

C++ Program Design



-- **vscode: hello world!**

Junjie Cao @ DLUT

Summer 2021

<https://github.com/jjcao-school/c>

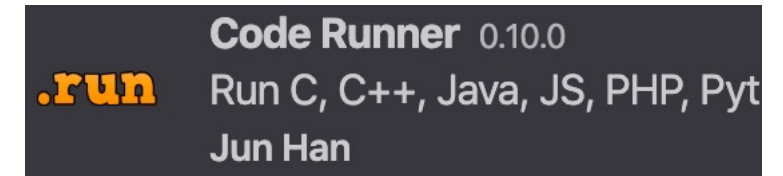
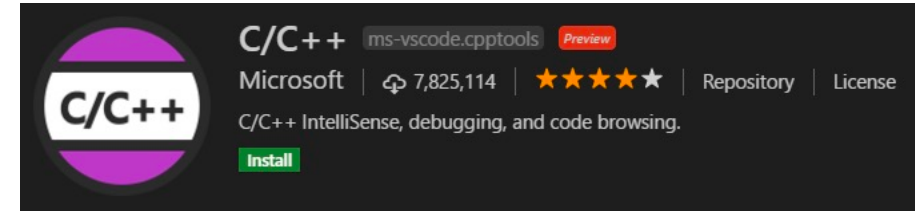
Why vscode?

As usual, everyone was using the [CodeBlocks IDE](#) and [Visual Studio IDE](#). But I was already used to Visual Studio Code

- A lightweight editor
- Versatile: c++, python, html, markdown, latex, ...
- Powerful
- Cross-platform

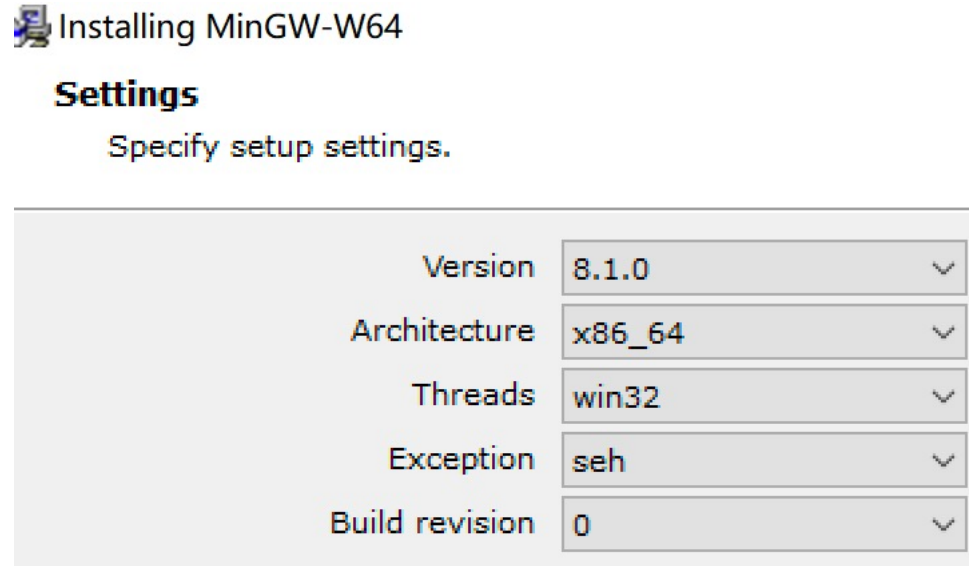
大纲

1. Install IDE: vscode
2. Install c++ compiler
 1. MinGW for windows
 2. Clang for mac
3. Install the "**ms-vscode.cpptools**" extension
4. Install "Code Runner" extension
5. Create hello.cpp and edit it
6. Run it
7. Check the project folder
8. Debug it



Install MinGW for Windows

- <https://code.visualstudio.com/docs/cpp/config-mingw>

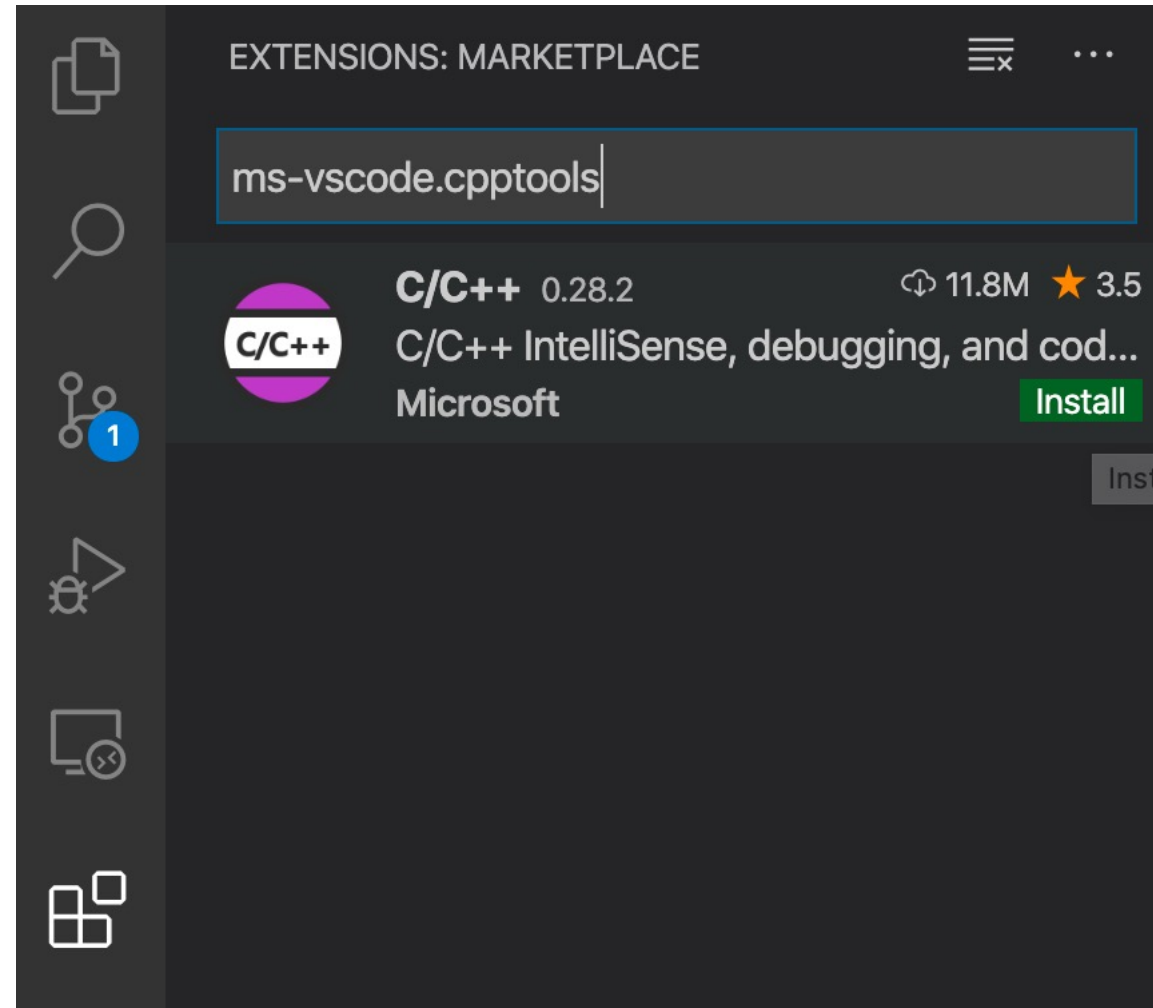


- Maybe you can stop after “Check your MinGW installation”

Install clang for Mac

- `xcode-select --install`

Install the "ms-vscode.cpptools" extension



To create the first program

- select "**File**" > "**New file**". This will open a new file window.



- Save the file ("**File**" > "**Save**") into a new directory.
 - You can name the directory anything you want, but this example will call the directory "**c_labs**" and the file "**hello.cpp**".

Write the actual program

```
#include <iostream>

int main() {
    // Output the hello world text
    std::cout << "Hello world!" << std::endl;
    return 0;
}
```


Run code

- ① Click the triangle at upper right corner
- ② This opens a OUTPUT window in the lower portion of the IDE.

③ Inside this window, we found

① [Running] ...

① `cd "/Users/jjcao/work/courses/c_labs/hello/"`

② `g++ hello.cpp -o hello`

③ `"/Users/jjcao/work/courses/c_labs/hello/"hello`

② Hello world!

③ [Done] ...

```
hello.cpp x
Users > jjcao > work > courses > c_labs > hello > hello.cpp > ... Run Code (^⌘N)
1  #include <iostream>
2  int main() {
3  // Output the hello world text
4  std::cout << "Hello world!" << std::endl;
5  return 0;
6  }
7

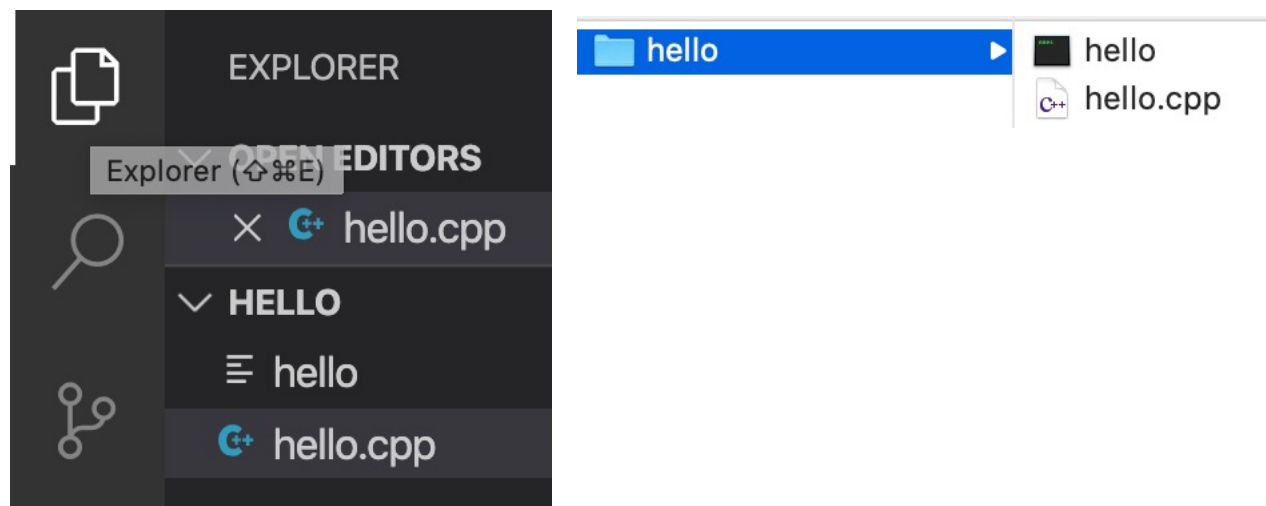
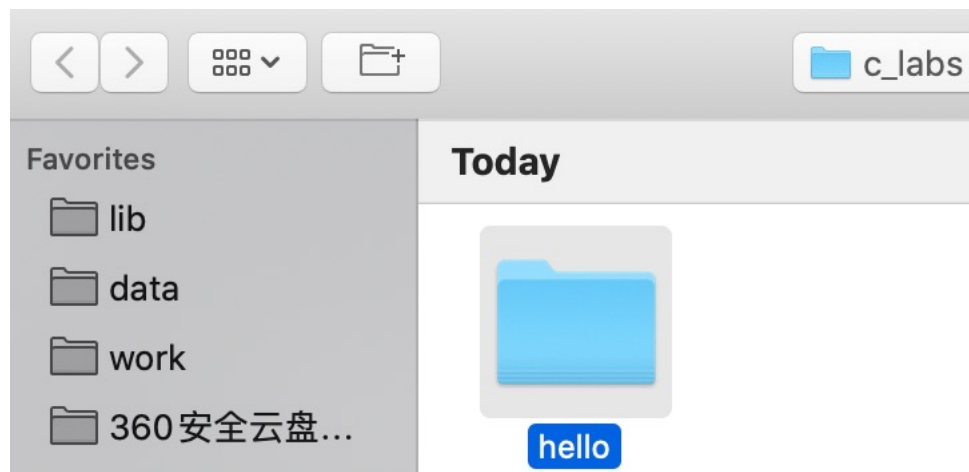
OUTPUT ... Code
[Running] cd "/Users/jjcao/work/courses/c_labs/hello/" && g++ hello.cpp -o hello
Hello world!

[Done] exited with code=0 in 4.74 seconds
```

Congratulations!

What do you have now

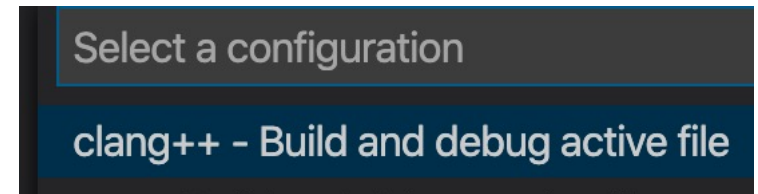
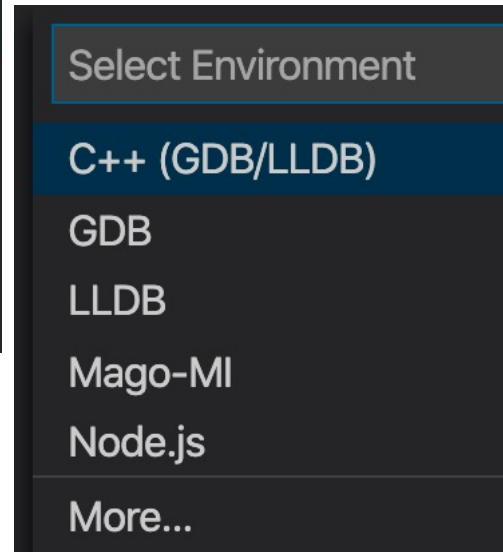
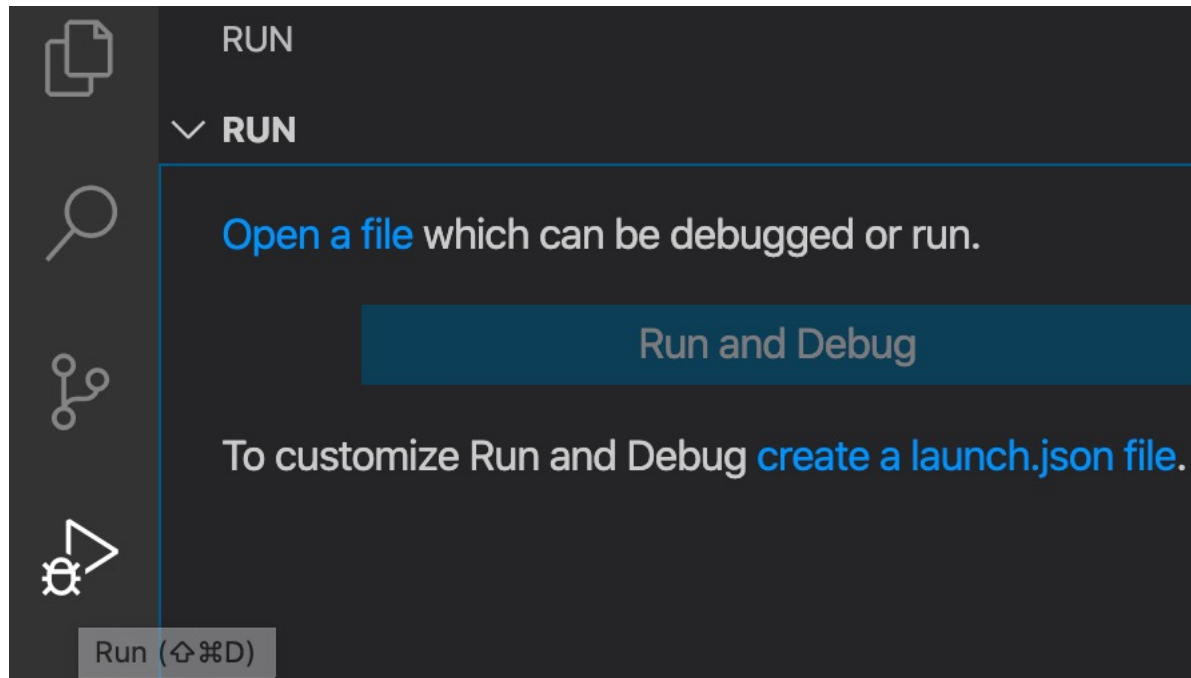
- File/open
 - Open the the project folder
- Click “Explorer”
 - Source file: hello.cpp
 - Executable file: hello or hello.exe



Questions?

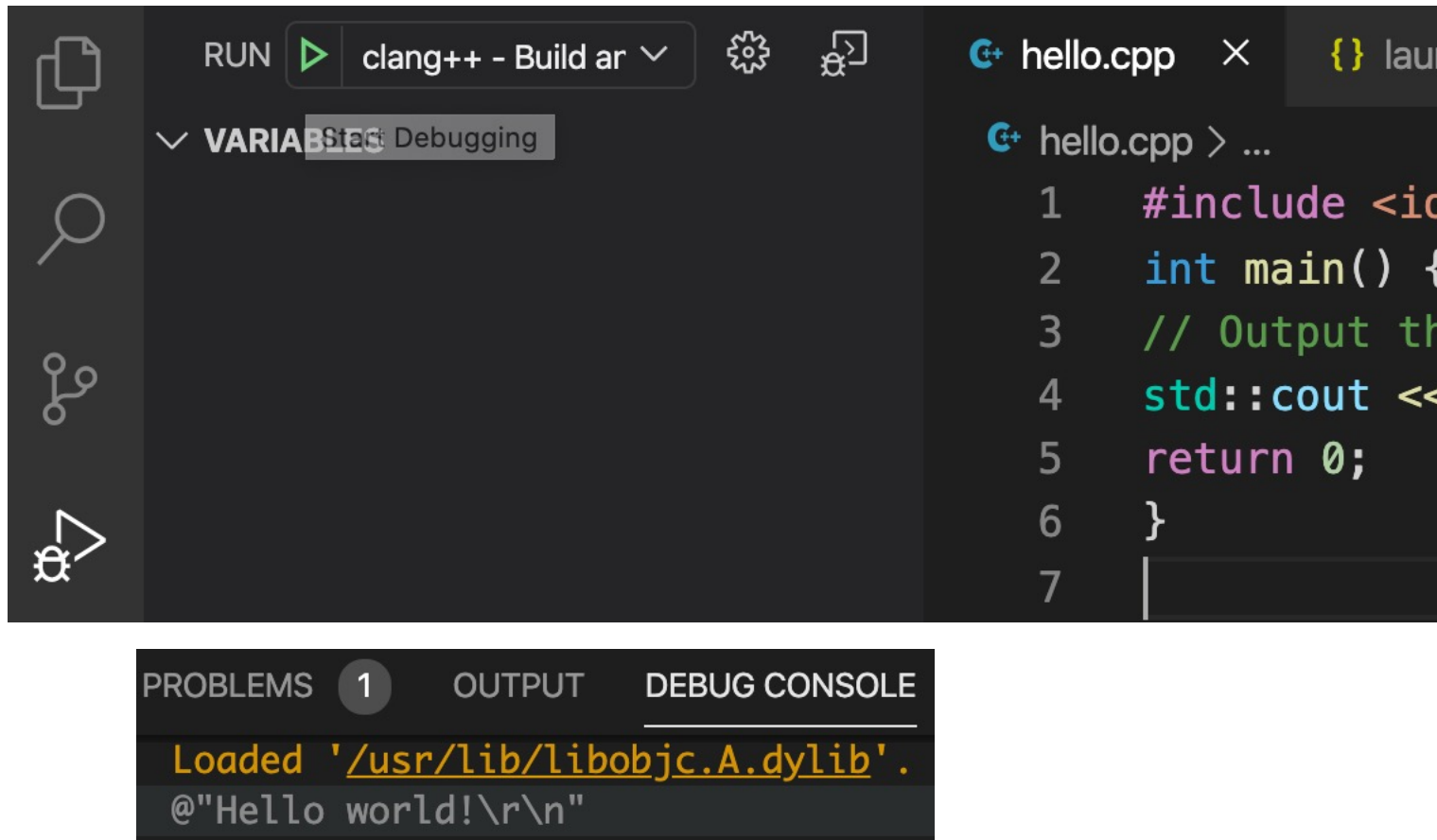
Build and debug active file

- Create a launch.json file
 - C++ (GDB/LLDB)
 - clang++ - Build and debug active file



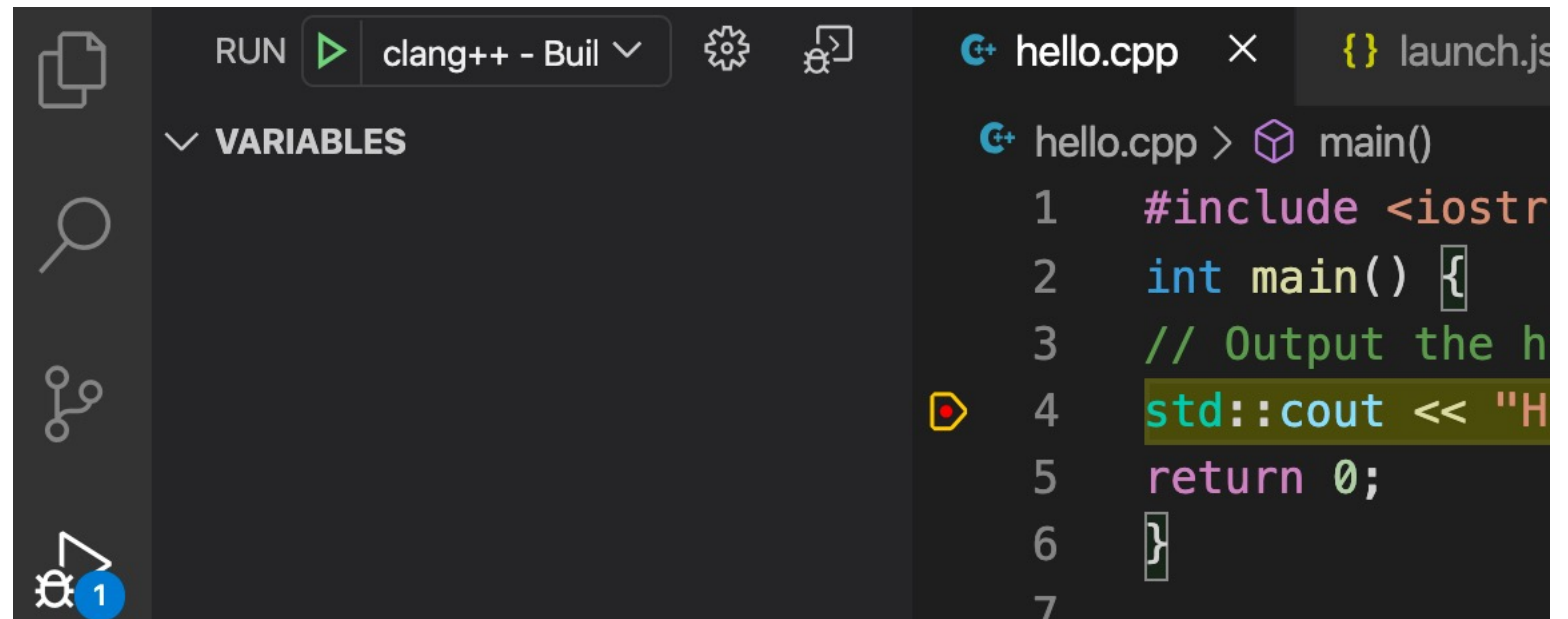
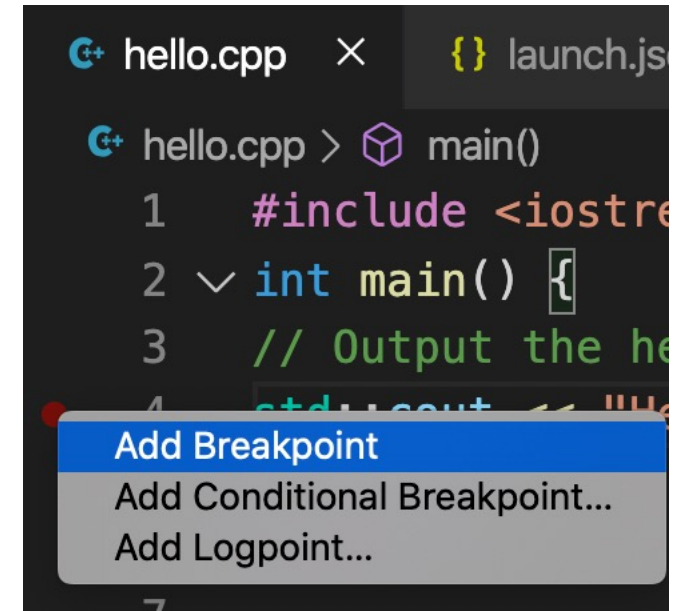
Run it

- Another way without using “Code Runner” extension
- Run (green triangle) with clang++ - Build ...

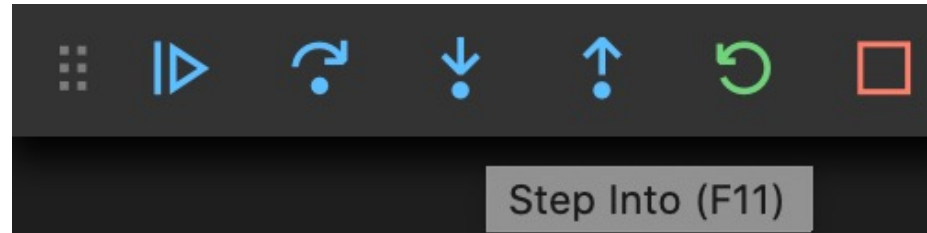


Debug it

- Click a breakpoint
- Run (green triangle) with clang++ - Build ...



Step through the code



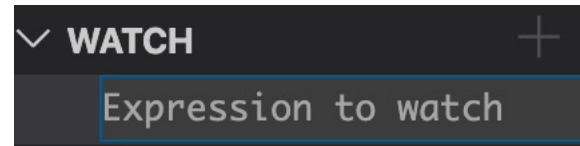
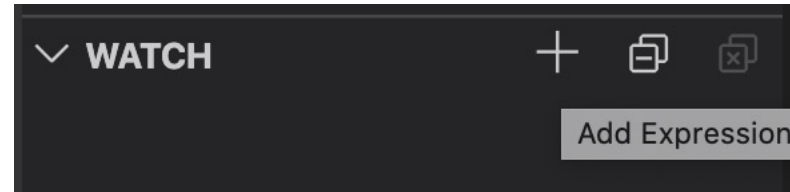
Add variables to the cpp

- Add
 - `int i(-2);`
- Add a breakpoint
- Debut the cpp
 - Then it stopped at line 6

```
1  #include <iostream>
2  int main() {
3  // Output the hello world
4  std::cout << "Hello world
5  int i(-2);
6  return 0;
7  }
```


Set a watch

- Add a watch



Call Stack

```
hello.cpp > test(int)
1  #include <iostream>
2  void test(int i=0)
3  {
4      i = 3;
5  }
6  int main() {
7      // Output the hello
8      std::cout << "Hello\n";
9      int i(-2);
10     test(i);
11     return 0;
```

WATCH

i: 3
> &i: 0x00007ffeefbffd9c

CALL STACK PAUSED ON BREAKPOINT

hello!test(int)	hello.cpp	5:1
hello!main	hello.cpp	10:1

```
hello.cpp > main()
1  #include <iostream>
2  void test(int i=0)
3  {
4      i = 3;
5  }
6  int main() {
7      // Output the hello
8      std::cout << "Hello\n";
9      int i(-2);
10     test(i);
11     return 0;
```

WATCH

i: -2
> &i: 0x00007ffeefbfdb8

CALL STACK PAUSED ON BREAKPOINT

hello!test(int)	hello.cpp	5:1
hello!main	hello.cpp	10:1

Debugging your code



Six Stages of Debugging

1. That can't happen.
2. That doesn't happen on my machine.
3. That shouldn't happen.
4. Why does that happen?
5. Oh, I see.
6. How did that ever work?

Windows debugging with GDB

- [Windows Debugging with MinGW64](#)
- <https://code.visualstudio.com/docs/cpp/config-mingw>

Step 1: Modify Main

```
#include <iostream>
using namespace std;
int main(int argc, char** args)
{
    // Notice I start from i=1 not 0 because the args[0] is reserved
    // for the name of this program.
    for(int i = 1; i < argc; i++)
    {
        cerr << i << "th argument is " << args[i] << "\n";
    }
}
```

Note: The relationship of argc and args:

1. args is **an array of char***
2. argc is the size of the array: args, which is determined when command line arguments are passed to the main() function. So after you change the size of args, argc is not updated automatically.

Launch.json

- “args”: [1, “str”]

```
hello.cpp  {} launch.json ×
vscode > {} launch.json > ...
1  {}
2  // Use IntelliSense to learn about possible attributes.
3  // Hover to view descriptions of existing attributes.
4  // For more information, visit: https://go.microsoft.com/fwlink
5  "version": "0.2.0",
6  "configurations": [
7      {
8          "name": "clang++ - Build and debug active file",
9          "type": "cppdbg",
10         "request": "launch",
11         "program": "${fileDirname}/${fileBasenameNoExtension}",
12         "args": [1, "str"],
13         "stopAtEntry": false,
14         "cwd": "${workspaceFolder}",
15         "environment": [],
16         "externalConsole": false,
17         "MIMode": "lldb",
18         "preLaunchTask": "C/C++: clang++ build active file"
19     }
20 ]
21 }
```

2. Command Line Argument 命令行参数

1. Open a Terminal in Mac.

A screenshot of a macOS Terminal window. The title bar shows three colored window control buttons (red, yellow, green) on the left and a title bar with a home icon, the text 'jjcao — -zsh — 80x24', and a close button on the right. The terminal content shows the last login time and the current prompt: '(base) jjcao@JunjiedeMacBook-Pro-2 ~ %'.

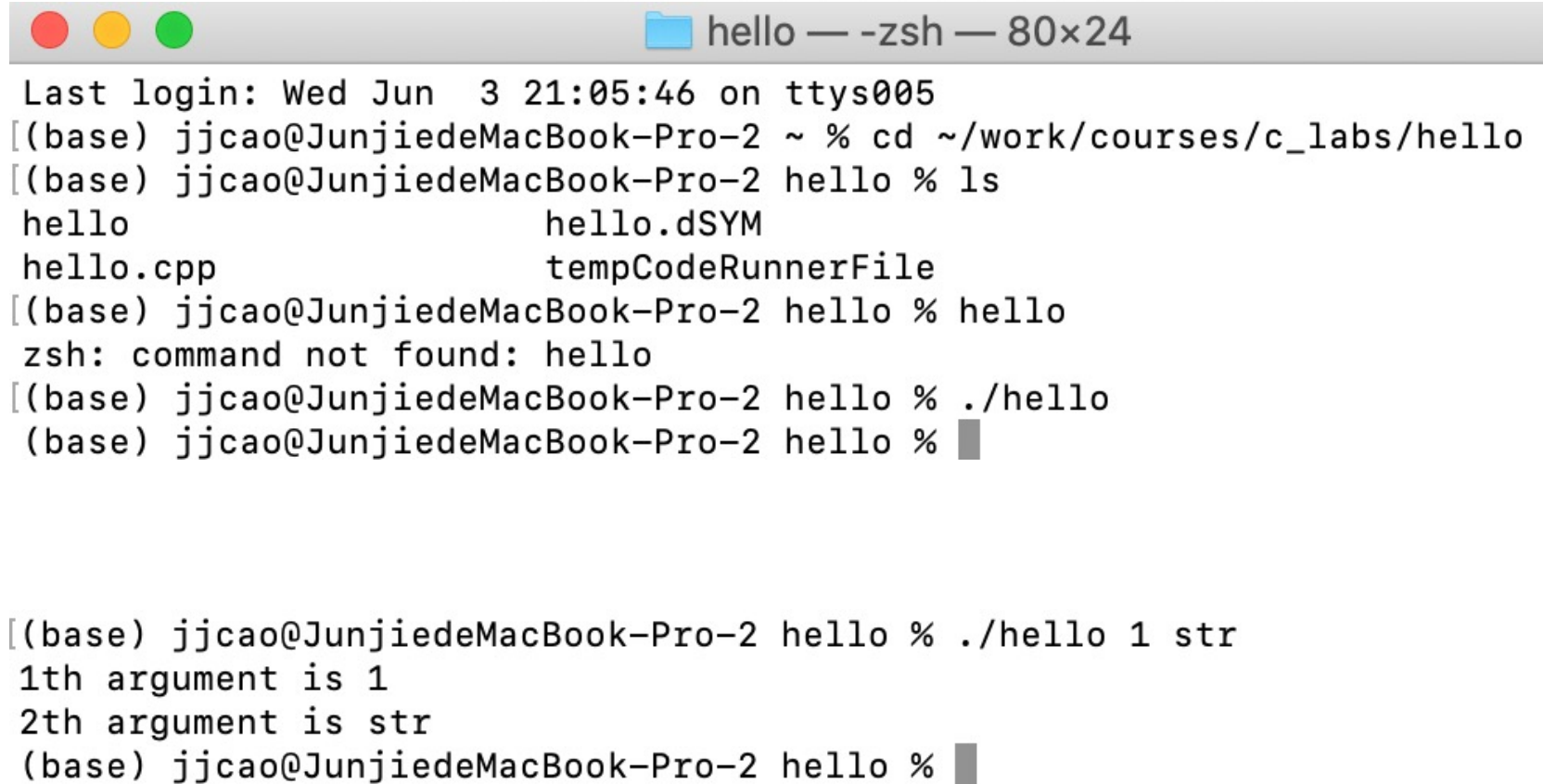
2. cd the project folder

3. run the program: hello

1. hello

2. ./hello

3. ./hello 1 str

A screenshot of a macOS Terminal window. The title bar shows three colored window control buttons (red, yellow, green) on the left and a title bar with a folder icon, the text 'hello — -zsh — 80x24', and a close button on the right. The terminal content shows the last login time, the user navigating to the project folder, listing files, and running the 'hello' program with arguments '1' and 'str'.

```
Last login: Wed Jun  3 21:05:46 on ttys005
(base) jjcao@JunjiedeMacBook-Pro-2 ~ % cd ~/work/courses/c_labs/hello
(base) jjcao@JunjiedeMacBook-Pro-2 hello % ls
hello                                hello.dSYM
hello.cpp                            tempCodeRunnerFile
(base) jjcao@JunjiedeMacBook-Pro-2 hello % hello
zsh: command not found: hello
(base) jjcao@JunjiedeMacBook-Pro-2 hello % ./hello
(base) jjcao@JunjiedeMacBook-Pro-2 hello %
[
(base) jjcao@JunjiedeMacBook-Pro-2 hello % ./hello 1 str
1th argument is 1
2th argument is str
(base) jjcao@JunjiedeMacBook-Pro-2 hello %
```

Questions

User Input

- Let user input values to the program (line 6)

- ctrl+z: cancel input from cin

```
1  #include <iostream>
2  using namespace std;
3
4  int main() {
5      int x;
6      cin >> x;
7
8      cout << x / 3 << ' ' << x * 2;
9
10     return 0;
11 }
```

iostream

- cin
- cout
- cerr
- clog
- Ordinarily, sys associates them with the console window.
- They can be redirected to files.

Thanks