Speare Code Editor
The Small PHP IDE for PHP Development

Version 1.0
Update: 4 Mar 2020

Free, Lightweight, Open Source, Extendable Flexibility

Speare (http://sevenuc.com/en/Speare.html) is an ultra lightweight code editor and a small IDE that has an efficient code navigation and call routines tracing ability, and has integrated debugging environment for C, C++, PHP and all kinds of PHP web frameworks. It was originally developed to providing a native scripting language debugging environment that seamlessly integrated with C and C++. It not only has very concise user interface but also has a very flexible architecture to easily add a new programming language code runner, parser, syntax highlighting, code formatter and debugger in it.

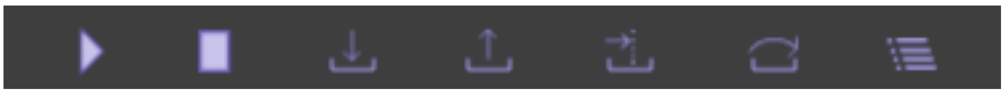
For general Speare code editor usage, please refer this document:
http://sevenuc.com/download/Speare_quick_reference.pdf

Debug Mode

1. Show the debug toolbar

Click siding bottom button.

2. Debug toolbar



From left to right, Start, Stop, Step Into, Step Out, Run To, Step Over, Show Watches.

The "Step Over" is equals to "Step next", and "Step To" is equals to "Continue" in common debugging words, and the "Step To" is the command that tell the debugger run to meet a breakpoint or an exception occurred or the program meet exit.

On the rightmost there are three other function units, they are, search items in the stackview, siding stackview, and clean the debug output.

Search in the debug output

Click in the output area and use the shortcut key "Control + F" to do the searching.

3. Socket Port

You can set the socket communication port number both used by Debug Server and the Speare code editor. Open the Preferences of Speare and select the "Debug Settings" tab then input your number.

Note: Please remember to empty the port number when you switched to debugging with the default builtin programming languages with default port number.

4. Watches

Watches used to evaluate variable or expression and their values can be realtime showing in stackview when debugging session paused, the nodes normally has a green colour and always placed on the top of stackview.

Caution:

a. Please ensure all source files have been dragged in the left side Treeview (Workspace Explorer) before start a debug session, because macOS app can't be allowed to access files outside of its sandbox.

b. When your source code file moved to another folder, you must drag the source code folder in Speare again then the debugging can correctly work.

C and C++ Debugger

The C and C++ debugger of Speare code editor implemented as a script client of LLDB (http://lldb.llvm.org/), and supports extend it by yourself. The builtin module supports parsing modern C++ source code files including C++11 and C++14 syntax, e.g namespace, anonymous function, structure, union, class, enum etc and so many new features of the C++ programming language. You can enjoy debugging almost any type of C and C++ applications under the lightweight debugging environment of Speare code editor.

Start Debugging Steps:

1. Download Speare Debug Server:
→ http://sevenuc.com/download/c_debugger.tar.gz (10KB)

The source code of the C and C++ debugger can be view online here: https://github.com/chengdu/Speare or here: https://sourceforge.net/projects/speare

2. Uncompress the tarball:
Uncompress it to your local directory. e.g ~/Desktop and take a look at the readme.txt in it.

3. Start the debug server:
Please refer the readme.txt file.

4. Debug session start:
click "Start" button on the debug toolbar of Speare code editor.
Add breakpoint, step in, step out, step next, watch stack trace ...

5. Run extra commands:
Right click in the stackview (bottom left side) and then input any lldb command when the debugging session paused. Left click anywhere outside of the input box to close it and the command will be directly send to the debug server.

a. Add function breakpoints
. breakpoint set --name functioname: add a C function breakpoint.
. breakpoint set --name classname:: functioname: add a C++ function breakpoint.

b. Process operation
. process attach --name xxx --waitfor: attach another process by name.
. process attach --pid xxx: attach another process by pid #xxx.

c. Thread operation
. thread list: show all thread of current process.
. thread select 2: select thread #2.
. thread backtrace all: show thread info.
. register read: read all CPU registers.
. thread step-inst: step one machine instruction.
. thread step-over-inst: step return one machine instruction.

d. Watchpoint operation
. watch list -v: list watchpoints.
. watchpoint set variable x: add a watchpoint x.

e. Frame operation
. frame list: print all frame of the current thread.
. frame select 9: select frame #9.

f. Display variable value
. frame variable x: print x value.
...

Tips: Run to (run to meet a breakpoint), the source file that you want debugger stopped in it must already opened and has at least one breakpoint before run the command.

Modify the C and C++ debugger

You can directly modify the script client of lldb to satisfy your requirements.

PHP Debugger

The PHP debugger of Speare code editor supports all kinds debugging of PHP applications and any version of PHP interpreter that has Xdebug support from PHP 5.x to PHP 7.x. Different with Lua, Ruby and Python debugging, this time Speare acts as debug server and Xdebug as the client.

Setting up Xdebug for PHP Debugging:

1. Download Xdebug:
https://xdebug.org/files/xdebug-2.6.0.tgz

2. Compile and install Xdebug:

```
$ rm configure.in (optional)
$ rm configure.ac (optional)
$ phpize && ./configure --enable-xdebug && make clean && make all
$ sudo make install
```

At this step Terminal will report:
```
/usr/lib/php/extensions Operation not permitted.
```

This is because SIP default set to be enabled by macOS, even you execute sudo operation, the system protected directories still can't be writable.

WARNING: THE FOLLOWING OPERATION IS VERY DANGERS.

Assuming that you know what you’re doing, here is how to change SIP (System Integrity Protection) settings on your Mac. Turn off your Mac (Apple → Shut Down...), hold down Command-R and press the Power button. Keep holding Command-R until the Apple logo appears and wait for OS X to boot into the "OS X Utility" window and then choose Utilities → Terminal.

Turn off SIP:
```
$ csrutil disable
$ csrutil status
$ reboot
```

Turn on SIP:

```
$ csrutil enable
$ csrutil status
$ reboot
```

After you turn off SIP and execute "$ sudo make install" again, Xdebug should be successfully installed on you system, you can check it by this command:

```
$ php –v
```

It should print something like the following:

```
PHP 7.1.23 (cli) (built: Nov 27 2018 16:59:25) ( NTS )
Copyright (c) 1997-2018 The PHP Group
Zend Engine v3.1.0, Copyright (c) 1998-2018 Zend Technologies
with Xdebug v2.6.0, Copyright (c) 2002-2018, by Derick Rethans
```

3. Configuring web server:

```
$ cp /etc/apache2/httpd.conf ~/Desktop/
```

```
LoadModule userdir_module libexec/apache2/mod_userdir.so
LoadModule alias_module libexec/apache2/mod_alias.so
LoadModule rewrite_module libexec/apache2/mod_rewrite.so
LoadModule php7_module libexec/apache2/libphp7.so
Include /private/etc/apache2/other/*.conf
```

Edit file http.conf and ensure the above lines not be commented.

```
$ sudo cp ~/Desktop/httpd.conf /etc/apache2/
```

Save back the settings file.

```
$ apachectl configtest
```

Check there syntax is legal (optional).

4. Configuring PHP interpreter:

```
$ cp /etc/php.ini.default ~/Desktop/
```

```
enable_dl = On
[xdebug]
zend_extension = /usr/lib/php/extensions/no-debug-non-zts-20160303/xdebug.so
xdebug.remote_enable = 1
xdebug.remote_host = "127.0.0.1"
xdebug.remote_port = 9000
xdebug.remote_handler = "dbgp"
xdebug.remote_mode = req
xdebug.remote_connect_back = 1
xdebug.remote_autostart=1
```

Edit file php.ini.default and carefully check the above content have been written in it.

Note: xdebug.remote_autostart=1 should be removed at product environment.

```
$ sudo cp ~/Desktop/php.ini.default /etc/
```

Save back the settings file.

```
$ sudo mv /etc/php.ini.default /etc/php.ini
```

Rename the settings file to take effect.

5. Run PHP test with Xdebug:

```
<?php
echo phpinfo();
?>
```

Save the above content in a file named test.php and put it under /Library/WebServer/Documents/

```
$ sudo launchctl unload -w
/System/Library/LaunchDaemons/org.apache.httpd.plist
$ sudo launchctl load -w
/System/Library/LaunchDaemons/org.apache.httpd.plist
```

Restart apache, launch Safari and request: http://127.0.0.1/test.php. There're some text in zend engine section should be as same as printed by execute the command "$ php -v" on command line.

Steps of PHP Debugging in Speare code editor:

1. Launch Speare and dragging the PHP source code folder into the "Workspace Explorer".

2. Select the startup php script and click siding bottom button to show the debug toolbar and then click "Start" button.

3. Launch Safari and request: http://127.0.0.1/xxx/xxx/index.php

Note: Don't use localhost but should use 127.0.0.1 instead.

4. After Speare paused and highlighting at a special line of your startup PHP script there that means the debugging session started, you can execute the common debugging command now:

Add breakpoint, step in, step out, step next, watch stack trace ...

Switch PHP interpreter

You can switch any version of PHP interpreter directly; it does not affect the server side Speare code editor. PHP is an excellent programming language to develop Web Applications based on frameworks such as Drupal, Zend Framework, CodeIgniter, Symfony and Yii framework etc, but not limited that, in fact PHP is also very suitable to develop command line applications.

Extend Speare Code Editor for PHP development

Speare Code Editor can be easily extended to support any type of PHP development including web applications that based on frameworks such as Drupal, Zend Framework, CodeIgniter, Symfony and Yii framework and all kinds of command line applications.

To add scripts to better support debugging PHP in Speare code editor, please download the guide from here: http://sevenuc.com/download/language_extension_protocol.pdf, and following the description in it.