when the script ends
`context_get`	Lists the variables and their values in the current scope
`property_get -n $a`	Retrieves the value of the property $a

There is a full description in the **DBGp documentation**.

If the client has been started in "fully featured mode" (`-f`), and you're running the latest Xdebug from **GitHub**, then it is possible to pause a running debugging session by pressing `Ctrl-C` . The client will return to the prompt where you can then issue commands as normal.

On the client's prompt you can abort the debugging session, and then the client, with `Ctrl-C`.