Speare Code Editor
The Small IDE for C and C++ Development

Version 0.0.2
Update: 4 Mar 2020

Free, Lightweight, Open Source, Extendable Flexibility

Speare (http://www.sevenuc.com/en/speare.html) is a small and friendly code editor. It was originally developed to provide a native scripting language debugging environment that seamlessly integrated with C and C++. Speare designed to make programming feels light, simple and free. Speare not only has an efficient code navigation and call routines tracing ability but also has flexibility to extend it to support special developing requirements. Speare includes an ultra lightweight debugging environment for C, C++, Ruby, mruby, Lua, Python, PHP, Perl and Tcl, and give people complete freedom to control and customise the debugging environment for a new programming language.

Debug C and C++ Projects

1. Show the debug toolbar

Click main menu "Navigate" → "Toggle Output".

2. Debug toolbar

From left to right: Start, Stop, Step Into, Step Out, Run To/Continue, Step Over/Step Next, Show Watches. The command "Run To" tell the debugger run to meet a breakpoint or an exception occurred or the program exited. On the rightmost there are three other icons, they are, search items in the stackview, siding stackview, and clean debug output.

Search in the debug output

Click in the output area to let it got focus and use the shortcut key "Command + F" to do the searching.

3. Socket Port

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

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

4. Watches

Watch List: click debug toolbar "Show Watches" to manage watched variables and expressions, batches of watched items can be removed by multiple selection.

a. Whenever values of watched variables changed, debug session will pause at that point.
b. The values of evaluated expressions will be sent as a *"virtual stack node"* that paired the expression and its value together and representing as JSON key and value.

When watched variables and expressions display in stackview, their nodes normally has a different icon and text colour, and always placed on the top of stackview.

Add Watches:

a. Click debug toolbar "Show Watches".
b. Right click stackview and select menu item "Watch Variable".

Remove Watches:

a. Click debug toolbar "Show Watches".
b. Right click stackview and select menu item "Remove Watch".

5. Run Extra Commands

a. Click main menu "Debug" → "Send Debug Command".
b. Right click stackview and select menu item "Send Command".
When send an extra debug command, the input box will prompt in

stackview, so please click stackview to let it got focus before select menu item.

6. Show Stackview Item Values

Right click stackview and select menu item "View Value AS", variables can be configured to display as "Hex", "Decimal", "Octal", "UTF-8" and "Unicode" sequences.

Caution:

Please ensure all source files have been dragged in the workspace before start a debug session, because macOS app can't be allowed to access files outside of its sandbox.

Startup Debugging Session

1. Launch a Terminal.app window or tab, and start Debug Server.
Note: *The content directly printed by the debugging program will be displayed in the terminal instead of in the debug "output view".*

2. Click start button on the "Debug toolbar" to start debugging session.

3. Continually click "Step Over" button several times on the "Debug toolbar" to ignore some initialising steps in debugging session until it reached the main entry point.

The 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 directly customised to meet special requirements of your projects.

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.

Pretty Debug Output Printing

Speare allows keywords highlighting in console output, the colour definition file located in Speare.app/Contents/Resources/console.plist, you can directly add some keywords to support your special debugging output colour scheme.

Note: *valid colour value must a hex string that has seven characters, e.g , right click the editor → select "Colour Picker" then you can fetch such a HTML colour value conveniently.*

Modify the C and C++ Debugger

You can directly modify the source code of the C and C++ debugger to satisfy your requirements.

Extend Speare Code Editor for C and C++ Development

Speare code editor well support C and C++ programming, any type of C and C++ project, from embedded systems to desktop applications, very big games and sever side projects. To add scripts to better debugging C and C++ code in Speare code editor, please download the guide document and following the description in it.