

```
// Michigan Robotics 102
// Introduction to AI and Programming

// Making our first program in C++:
//   code, compile, run, repeat
```

```
#include <iostream>
int main()
{
```

```
    std::cout << "Hello World!";
}
```

# A simple program in C++

```
#include <iostream>
int main()
{
    std::cout << "Hello World!";
}
```

# A simple program in C++

## *What is C++?*

```
#include <iostream>
int main()
{
    std::cout << "Hello World!";
}
```

# C++

From Wikipedia, the free encyclopedia

"CXX" redirects here. For other uses, see *CXX* (disambiguation).

**C++** (/ˌskɪpləˈplʌs/) is a general-purpose programming language created by Bjarne Stroustrup as an extension of the C programming language, or "C with Classes". The language has expanded significantly over time, and modern C++ now has object-oriented, generic, and functional features in addition to facilities for low-level memory manipulation. It is almost always implemented as a compiled language, and many vendors provide C++ compilers, including the Free Software Foundation, LLVM, Microsoft, Intel, Oracle, and IBM, so it is available on many platforms.<sup>[9]</sup>

C++ was designed with an orientation toward system programming and embedded, resource-constrained software and large systems, with performance, efficiency, and flexibility of use as its design highlights.<sup>[10]</sup> C++ has also been found useful in many other contexts, with key strengths being software infrastructure and resource-constrained applications,<sup>[10]</sup> including desktop applications, video games, servers (e.g. e-commerce, web search, or databases), and performance-critical applications (e.g. telephone switches or space probes).<sup>[11]</sup>

C++ is standardized by the International Organization for Standardization (ISO), with the latest standard version ratified and published by ISO in December 2020 as *ISO/IEC 14882:2020* (informally known as C++20).<sup>[12]</sup> The C++ programming language was initially standardized in 1998 as *ISO/IEC 14882:1998*, which was then amended by the C++03, C++11, C++14, and C++17 standards. The current C++20 standard supersedes these with new features and an enlarged standard library. Before the initial standardization in 1998, C++ was developed by Danish computer scientist Bjarne Stroustrup at Bell Labs since 1979 as an extension of the C language; he wanted an efficient and flexible language similar to C that also provided high-level features for program organization.<sup>[13]</sup> Since 2012, C++ has been on a three-year release schedule<sup>[14]</sup> with C++23 as the next planned standard.<sup>[15]</sup>

## Contents [hide]

- 1 History
  - 1.1 Etymology
  - 1.2 Philosophy

# What is C++?

## C++



The C++ logo endorsed by Standard C++

<b>Paradigms</b>	Multi-paradigm: procedural, functional, object-oriented, generic, modular
<b>Family</b>	C
<b>Designed by</b>	Bjarne Stroustrup
<b>Developer</b>	ISO/IEC JTC1 (Joint Technical Committee 1) / SC22 (Subcommittee 22) / WG21 (Working Group 21)
<b>First appeared</b>	1985; 36 years ago
<b>Stable release</b>	C++20 (ISO/IEC 14882:2020) / 15 December 2020; 8 months ago
<b>Preview release</b>	C++23 <span><span><span></span></span></span> / 18 June 2021; 2 months ago

# What is C++?

High-level programming language



# What is C++?

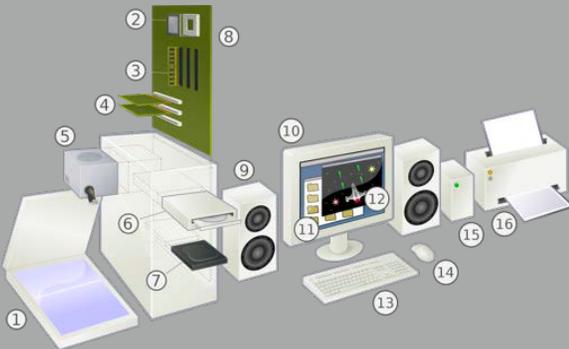
High-level programming language



Computing Hardware

# What is C++?

High-level programming language



**Computing Hardware**

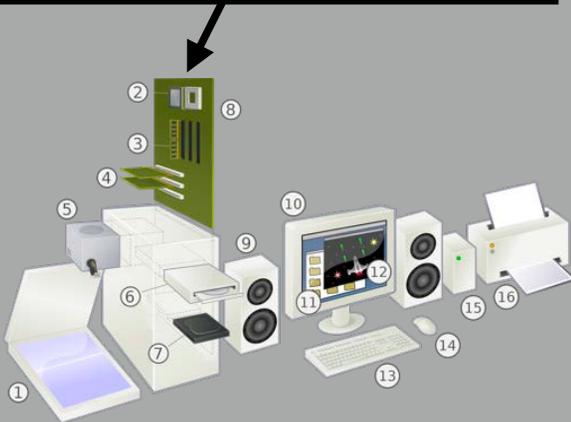
# What is C++?



High-level programming language



Main circuit board  
or “*Motherboard*”

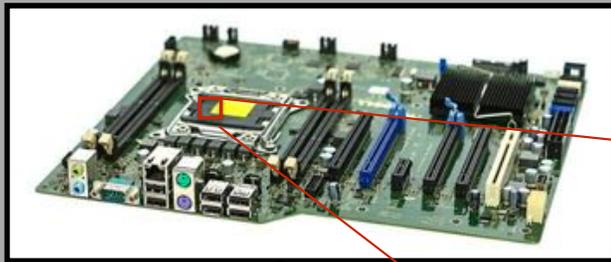


Computing Hardware

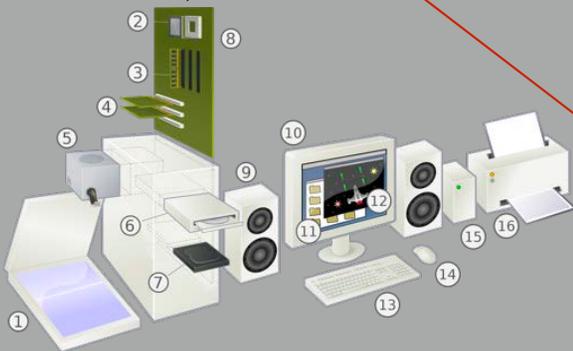
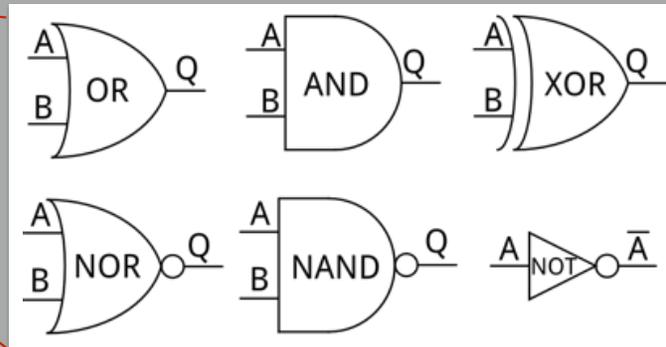
# What is C++?



High-level programming language



Electronics that can carry out digital logic



Computing Hardware

# What is C++?

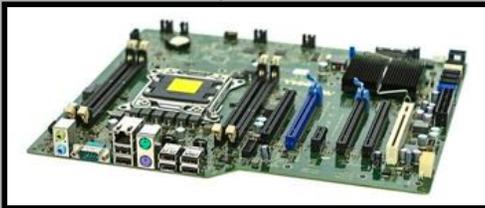


High-level programming language

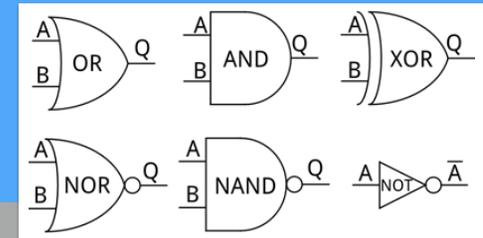


A program in binary code

**Executable Software**



**Computing Hardware**



Electronics that can carry out digital logic

# What is C++?

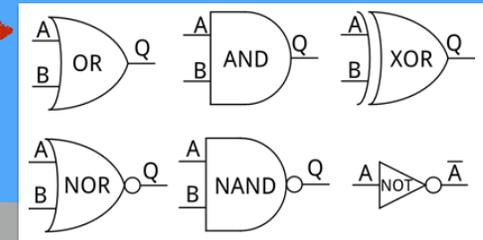


## High-level programming language

A program in binary code specifying a sequence of digital logic instructions that run on a computer

```
11101110
01100100
11001001
11101110
01100100
```

**Executable Software**



Electronics that can carry out digital logic

**Computing Hardware**



# What is C++?

## High-level programming language

```
#include <iostream>
int main()
{
    std::cout << "Hello World!";
}
```

**Source Code**

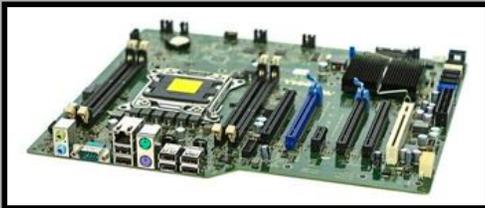
A program in readable text that can be compiled into a software executable



**Executable Software**

A program in binary code specifying a sequence of digital logic instructions that run on a computer

```
11101110
01100100
11001001
11101110
01100100
```



**Computing Hardware**

Electronics that can carry out digital logic

# What is C++?

High **"Coding"** is writing source code

```
#include <iostream>
int main()
{
    std::cout << "Hello World!";
}
```

**Source Code**

A program in readable text that can be compiled into a software executable



**Executable Software**

A program in binary code specifying a sequence of digital logic instructions that run on a computer



**Computing Hardware**

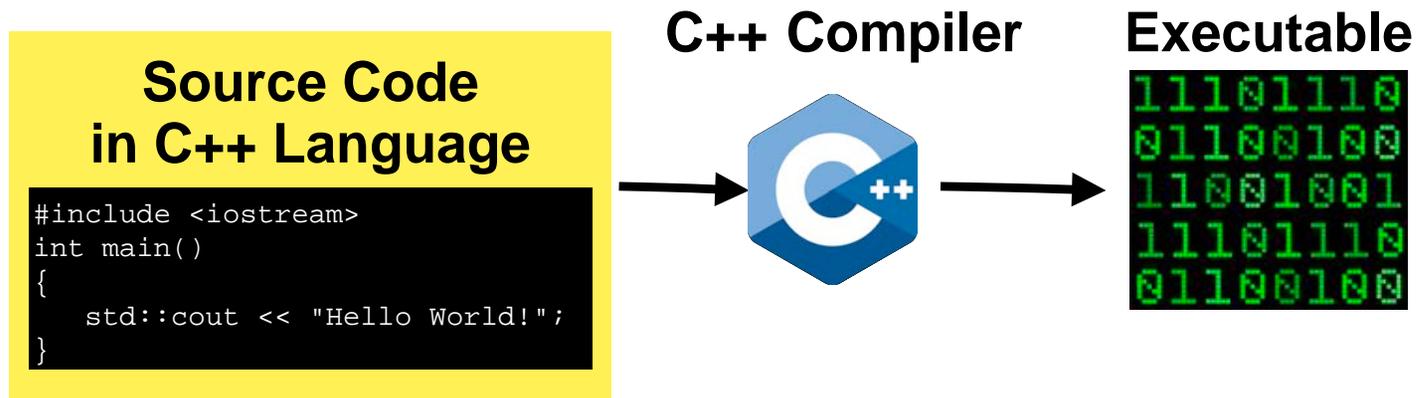
Electronics that can carry out digital logic



# What is C++?

High-level programming language

*Compiles* source files into executable apps



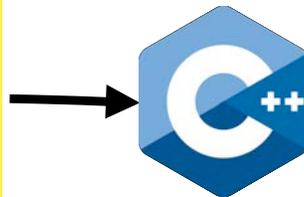
# What is C++?

High-level programming language

*Portable* across nearly all computers



C++ Compiler



Source Code  
in C++ Language

```
#include <iostream>
int main()
{
    std::cout << "Hello World!";
}
```

# What is C++?

High-level programming language

*Portable* across nearly all computers



```
Source Code  
in C++ Language
```

```
#include <iostream>  
int main()  
{  
    std::cout << "Hello World!";  
}
```

Compiler



# What is C++?

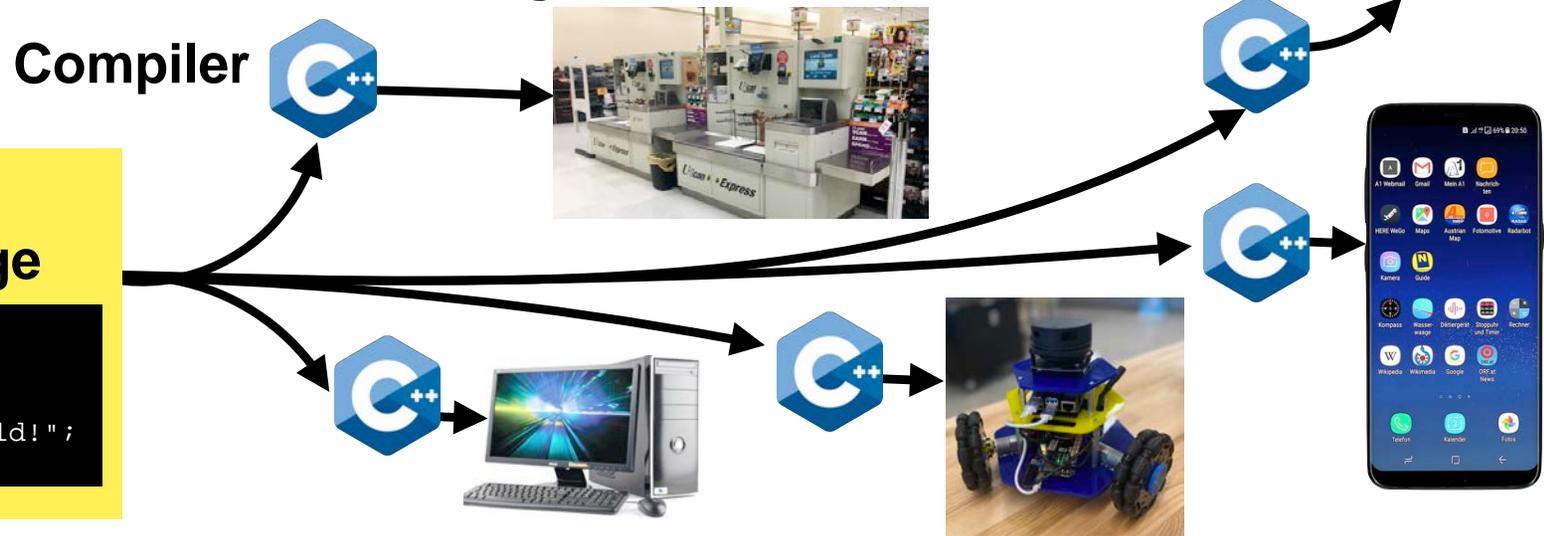
High-level programming language

*Portable* across nearly all computers



```
Source Code  
in C++ Language
```

```
#include <iostream>  
int main()  
{  
    std::cout << "Hello World!";  
}
```



# What is C++?

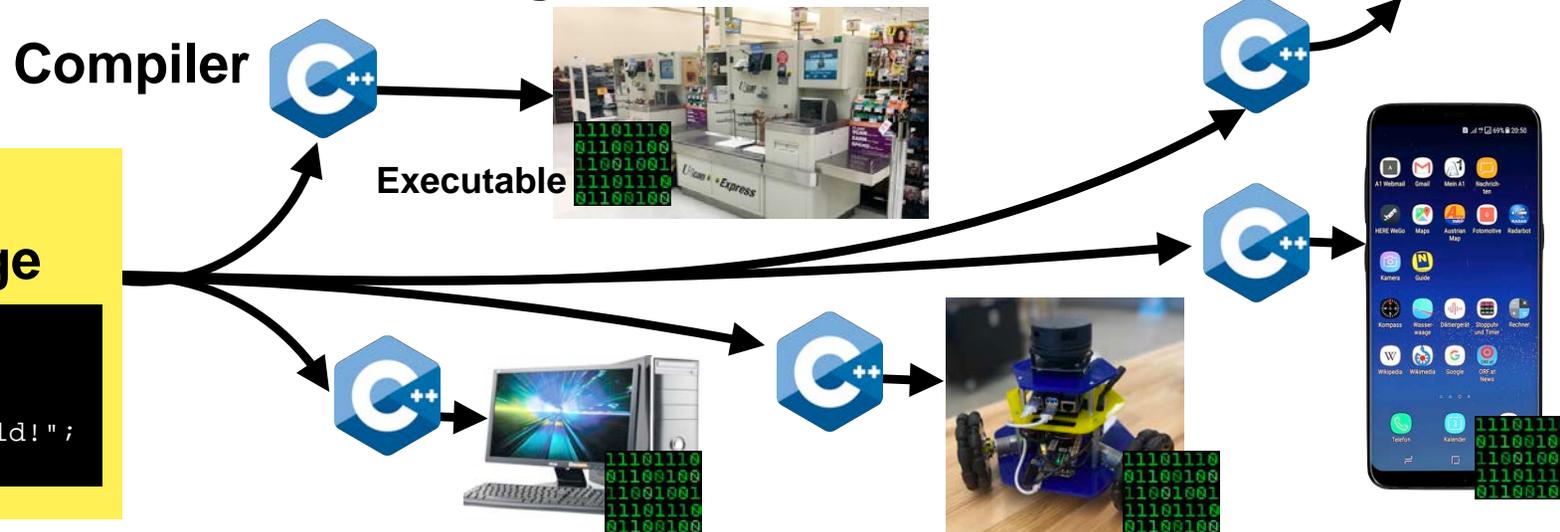
High-level programming language

*Portable* across nearly all computers



```
Source Code  
in C++ Language
```

```
#include <iostream>  
int main()  
{  
    std::cout << "Hello World!";  
}
```



# Software Portability



**It runs Doom!**



it runs doom



All Images Videos News Shopping More

Tools

Collections Settings



It Runs Doom | Know Your Meme  
knowyourmeme.com



Programmer Plays Doom on a Pregnancy ...  
gophermechanics.com



DOOM will run on literally anything ...  
gamezebo.com



run the game 'Doom' ...  
inputmag.com



It Runs Doom! ...  
shredbooth.tumblr.com



The Treadmill at my school runs D...  
reddit.com



Nintendo's Game & Watch can run Doom...  
theverge.com



Chip From a \$15 Ikea Smart Lamp  
uk.pcmag.com



It Runs Doom | Know Your Meme  
knowyourmeme.com



Devices That Can Somehow Run 'Doom' ...  
vice.com



It Runs Doom | Rip It Apart - Jason's ...  
ripitapart.com



It Runs Doom! ...  
shredbooth.tumblr.com



DOOM on a Digital Camera from 1998 ...  
youtube.com

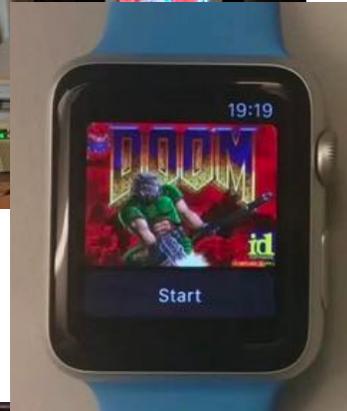
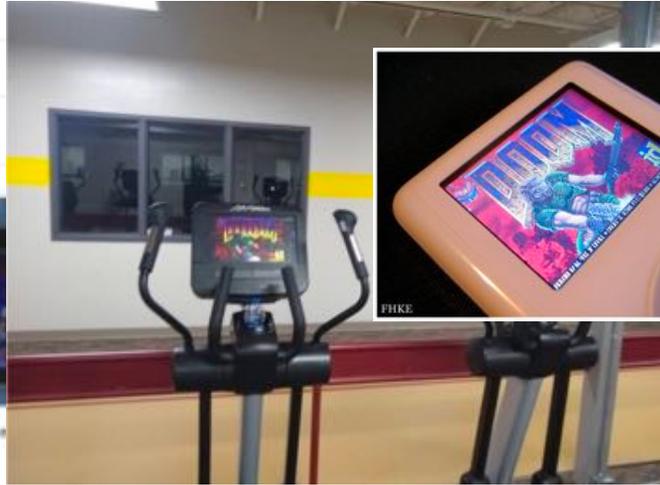


Yes, DOOM Can Run On A Pregnancy Test ...  
gamezebo.com



If it has a processor, ...  
reddit.com

Google it runs doom



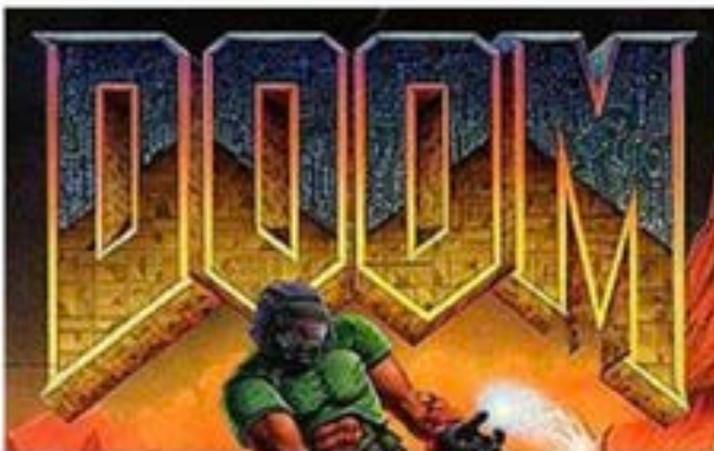
## Neil's Place / DoomPhone

Technical Review: How I hacked an office telephone to play DOOM

Published 2017-08-03

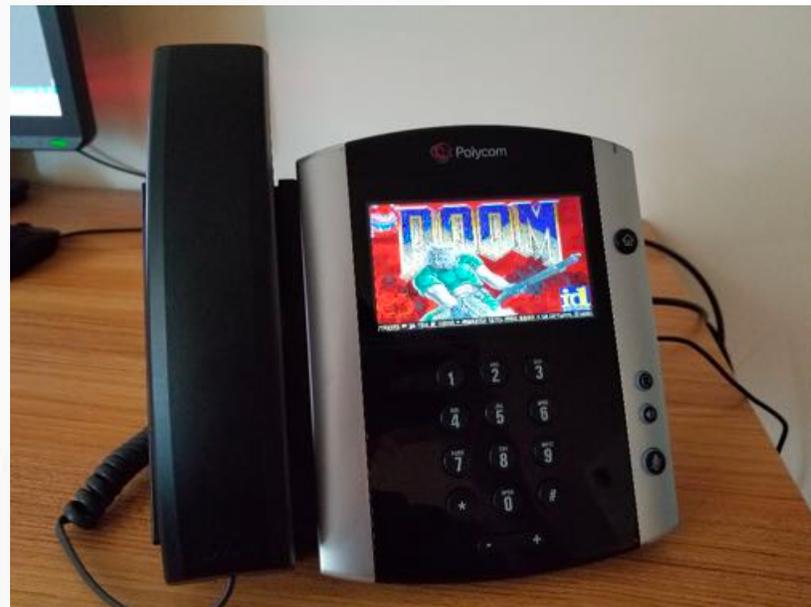
### Contents

- 1. Background
- 2. Hacking the phone
- 3. Make it run DOOM
  - Display.Driver
  - Keypad.Driver
  - Compiling DOOM
- 4. Conclusion



### Background

In late 2017 I was a new hire on a small (maybe 10 person) IT team. We generally had an option to take hardware for ourselves if



# What is C++?

High-level programming language

*Portable* across nearly all computers

*Compiles* source files into executable apps



# What is C++?

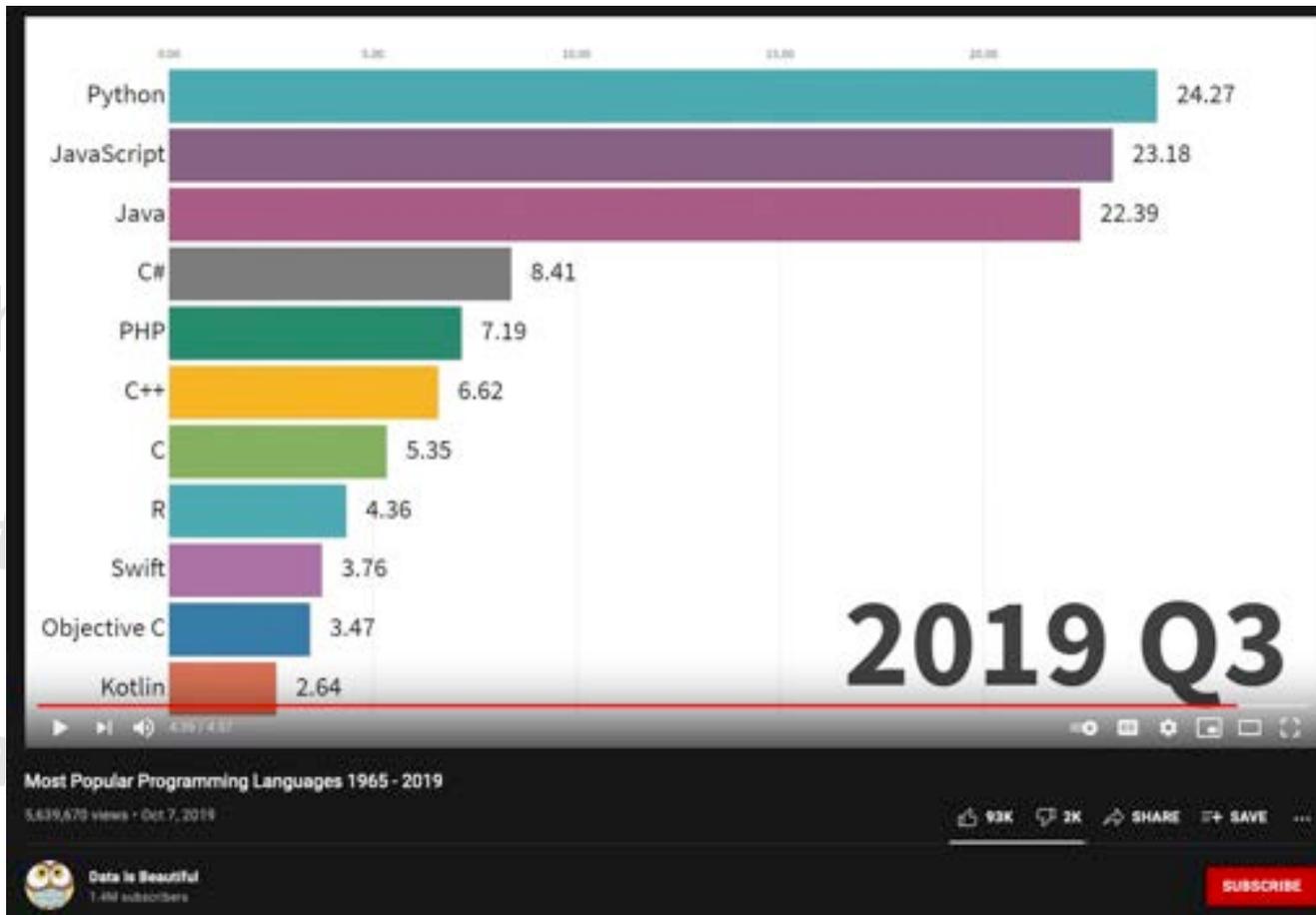
High-level programming language

*Portable* across nearly all computers

*Compiles* source files into executable apps

One of the most popular languages





One of the most popular languages



*You code in C++.  
You can learn any of these.*

One of the most popular languages

# A simple program in C++

*How to code in C++ ?*

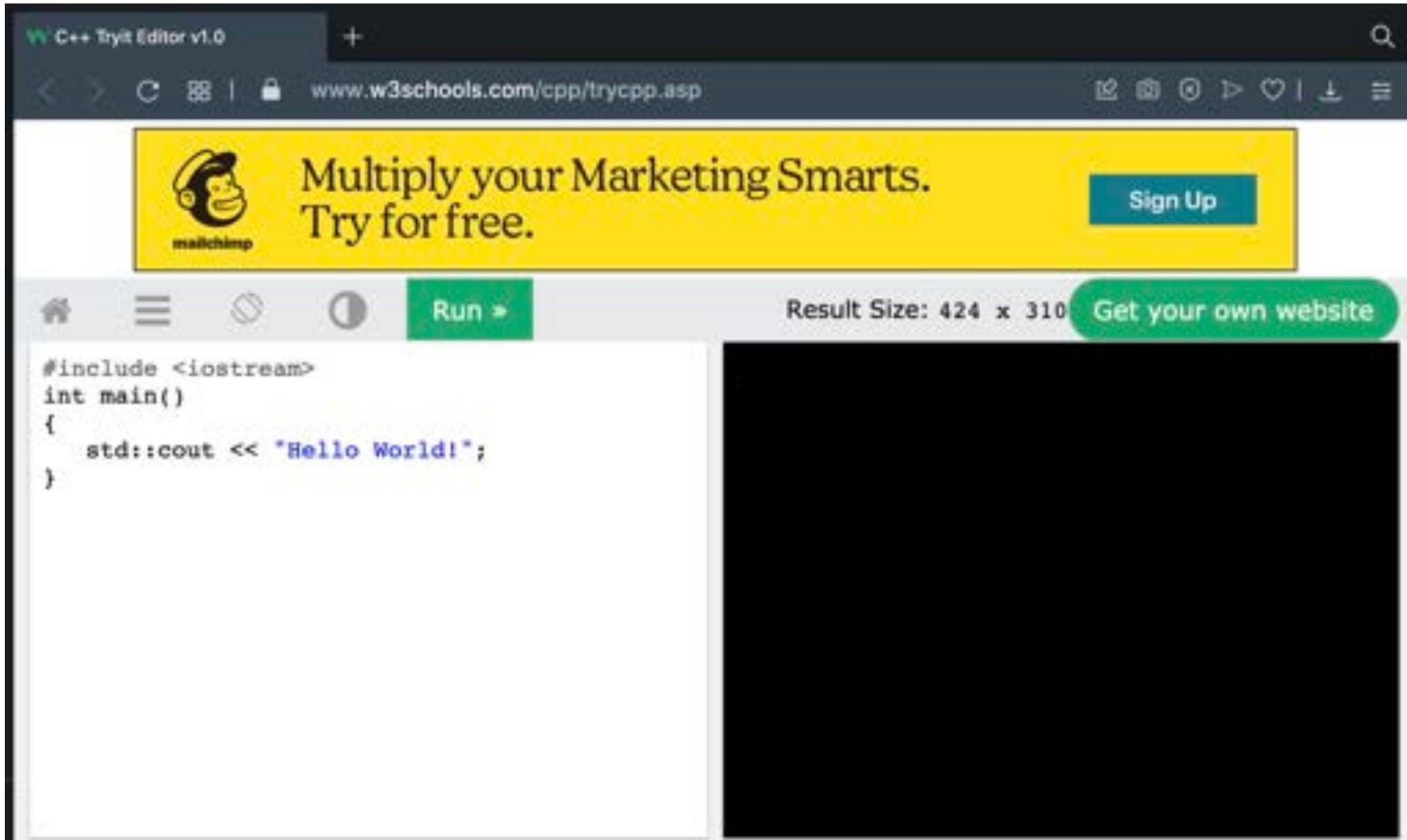
```
#include <iostream>
int main()
{
    std::cout << "Hello World!";
}
```

# *How to code in C++ ?*

## Step 1: Type code into a file

```
#include <iostream>
int main()
{
    std::cout << "Hello World!";
}
```

# One option: online C++ editors

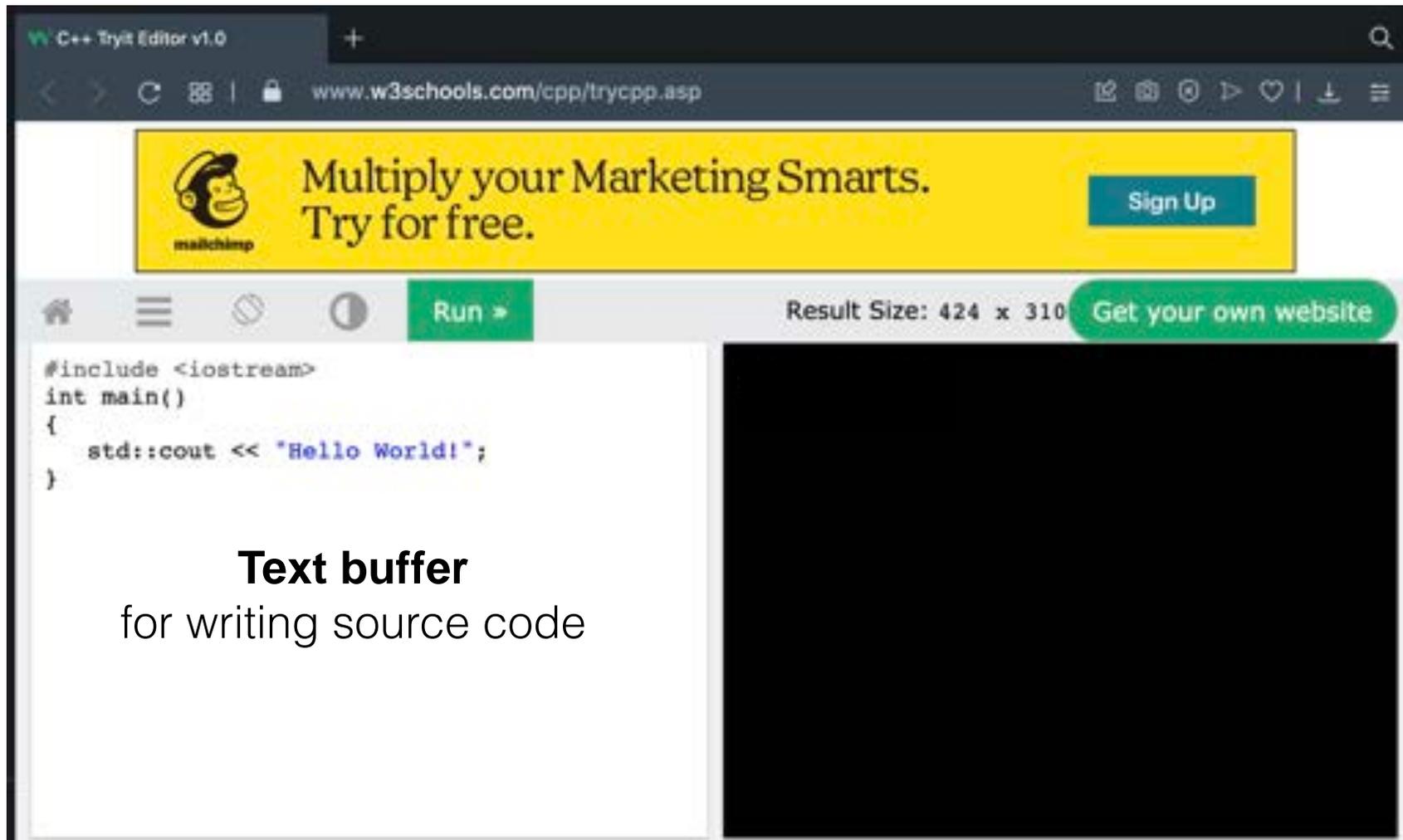


The screenshot shows a web browser window with the address bar displaying `www.w3schools.com/cpp/trycpp.asp`. The page title is "C++ Tryit Editor v1.0". A yellow banner at the top contains the Matchimp logo and the text "Multiply your Marketing Smarts. Try for free." with a "Sign Up" button. Below the banner is a toolbar with a "Run" button and a "Result Size: 424 x 310" indicator. The main editor area contains the following C++ code:

```
#include <iostream>
int main()
{
    std::cout << "Hello World!";
}
```

The output window on the right is currently black, indicating that the program has not yet been executed.

# One option: online C++ editors



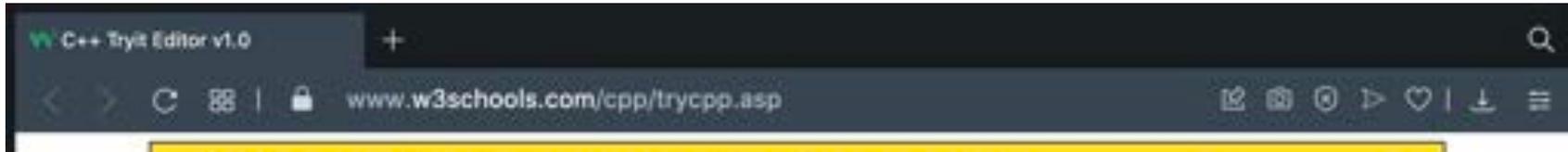
The screenshot shows a web browser window with the address bar displaying `www.w3schools.com/cpp/trycpp.asp`. The page features a yellow banner for Matchimp with the text "Multiply your Marketing Smarts. Try for free." and a "Sign Up" button. Below the banner is a toolbar with a "Run" button and a "Result Size: 424 x 310" indicator. The code editor contains the following C++ code:

```
#include <iostream>
int main()
{
    std::cout << "Hello World!";
}
```

The terminal output area is currently blank. A green button labeled "Get your own website" is visible in the top right corner of the editor interface.

**Text buffer**  
for writing source code

# One option: online C++ editors

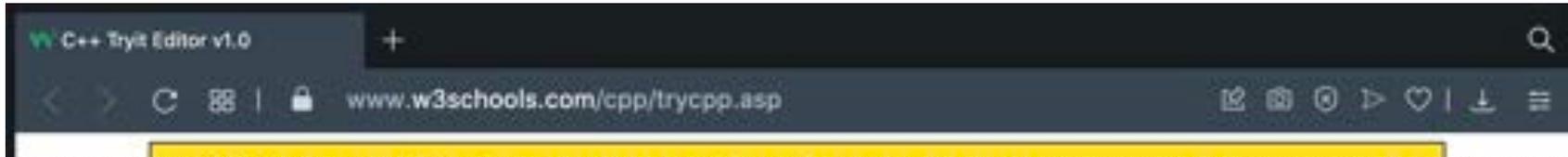


**Press “Run”**

to compile source code into executable program, and then run the executable

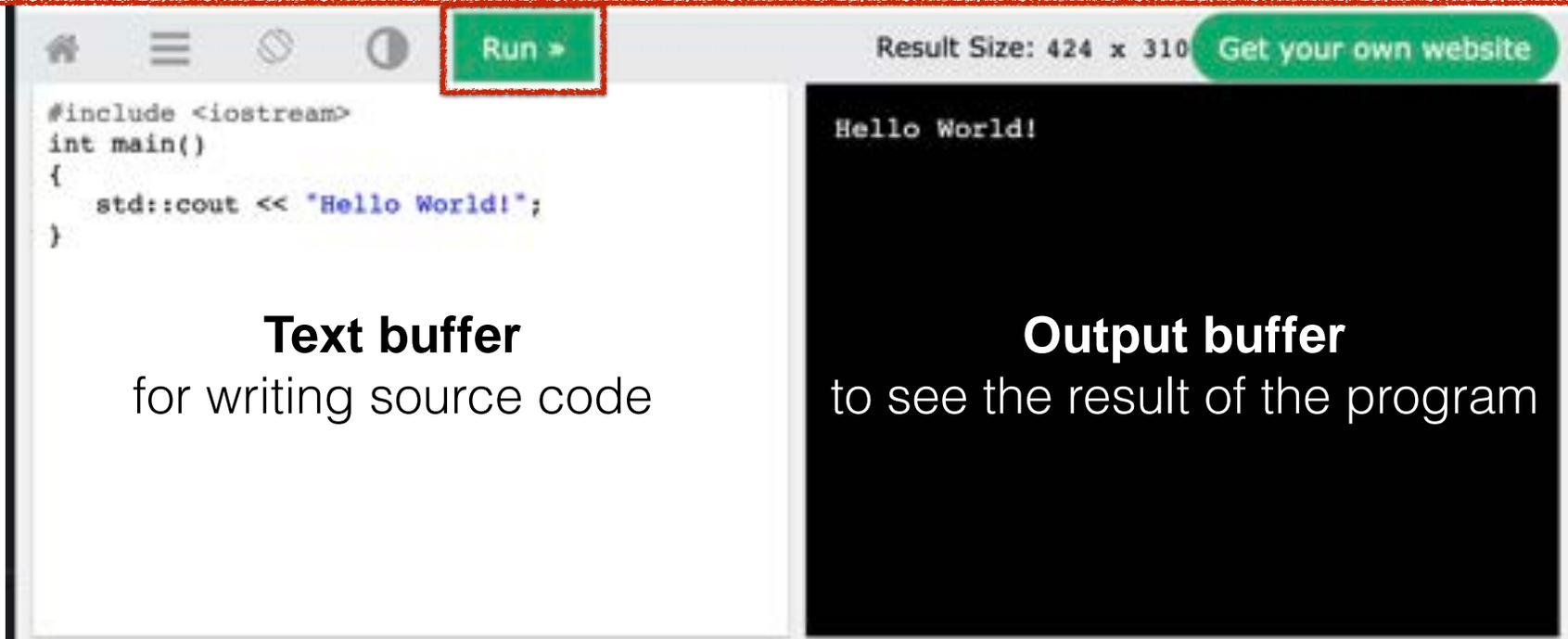


# One option: online C++ editors



**Press “Run”**

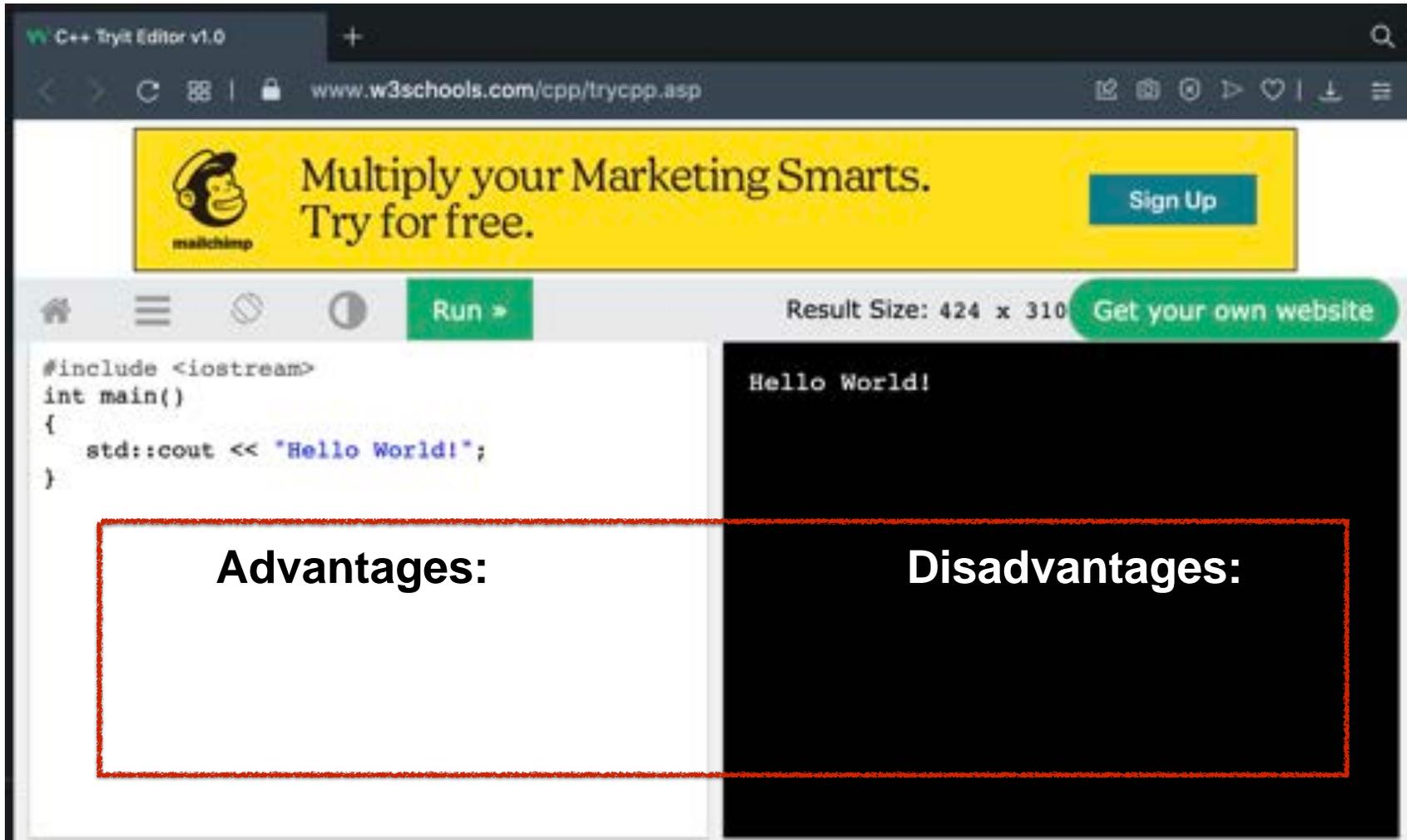
to compile source code into executable program, and then run the executable



**Text buffer**  
for writing source code

**Output buffer**  
to see the result of the program

# One option: online C++ editors

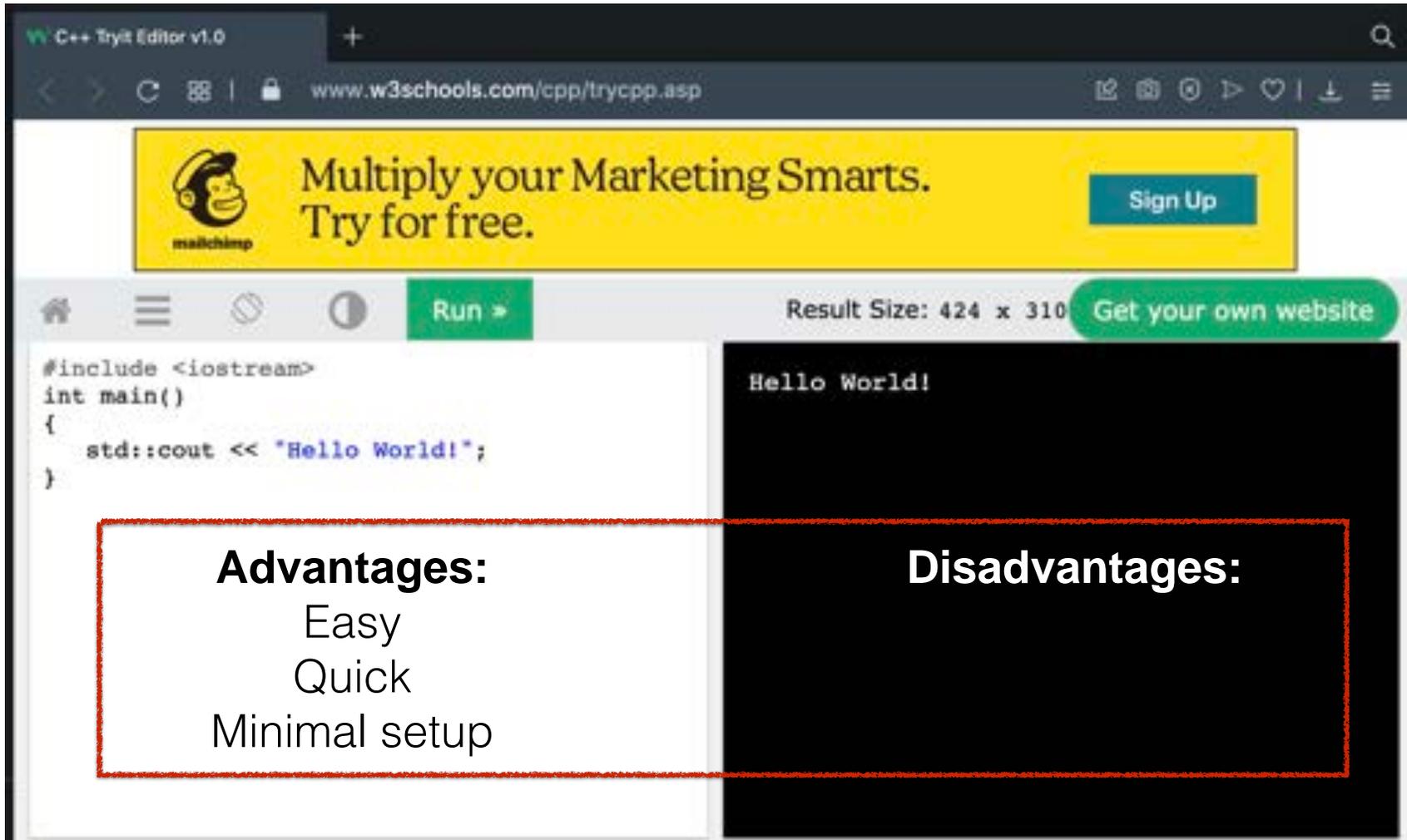


The screenshot shows a web browser window with the address bar displaying `www.w3schools.com/cpp/trycpp.asp`. The page features a yellow banner for Matchimp with the text "Multiply your Marketing Smarts. Try for free." and a "Sign Up" button. Below the banner is a toolbar with a "Run" button and a "Result Size: 424 x 310" indicator. The main area is split into two panels: a code editor on the left and an output window on the right. The code editor contains the following C++ code:

```
#include <iostream>
int main()
{
    std::cout << "Hello World!";
}
```

The output window displays "Hello World!". A red dashed box is overlaid on the bottom half of the image, containing the text "Advantages:" on the left and "Disadvantages:" on the right.

# One option: online C++ editors



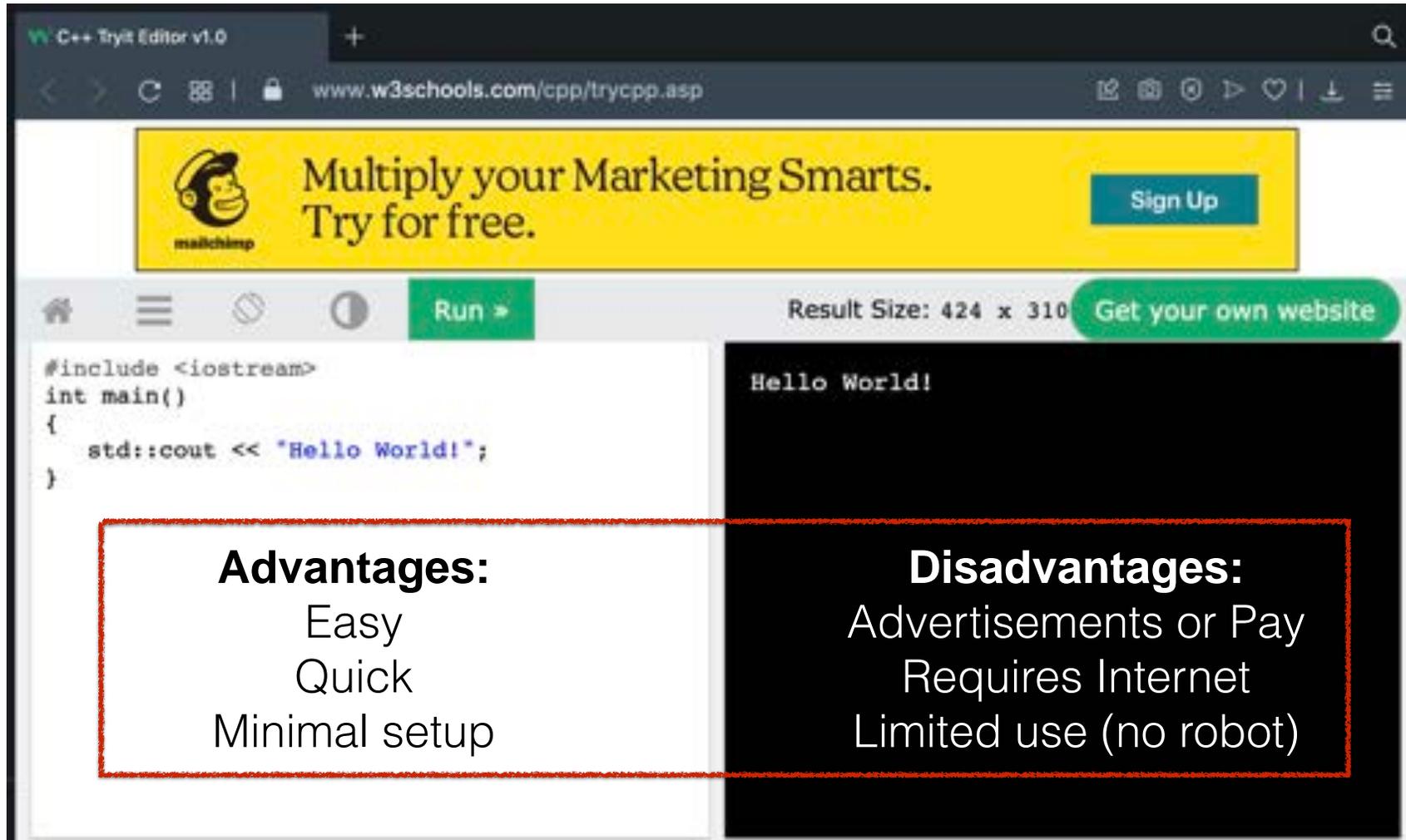
The screenshot shows a web browser window with the address bar displaying `www.w3schools.com/cpp/trycpp.asp`. The page features a yellow banner for Matchimp with the text "Multiply your Marketing Smarts. Try for free." and a "Sign Up" button. Below the banner is a toolbar with a "Run" button and a "Result Size: 424 x 310" indicator. The main area is split into two panes: the left pane contains C++ code, and the right pane shows the output "Hello World!".

```
#include <iostream>
int main()
{
    std::cout << "Hello World!";
}
```

**Advantages:**  
Easy  
Quick  
Minimal setup

**Disadvantages:**

# One option: online C++ editors



The screenshot shows a web browser window with the URL `www.w3schools.com/cpp/trycpp.asp`. The page features a yellow banner for 'Matchimp' with the text 'Multiply your Marketing Smarts. Try for free.' and a 'Sign Up' button. Below the banner is a toolbar with a 'Run' button and a 'Result Size: 424 x 310' indicator. The main area is split into two panes: the left pane contains C++ code for a 'Hello World' program, and the right pane shows the output 'Hello World!'. Two red dashed boxes are overlaid on the image, one on the left pane and one on the right pane, containing text about advantages and disadvantages.

```
#include <iostream>
int main()
{
    std::cout << "Hello World!";
}
```

**Advantages:**  
Easy  
Quick  
Minimal setup

**Disadvantages:**  
Advertisements or Pay  
Requires Internet  
Limited use (no robot)

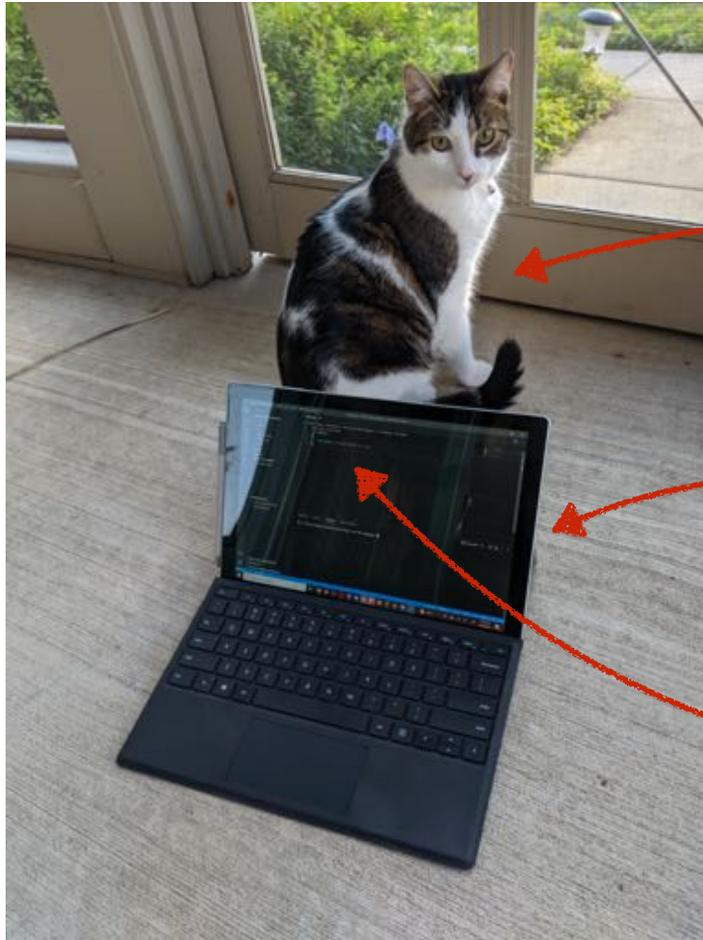
# How to code in C++ ?

## Step 1: Type code into a file

```
#include <iostream>
int main()
{
    std::cout << "Hello World!";
}
```

hello.cpp

# Another option: coding in local files



**Catari 2600**

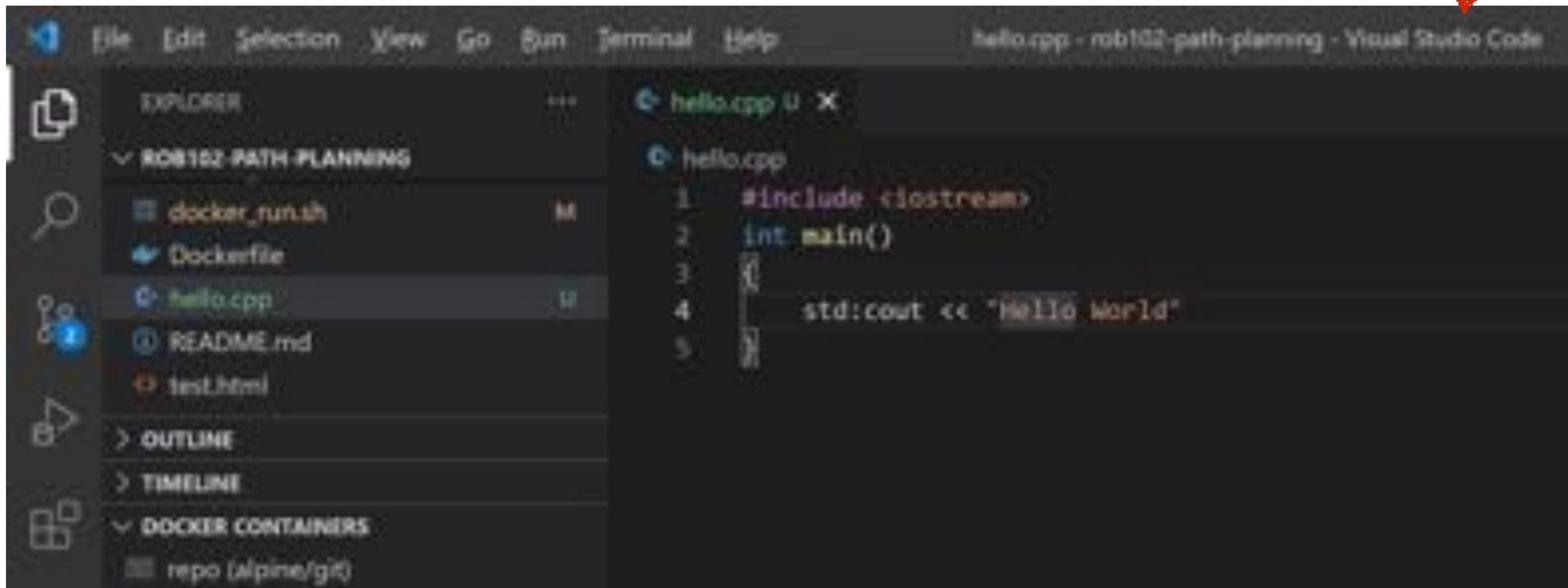
**Microsoft Surface running  
Visual Studio Code (VS Code)**

```
#include <iostream>
int main()
{
    std::cout << "Hello World!";
}
```

**hello.cpp**

# Step 1: Type code into a file

hello.cpp in VS Code 



```
File Edit Selection View Go Run Terminal Help
hello.cpp - rob102-path-planning - Visual Studio Code

EXPLORER
ROB102-PATH-PLANNING
  docker_run.sh
  Dockerfile
  hello.cpp
  README.md
  test.html
OUTLINE
TIMELINE
DOCKER CONTAINERS
  repo (Alpine/git)

hello.cpp
1 #include <iostream>
2 int main()
3 {
4     std::cout << "Hello World"
5 }
```

# Step 1: Type code into a file

## Text editor

Make changes to C++ code

A screenshot of a text editor window titled 'hello.cpp'. The window shows the following C++ code:

```
C:\Users\logan\Documents\Directory for My Programs > hello.cpp > main()
1  #include <iostream>
2  int main()
3  {
4      std::cout << "Hello World";
5  }
```

**hello.cpp**

# Step 1: Type code into a file

## Text editor

Make changes to C++ code

```
#include <iostream>
int main()
{
    std::cout << "Hello World!";
}
```

**hello.cpp**

# Step 1: Type code into a file

## Text editor

Make changes to C++ code

```
#include <iostream>
int main()
{
    std::cout << "Hello World!";
}
```

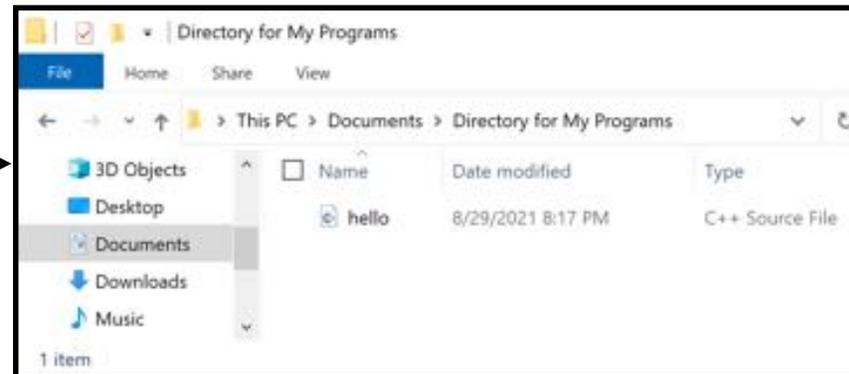
hello.cpp

*OPEN FILE*

*SAVE FILE*

## Source code

C++ files on computer file system containing your code



# Step 1: Type code into a file

## What is a filesystem?

### Text editor

Make changes to C++ code

```
#include <iostream>
int main()
{
    std::cout << "Hello World!";
}
```

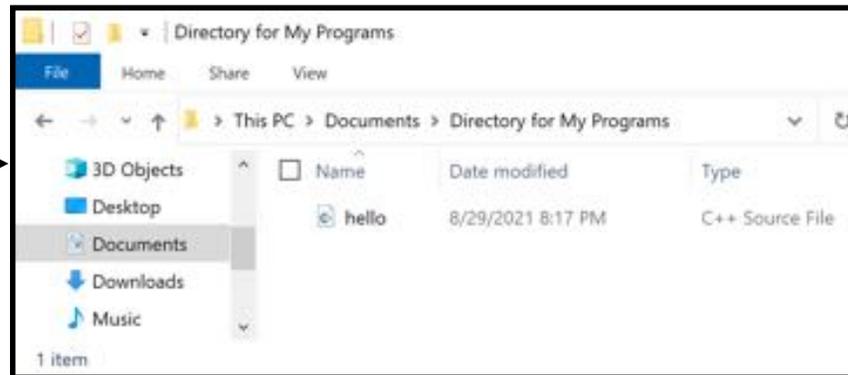
hello.cpp

OPEN FILE

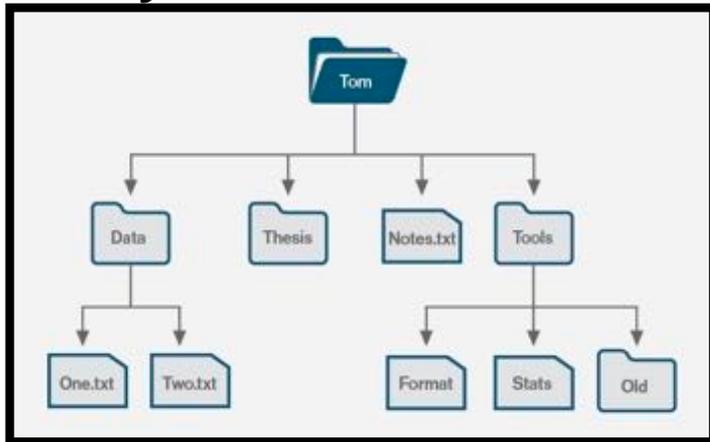
SAVE FILE

### Source code

C++ files on computer file system containing your code

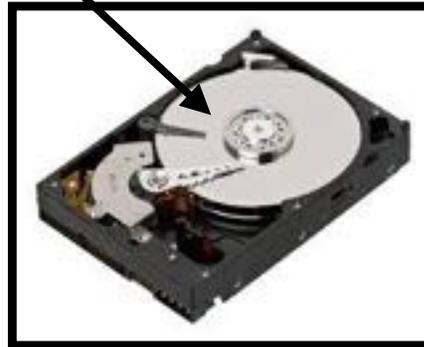


## Filesystem

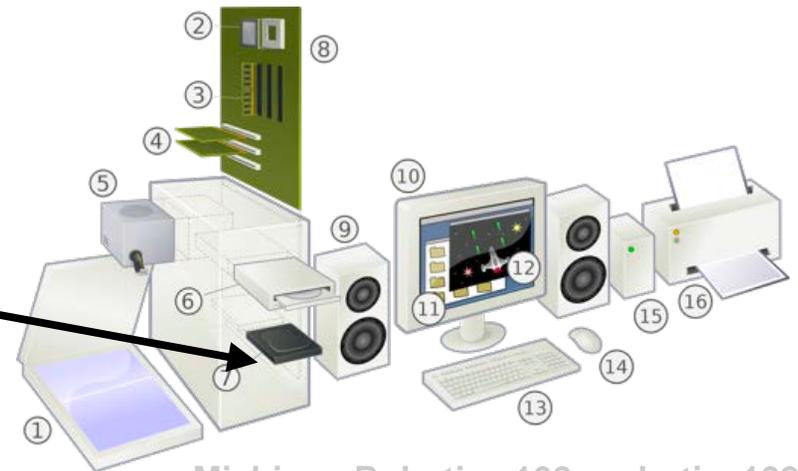


# What is a filesystem?

A filesystem organizes information on a storage device into a hierarchy of *directories* that contain data in *files*



Hard Drive Storage



# Step 2: Compile source code

## Text editor

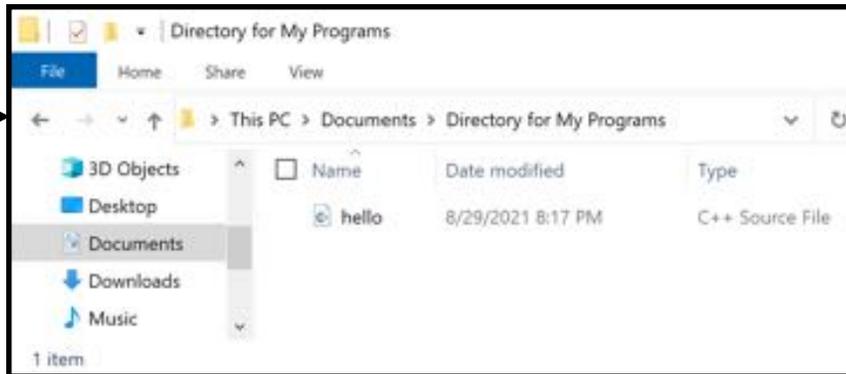
Make changes to C++ code

```
#include <iostream>
int main()
{
    std::cout << "Hello World!";
}
```

hello.cpp

*OPEN FILE*

*SAVE FILE*



## Source code

C++ files on computer file system containing your code

# Step 2: Compile source code

## Text editor

Make changes to C++ code

```
#include <iostream>
int main()
{
    std::cout << "Hello World!";
}
```

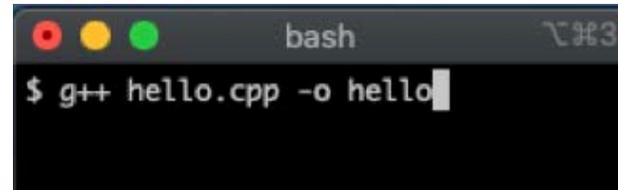
hello.cpp

*OPEN FILE*

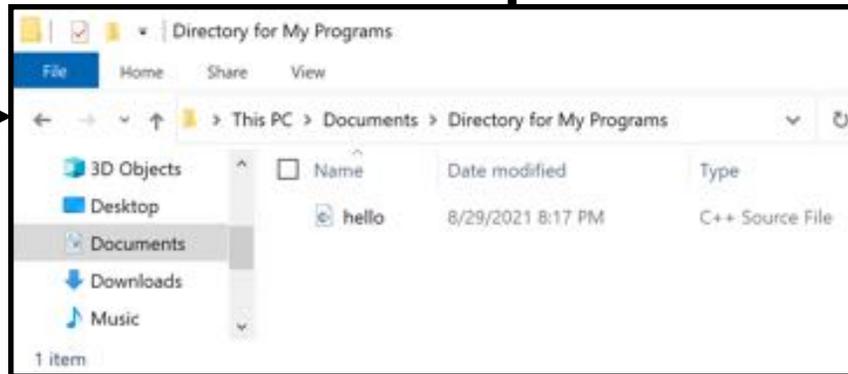
*SAVE FILE*

## Compiler

Build executable program from C++ source files



```
bash
$ g++ hello.cpp -o hello
```



## Source code

C++ files on computer file system containing your code

# Step 2: Compile source code

## Text editor

Make changes to C++ code

```
#include <iostream>
int main()
{
    std::cout << "Hello World!";
}
```

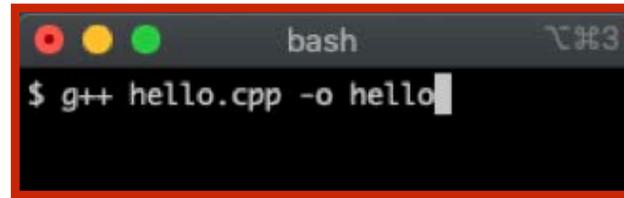
hello.cpp

*OPEN FILE*

*SAVE FILE*

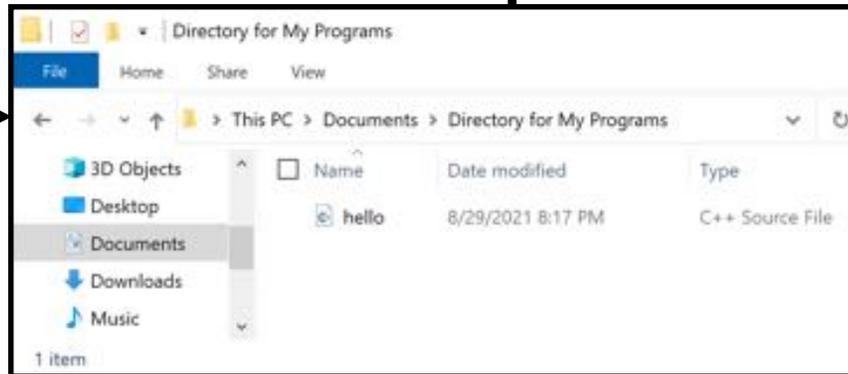
## Compiler

Build executable program from C++ source files



```
bash
$ g++ hello.cpp -o hello
```

*COMMAND LINE  
TERMINAL*



## Source code

C++ files on computer file system containing your code

# Step 2: Compile source code

## Text editor

Make changes to C++ code

```
#include <iostream>
int main()
{
    std::cout << "Hello World!";
}
```

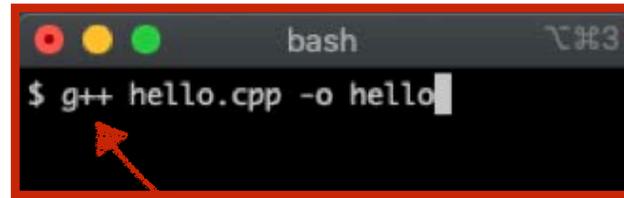
hello.cpp

*OPEN FILE*

*SAVE FILE*

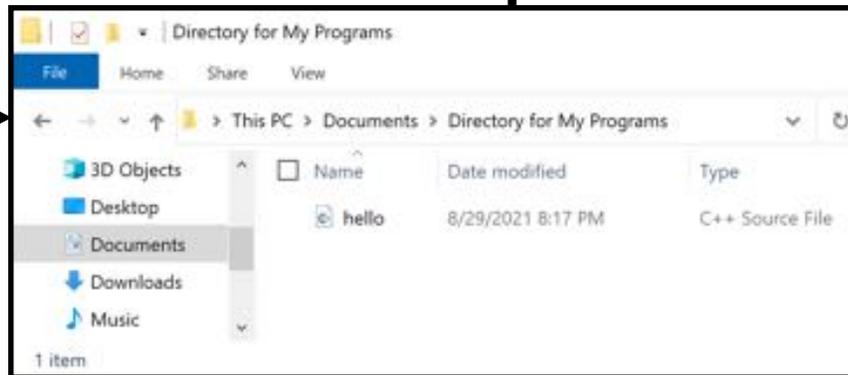
## Compiler

Build executable program from C++ source files



*COMPILER PROGRAM*

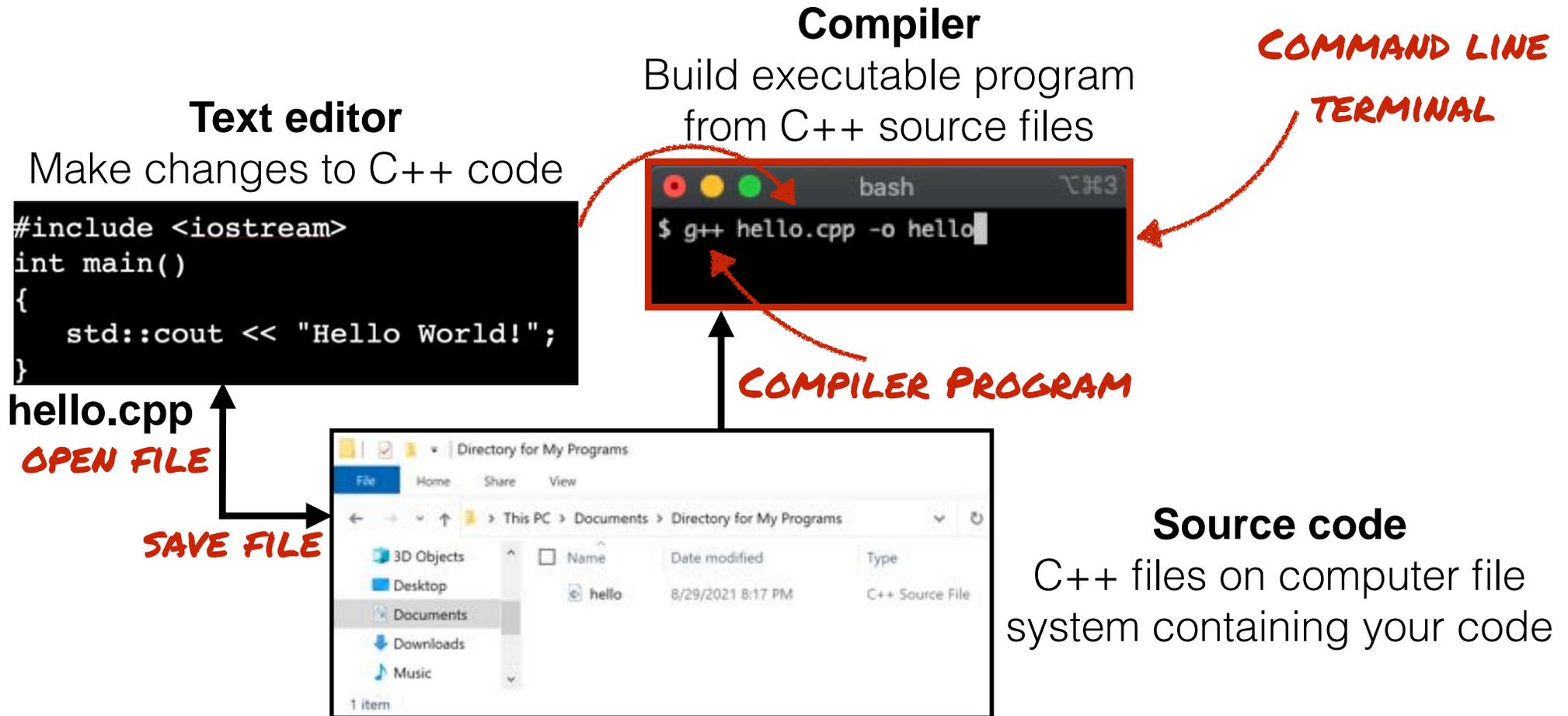
*COMMAND LINE  
TERMINAL*



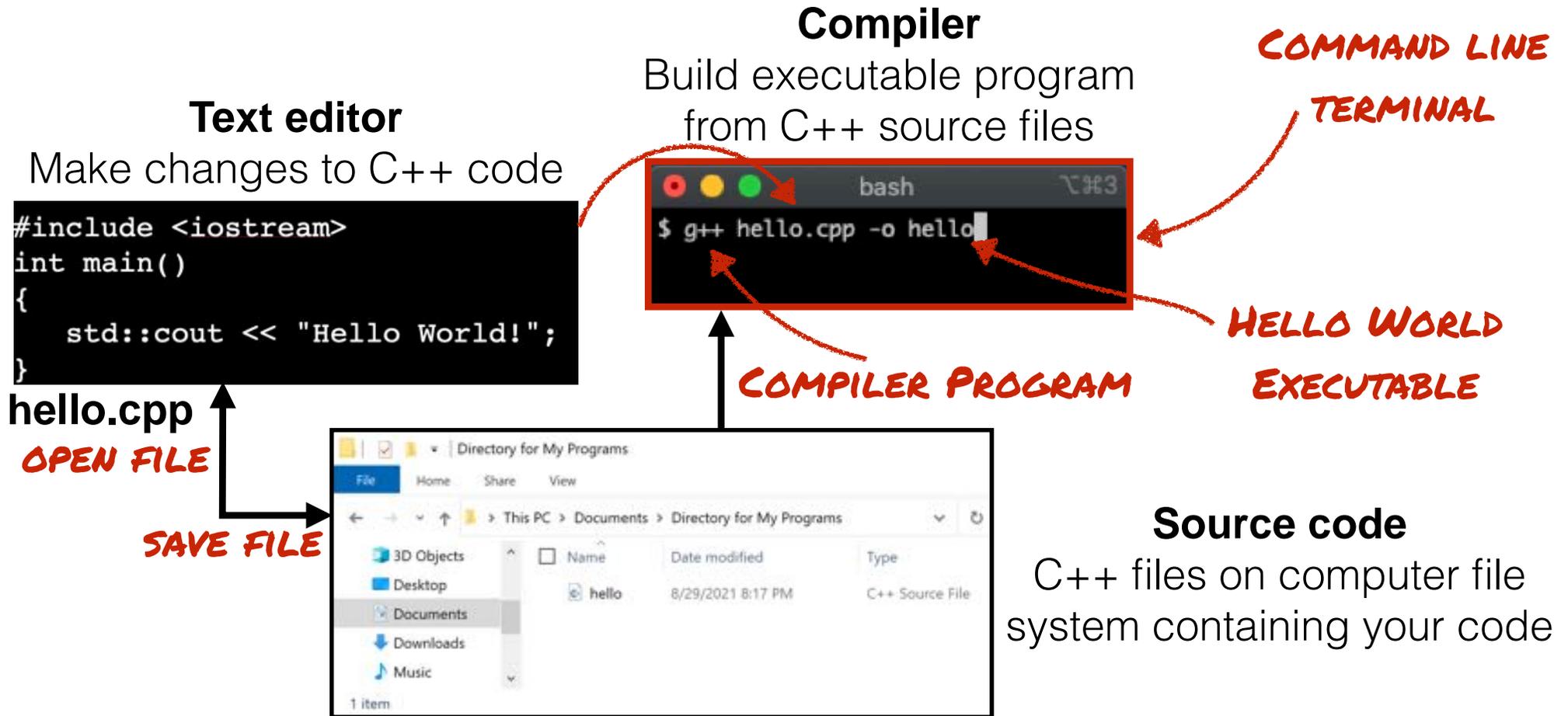
## Source code

C++ files on computer file system containing your code

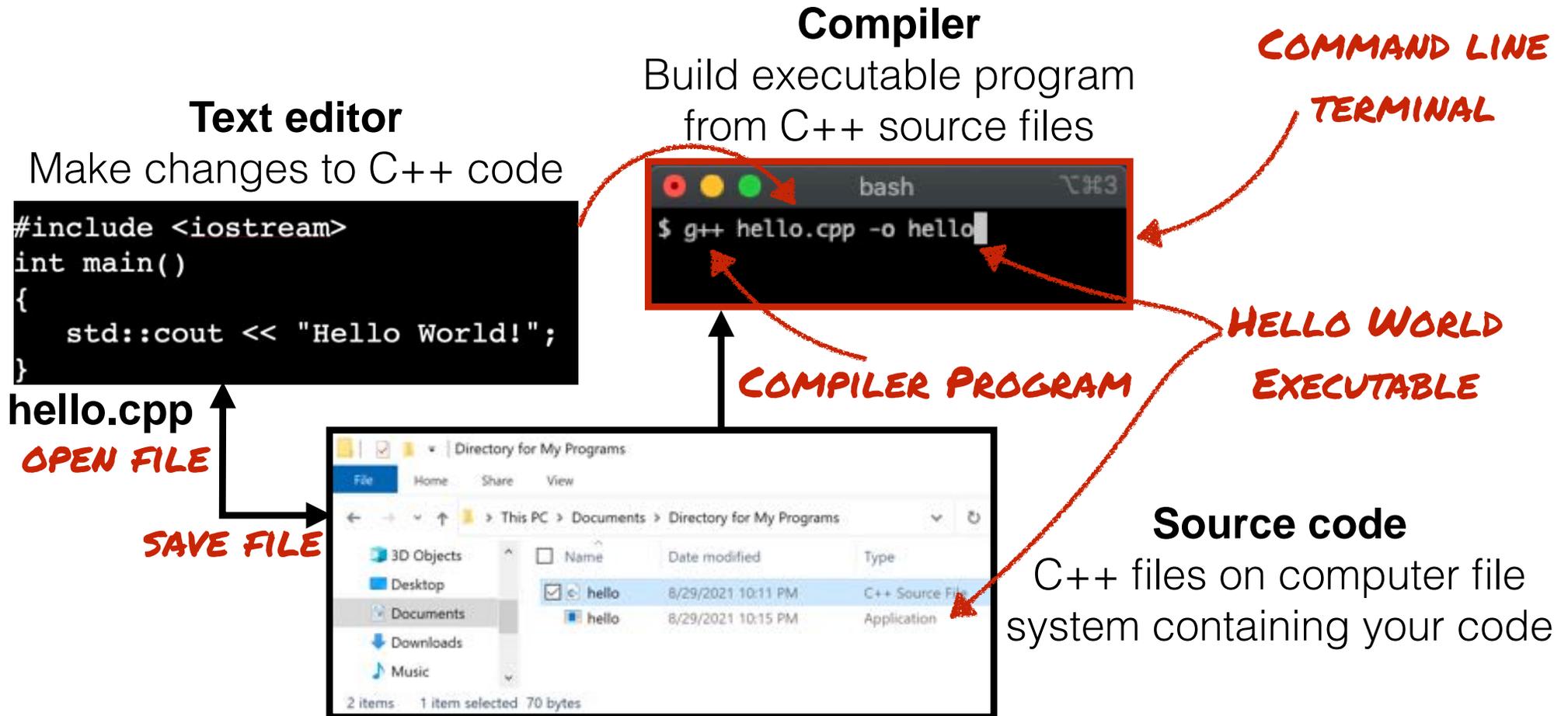
# Step 2: Compile source code



# Step 2: Compile source code



# Step 2: Compile source code



# Step 2: Compile source code

## Text editor

Make changes to C++ code

```
#include <iostream>
int main()
{
    std::cout << "Hello World!";
}
```

hello.cpp

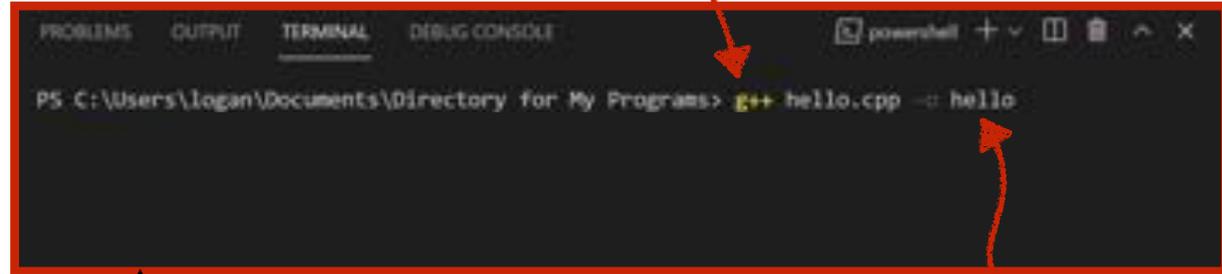
*OPEN FILE*

*SAVE FILE*

## Compiler

Build executable program from C++ source files

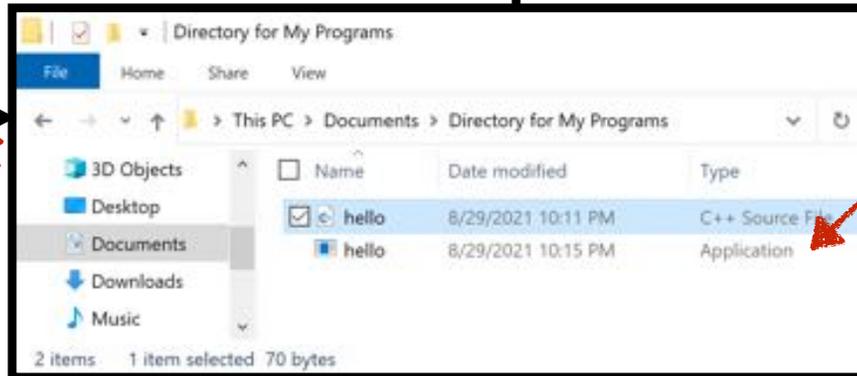
## VS Code Terminal



*HELLO WORLD EXECUTABLE*

## Source code

C++ files on computer file system containing your code



# Step 3: Run Executable Program

## Text editor

Make changes to C++ code

```
#include <iostream>
int main()
{
    std::cout << "Hello World!";
}
```

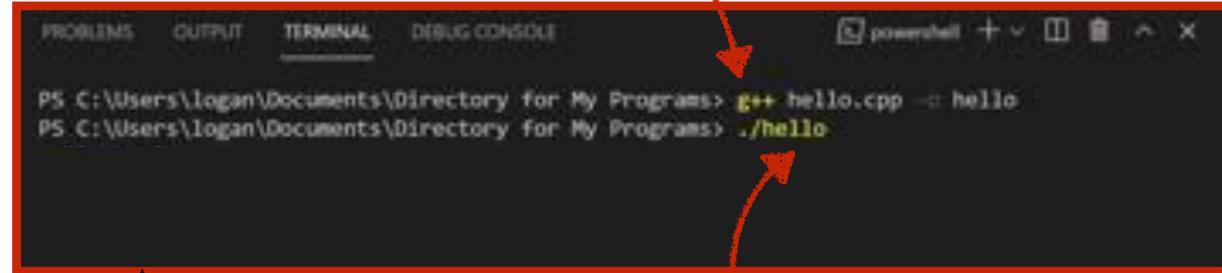
hello.cpp

*OPEN FILE*

*SAVE FILE*

## Compiler

Build executable program  
from C++ source files



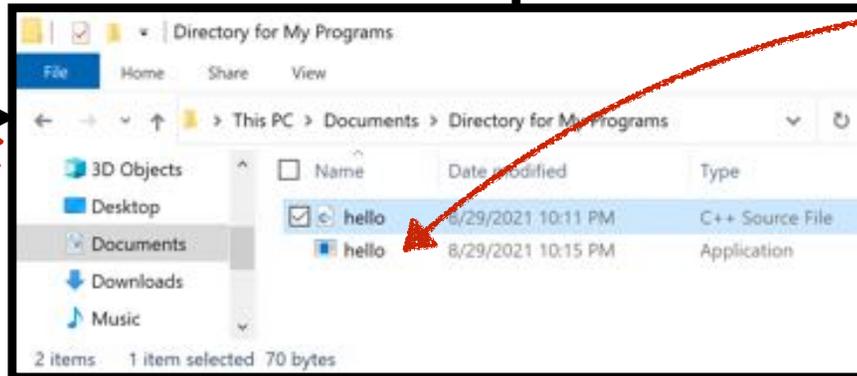
```
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
PS C:\Users\logan\Documents\Directory for My Programs> g++ hello.cpp -o hello
PS C:\Users\logan\Documents\Directory for My Programs> ./hello
```

## Executable

Binary file on computer file  
system that can be run

## Source code

C++ files on computer file  
system containing your code



# Step 3: Run Executable Program

## Text editor

Make changes to C++ code

```
#include <iostream>
int main()
{
    std::cout << "Hello World!";
}
```

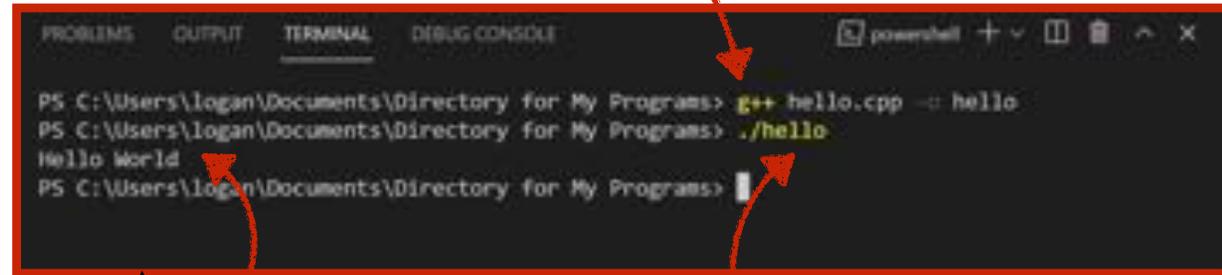
hello.cpp

*OPEN FILE*

*SAVE FILE*

## Compiler

Build executable program from C++ source files



```
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
PS C:\Users\logan\Documents\Directory for My Programs> g++ hello.cpp -o hello
PS C:\Users\logan\Documents\Directory for My Programs> ./hello
Hello World
PS C:\Users\logan\Documents\Directory for My Programs> |
```

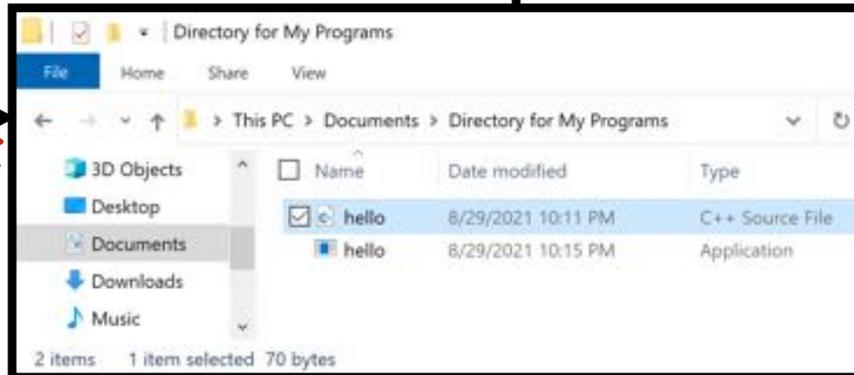
*PROGRAM OUTPUT*

## Executable

Binary file on computer file system that can be run

## Source code

C++ files on computer file system containing your code





Compiler  
Executable

```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE powershell + v
PS C:\Users\logan\Documents\Directory for My Programs> g++ hello.cpp -o hello
PS C:\Users\logan\Documents\Directory for My Programs> ./hello
Hello World
PS C:\Users\logan\Documents\Directory for My Programs>

```

IDEAS

Text editor

```

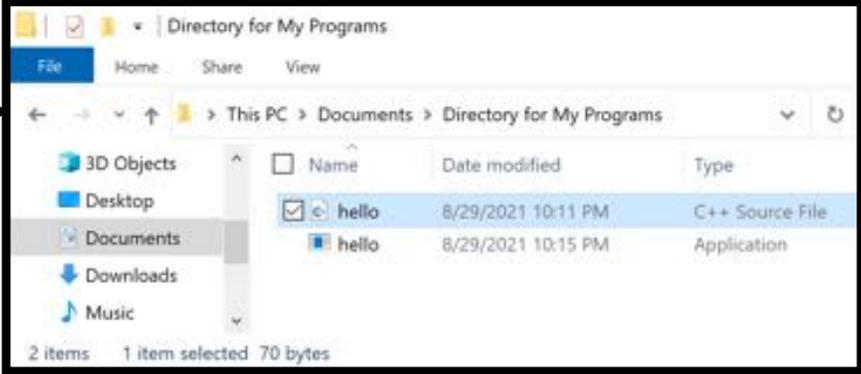
#include <iostream>
int main()
{
    std::cout << "Hello World!";
}

```

hello.cpp  
OPEN FILE

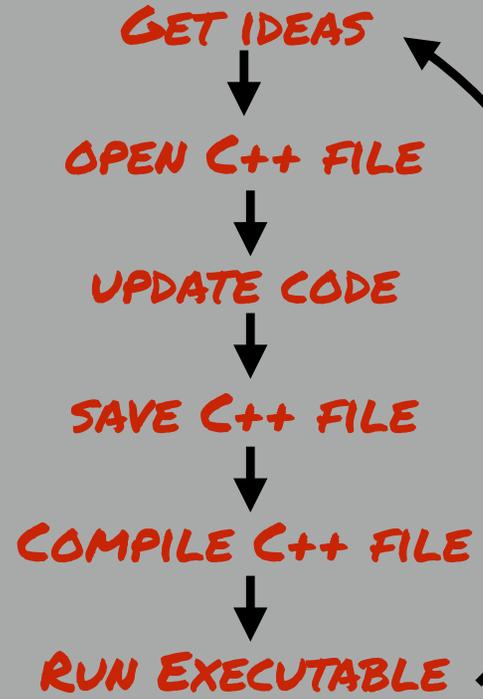
COMPILE  
AND EXECUTE

SAVE FILE



Source code

### Coding process



Compiler  
Executable

```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE powershell + v
PS C:\Users\logan\Documents\Directory for My Programs> g++ hello.cpp -o hello
PS C:\Users\logan\Documents\Directory for My Programs> ./hello
Hello World
PS C:\Users\logan\Documents\Directory for My Programs>

```

IDEAS

Text editor

```

#include <iostream>
int main()
{
    std::cout << "Hello World!";
}

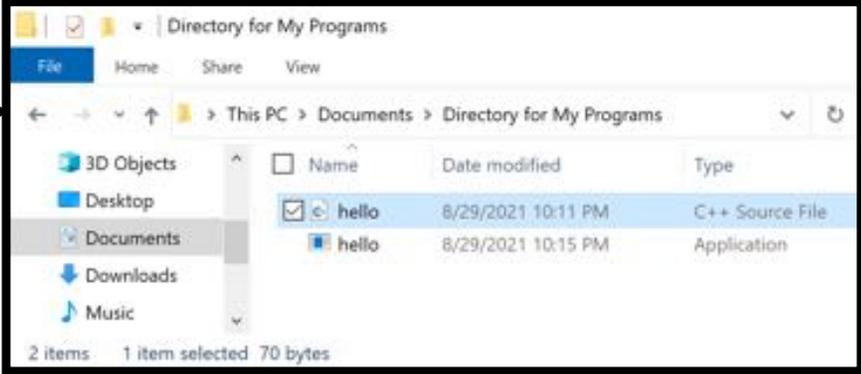
```

hello.cpp  
OPEN FILE

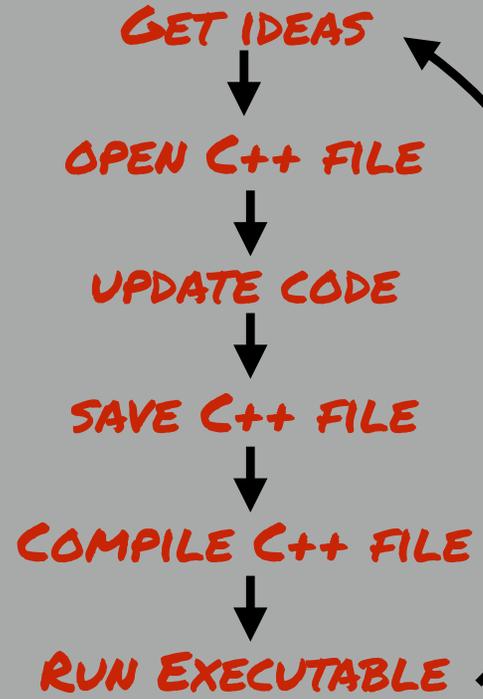
SAVE FILE

Source code

COMPILE  
AND EXECUTE



### Coding process



New idea:  
Print that I am in 102

Compiler  
Executable

```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE powershell + v
PS C:\Users\logan\Documents\Directory for My Programs> g++ hello.cpp -o hello
PS C:\Users\logan\Documents\Directory for My Programs> ./hello
Hello World
PS C:\Users\logan\Documents\Directory for My Programs>

```

*IDEAS*

Text editor

```

#include <iostream>
int main()
{
    std::cout << "Hello World";
    std::cout << "Chad is in Robotics 102";
}

```

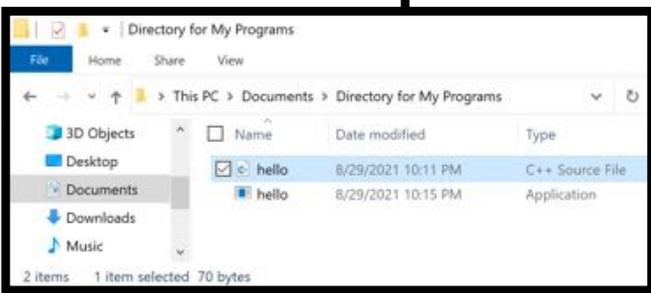
hello.cpp

*OPEN FILE*

*SAVE FILE*

Source code

*COMPILE  
AND EXECUTE*



## Coding process

*GET IDEAS*

*OPEN C++ FILE*

*UPDATE CODE*

*SAVE C++ FILE*

*COMPILE C++ FILE*

*RUN EXECUTABLE*



Compiler  
Executable

*IDEAS*

Text editor

```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE powershell + v
PS C:\Users\logan\Documents\Directory for My Programs> g++ hello.cpp -o hello
PS C:\Users\logan\Documents\Directory for My Programs> ./hello
Hello World
PS C:\Users\logan\Documents\Directory for My Programs> g++ hello.cpp -o hello
PS C:\Users\logan\Documents\Directory for My Programs> ./hello
Hello WorldChad is in Robotics 102
PS C:\Users\logan\Documents\Directory for My Programs>

```

```

#include <iostream>
int main()
{
    std::cout << "Hello World";
    std::cout << "Chad is in Robotics 102";
}

```

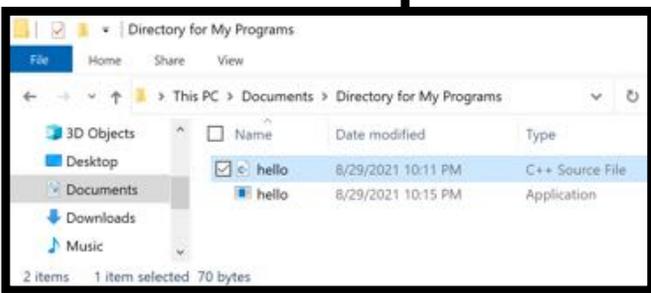
hello.cpp

*OPEN FILE*

*SAVE FILE*

Source code

*COMPILE  
AND EXECUTE*



### Coding process

*GET IDEAS*

*OPEN C++ FILE*

*UPDATE CODE*

*SAVE C++ FILE*

*COMPILE C++ FILE*

*RUN EXECUTABLE*



Compiler  
Executable

*IDEAS*

Text editor

```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE powershell + v
PS C:\Users\logan\Documents\Directory for My Programs> g++ hello.cpp -o hello
PS C:\Users\logan\Documents\Directory for My Programs> ./hello
Hello World
PS C:\Users\logan\Documents\Directory for My Programs> g++ hello.cpp -o hello
PS C:\Users\logan\Documents\Directory for My Programs> ./hello
Hello WorldChad is in Robotics 102
PS C:\Users\logan\Documents\Directory for My Programs>

```

```

#include <iostream>
int main()
{
    std::cout << "Hello World";
    std::cout << "Chad is in Robotics 102";
}

```

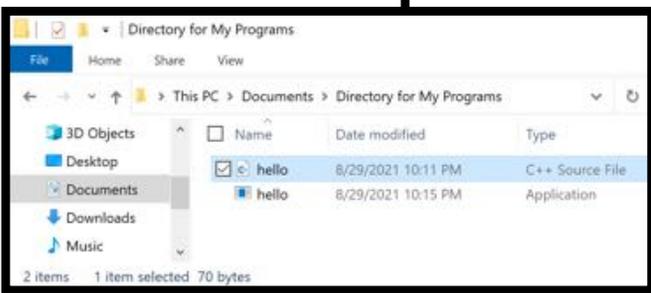
hello.cpp

*OPEN FILE*

*SAVE FILE*

Source code

*COMPILE  
AND EXECUTE*



### Coding process

*GET IDEAS*

*OPEN C++ FILE*

*UPDATE CODE*

*SAVE C++ FILE*

*COMPILE C++ FILE*

*RUN EXECUTABLE*

**New idea:  
Break each line**

```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
PS C:\Users\logan\Documents\Directory for My Programs> g++ hello.cpp -o hello
PS C:\Users\logan\Documents\Directory for My Programs> ./hello
Hello World
PS C:\Users\logan\Documents\Directory for My Programs> g++ hello.cpp -o hello
PS C:\Users\logan\Documents\Directory for My Programs> ./hello
Hello WorldChad is in Robotics 102
PS C:\Users\logan\Documents\Directory for My Programs> g++ hello.cpp -o hello
PS C:\Users\logan\Documents\Directory for My Programs> ./hello
Hello World
Chad is in Robotics 102
PS C:\Users\logan\Documents\Directory for My Programs>

```

**Compiler  
Executable**

*IDEAS*

**Text editor**

```

#include <iostream>
int main()
{
    std::cout << "Hello World" << "\n";
    std::cout << "Chad is in Robotics 102";
}

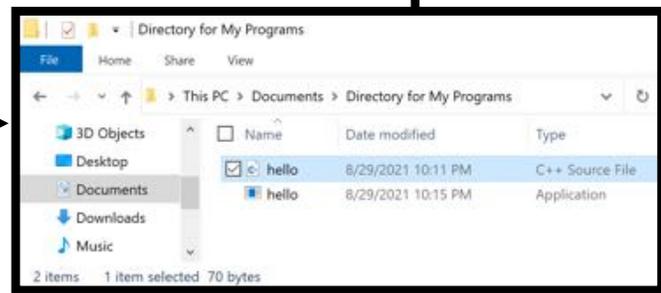
```

**hello.cpp**

*OPEN FILE*

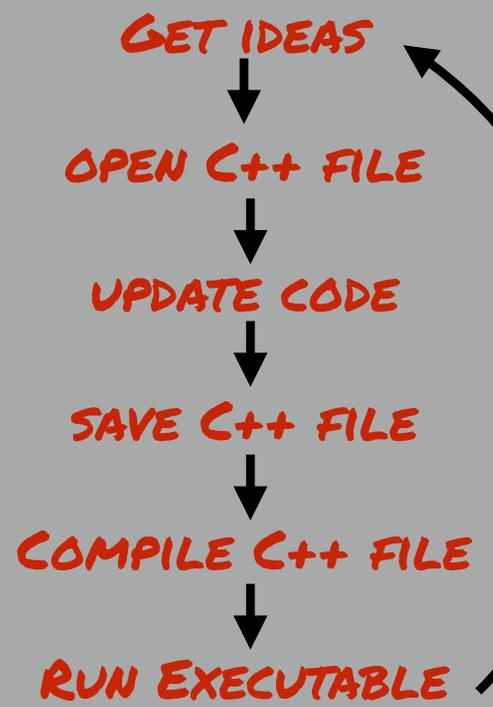
*SAVE FILE*

**Source code**



*COMPILE  
AND EXECUTE*

**Coding process**



```
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE powershell +
PS C:\Users\logan\Documents\Directory for My Programs> g++ hello.cpp -o hello
PS C:\Users\logan\Documents\Directory for My Programs> ./hello
Hello World
PS C:\Users\logan\Documents\Directory for My Programs> g++ hello.cpp -o hello
PS C:\Users\logan\Documents\Directory for My Programs> ./hello
Hello WorldChad is in Robotics 102
PS C:\Users\logan\Documents\Directory for My Programs> g++ hello.cpp -o hello
PS C:\Users\logan\Documents\Directory for My Programs> ./hello
Hello World
Chad is in Robotics 102
PS C:\Users\logan\Documents\Directory for My Programs>
```

Compiler  
Executable

*IDEAS*

Text editor

Coding process

*GET IDEAS*

*OPEN C++ FILE*

**Wait**

*AND EXECUTE*

**You did what?**

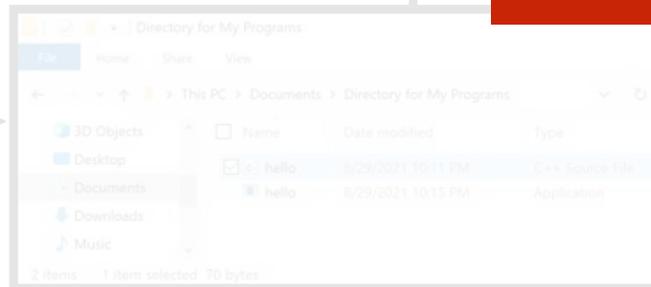
```
#include <iostream>
int main()
{
    std::cout << "Hello World" << "\n";
    std::cout << "Chad is in Robotics 102";
}
```

hello.cpp

*OPEN FILE*

*SAVE FILE*

Source code



# Anatomy of a C++ Program

```
#include <iostream>
int main()
{
    std::cout << "Hello World" << "\n";
    std::cout << "Chad is in Robotics 102";
}
```

**hello.cpp**

# Anatomy of a C++ Program

*All programs start in  
the main function*



```
#include <iostream>
int main()
{
    std::cout << "Hello World" << "\n";
    std::cout << "Chad is in Robotics 102";
}
```

**hello.cpp**

# Anatomy of a C++ Program

**All programs start in the main function**

```
#include <iostream>
int main()
{
    std::cout << "Hello World" << "\n";
    std::cout << "Chad is in Robotics 102";
}
```

hello.cpp

**Scope of main function delimited by matching braces**

# Anatomy of a C++ Program

**All programs start in the main function**

```
#include <iostream>
int main()
{
    std::cout << "Hello World" << "\n";
    std::cout << "Chad is in Robotics 102";
}
```

hello.cpp

**Scope of main function delimited by matching braces**

**Statements are program instructions and end with a semicolon**

**All programs start in the main function**

**First statement prints to "standard output" (the screen) the string (in quotes) followed by a new line character**

```
#include <iostream>
int main()
{
    std::cout << "Hello World" << "\n";
    std::cout << "Chad is in Robotics 102";
}
```

**hello.cpp**

**Scope of main function delimited by matching braces**

**Statements are program instructions and end with a semicolon**

**All programs start in the main function**

**First statement prints to "standard output" (the screen) the string (in quotes) followed by a new line character**

```
#include <iostream>
int main()
{
    std::cout << "Hello World" << "\n";
    std::cout << "Chad is in Robotics 102";
}
```

**Insertion ("put") operator**

**hello.cpp**

**Scope of main function delimited by matching braces**

**Statements are program instructions and end with a semicolon**

**Include the C++ Input-Output Stream Library, which provides `std::cout`**

**All programs start in the main function**

**First statement prints to "standard output" (the screen) the string (in quotes) followed by a new line character**

```
#include <iostream>
int main()
{
    std::cout << "Hello World" << "\n";
    std::cout << "Chad is in Robotics 102";
}
```

**Insertion ("put") operator**

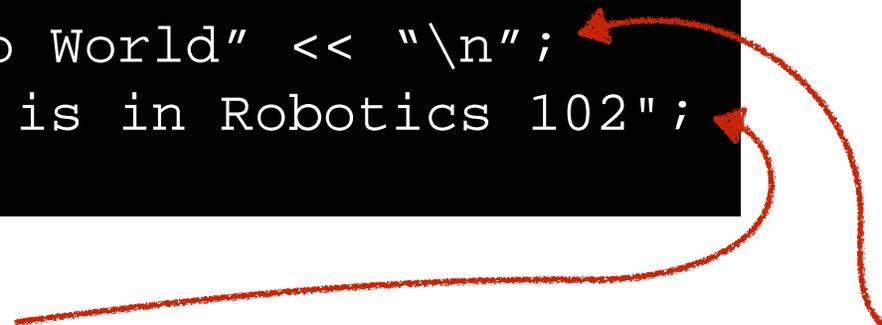
**hello.cpp**

**Scope of main function delimited by matching braces**

**Statements are program instructions and end with a semicolon**

***Statements are executed in sequential order based on where they appear in the program***

```
#include <iostream>
int main()
{
    std::cout << "Hello World" << "\n";
    std::cout << "Chad is in Robotics 102";
}
```



**hello.cpp**

***This statement is executed after this statement***

```
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE powershell +
PS C:\Users\logan\Documents\Directory for My Programs> g++ hello.cpp -o hello
PS C:\Users\logan\Documents\Directory for My Programs> ./hello
Hello World
PS C:\Users\logan\Documents\Directory for My Programs> g++ hello.cpp -o hello
PS C:\Users\logan\Documents\Directory for My Programs> ./hello
Hello WorldChad is in Robotics 102
PS C:\Users\logan\Documents\Directory for My Programs> g++ hello.cpp -o hello
PS C:\Users\logan\Documents\Directory for My Programs> ./hello
Hello World
Chad is in Robotics 102
PS C:\Users\logan\Documents\Directory for My Programs> |
```

Compiler  
Executable

*IDEAS*

Text editor

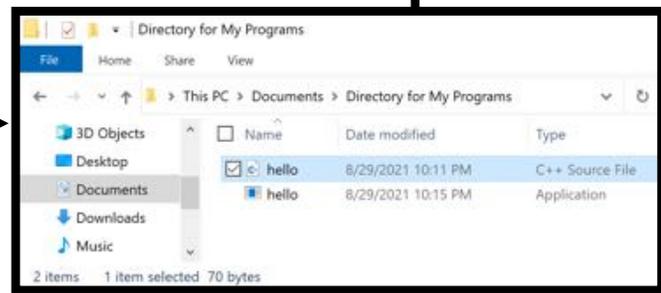
```
2 int main()
3 {
4     std::cout << "Hello World" << "\n";
5     std::cout << "Chad is in Robotics 102";
6 }
```

hello.cpp

*OPEN FILE*

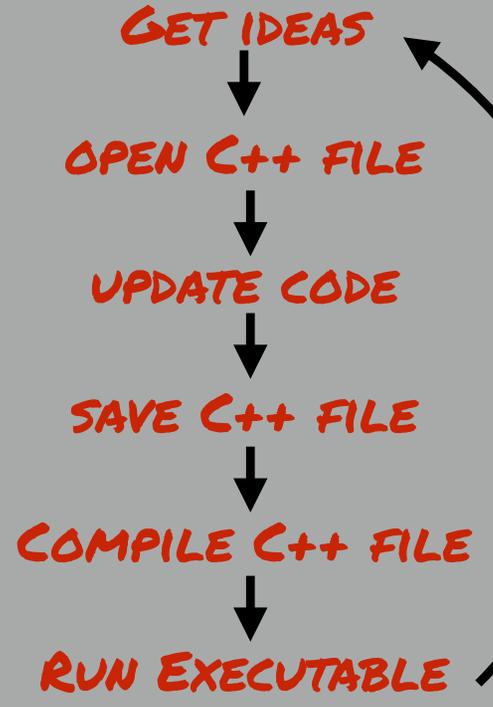
*SAVE FILE*

Source code

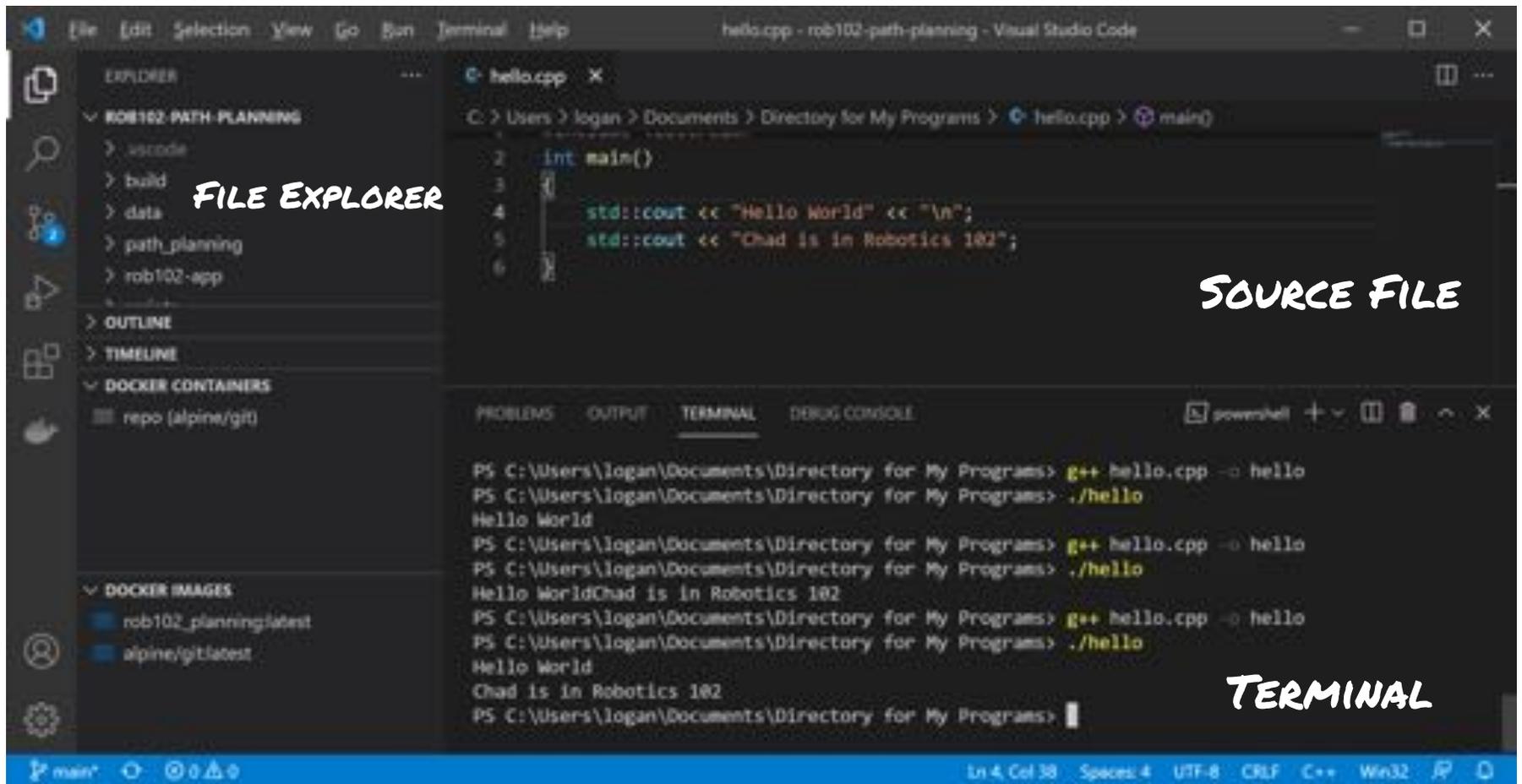


*COMPILE  
AND EXECUTE*

## Coding process



# VS Code is an Integrated Development Environment (or IDE)

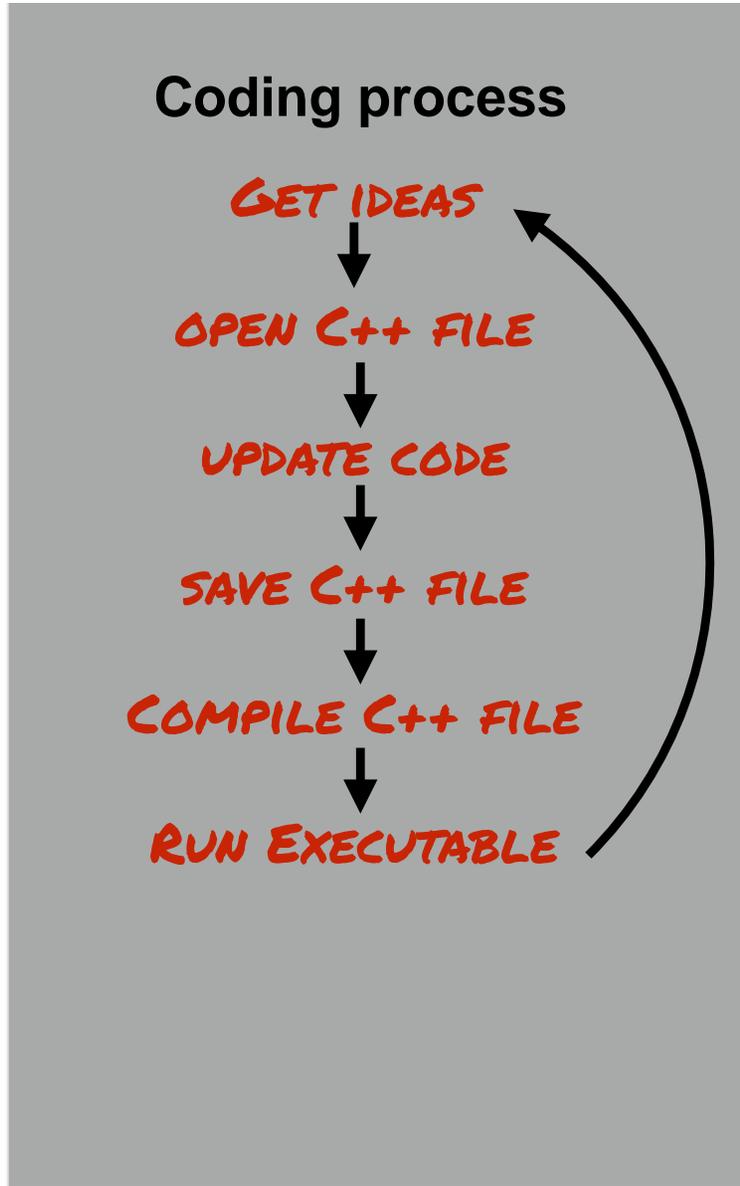


```
terminal | help | hello.cpp - rob102-path-planning - Visual Studio Code
hello.cpp x
C:\Users\logan\Documents\Directory for My Programs\hello.cpp> main()
2 int main()
3 {
4     std::cout << "Hello World" << "\n";
5     std::cout << "Chad is in Robotics 102";
6 }

PS C:\Users\logan\Documents\Directory for My Programs> g++ hello.cpp -o hello
PS C:\Users\logan\Documents\Directory for My Programs> ./hello
Hello World
PS C:\Users\logan\Documents\Directory for My Programs> g++ hello.cpp -o hello
PS C:\Users\logan\Documents\Directory for My Programs> ./hello
Hello WorldChad is in Robotics 102
PS C:\Users\logan\Documents\Directory for My Programs> g++ hello.cpp -o hello
PS C:\Users\logan\Documents\Directory for My Programs> ./hello
Hello World
Chad is in Robotics 102
PS C:\Users\logan\Documents\Directory for My Programs>
```

**SOURCE FILE**

**COMPILE**  
**EXECUTE**  
**COMPILE**  
**EXECUTE**  
**COMPILE**  
**EXECUTE**



# What happens if I make a mistake?

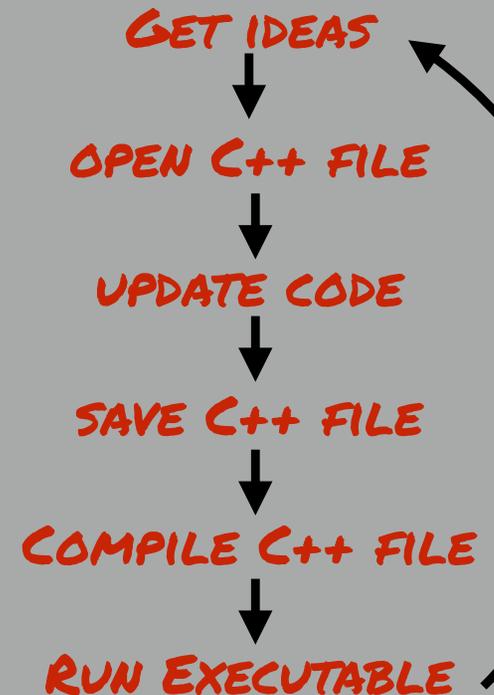
```
terminal | help | hello.cpp - rob102-path-planning - Visual Studio Code
C: > Users > logan > Documents > Directory for My Programs > hello.cpp > main()
2 int main()
3 {
4     std::cout << "Hello World" << "\n";
5     std::cout << "Chad is in Robotics 102";
6 }

PS C:\Users\logan\Documents\Directory for My Programs> g++ hello.cpp -o hello
PS C:\Users\logan\Documents\Directory for My Programs> ./hello
Hello World
PS C:\Users\logan\Documents\Directory for My Programs> g++ hello.cpp -o hello
PS C:\Users\logan\Documents\Directory for My Programs> ./hello
Hello WorldChad is in Robotics 102
PS C:\Users\logan\Documents\Directory for My Programs> g++ hello.cpp -o hello
PS C:\Users\logan\Documents\Directory for My Programs> ./hello
Hello World
Chad is in Robotics 102
PS C:\Users\logan\Documents\Directory for My Programs>
```

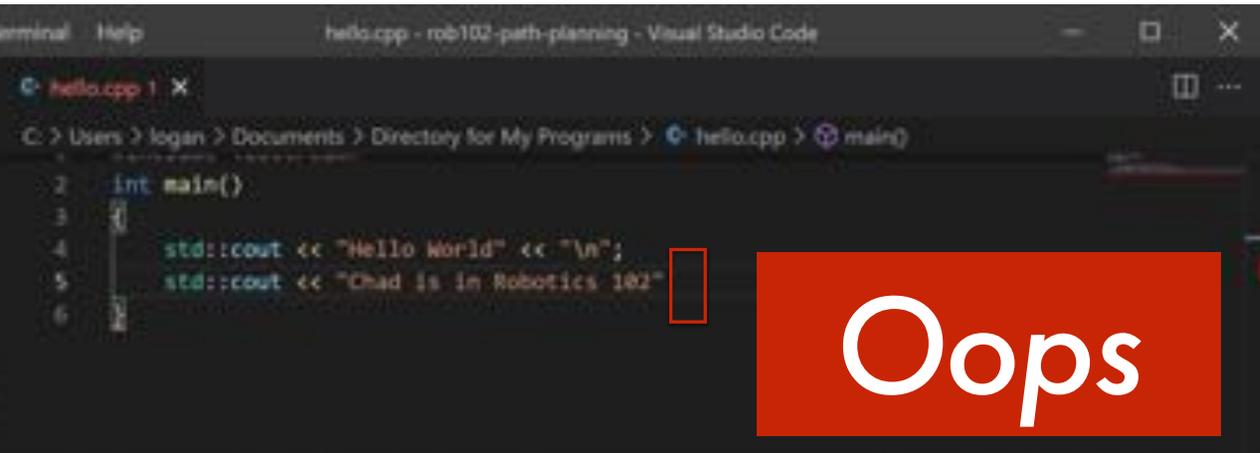
**SOURCE FILE**

**COMPILE**  
**EXECUTE**  
**COMPILE**  
**EXECUTE**  
**COMPILE**  
**EXECUTE**

## Coding process



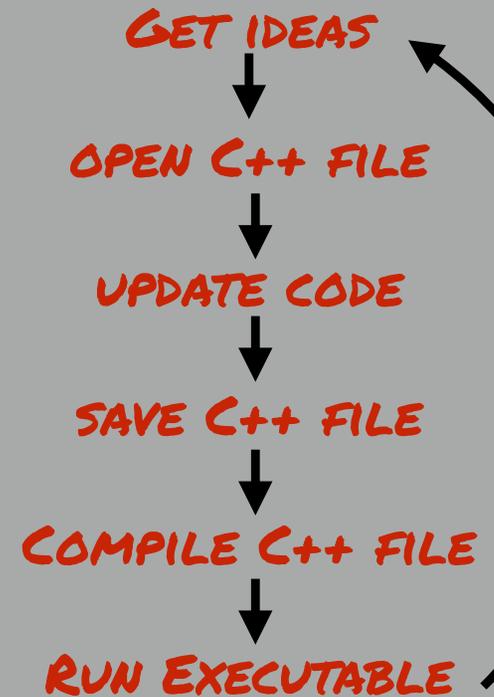
# Suppose a semicolon is forgotten



```
1 int main()  
2 {  
3     std::cout << "Hello World" << "\n";  
4     std::cout << "Chad is in Robotics 102"  
5 }  
6
```

Oops

## Coding process



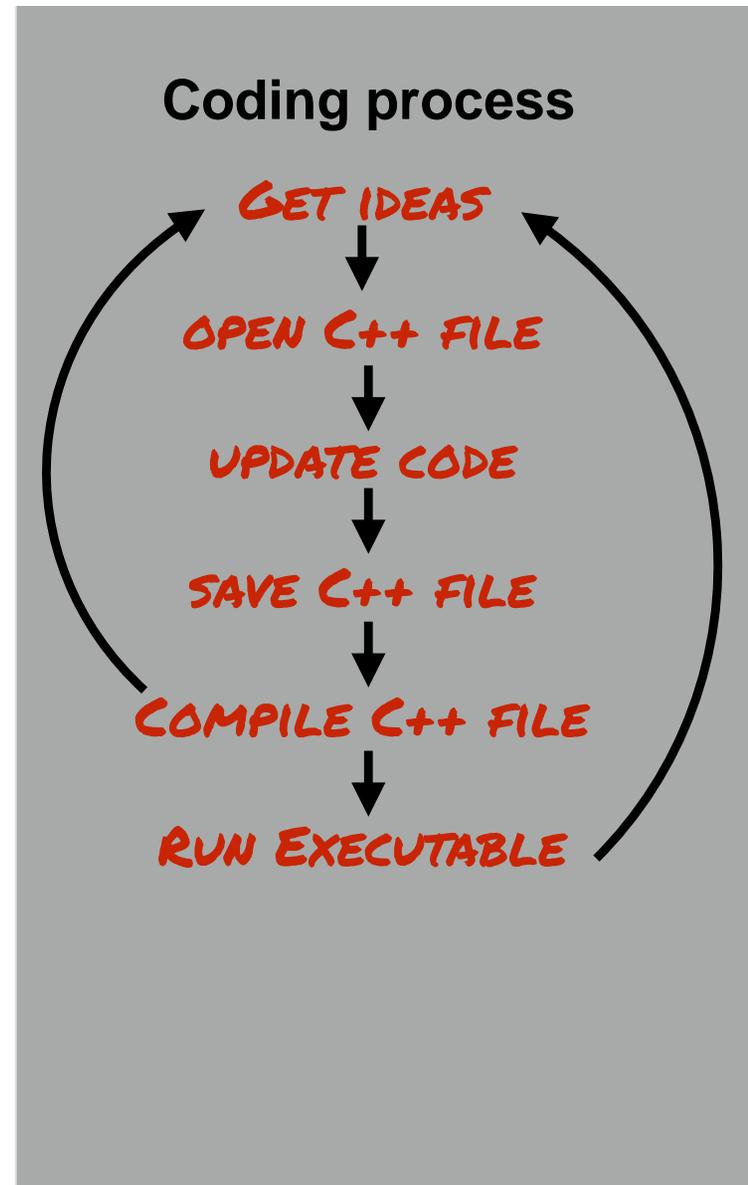


# Suppose a scoping brace is forgotten

```
terminal | help | hello.cpp - rob102-path-planning - Visual Studio Code
hello.cpp x
C:\Users\logan\Documents\Directory for My Programs\hello.cpp
2 int main()
3 {
4     std::cout << "Hello World" << "\n";
5     std::cout << "Chad is in Robotics 102";
6
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
PS C:\Users\logan\Documents\Directory for My Programs> g++ hello.cpp -o hello
hello.cpp: In function 'int main()':
hello.cpp:5:43: error: expected ')' at end of input
5 |     std::cout << "Chad is in Robotics 102";
|
hello.cpp:3:1: note: to match this '('
3 | |
|
PS C:\Users\logan\Documents\Directory for My Programs>
```

**Oops**

## Compilation will fail with an error



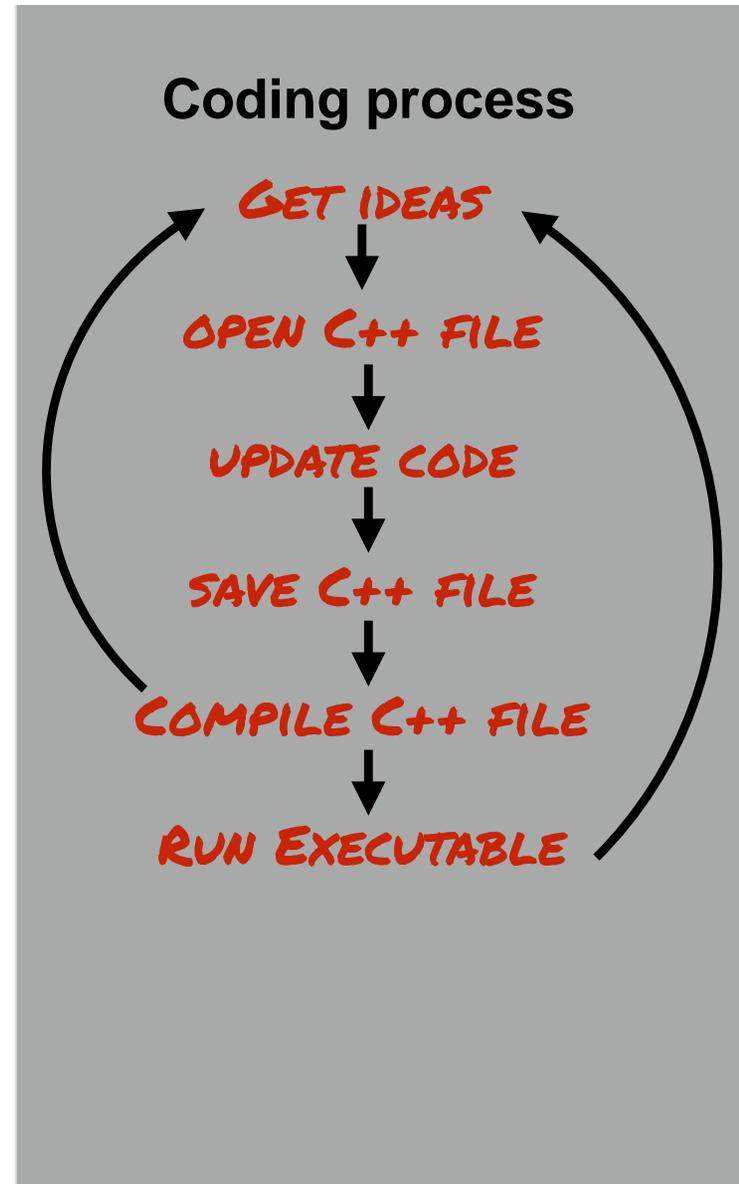
# Suppose main function is forgotten

```
1 #include <iostream>
2
3 {
4     std::cout << "Hello World" << "\n";
5     std::cout << "Chad is in Robotics 102";
6 }
```

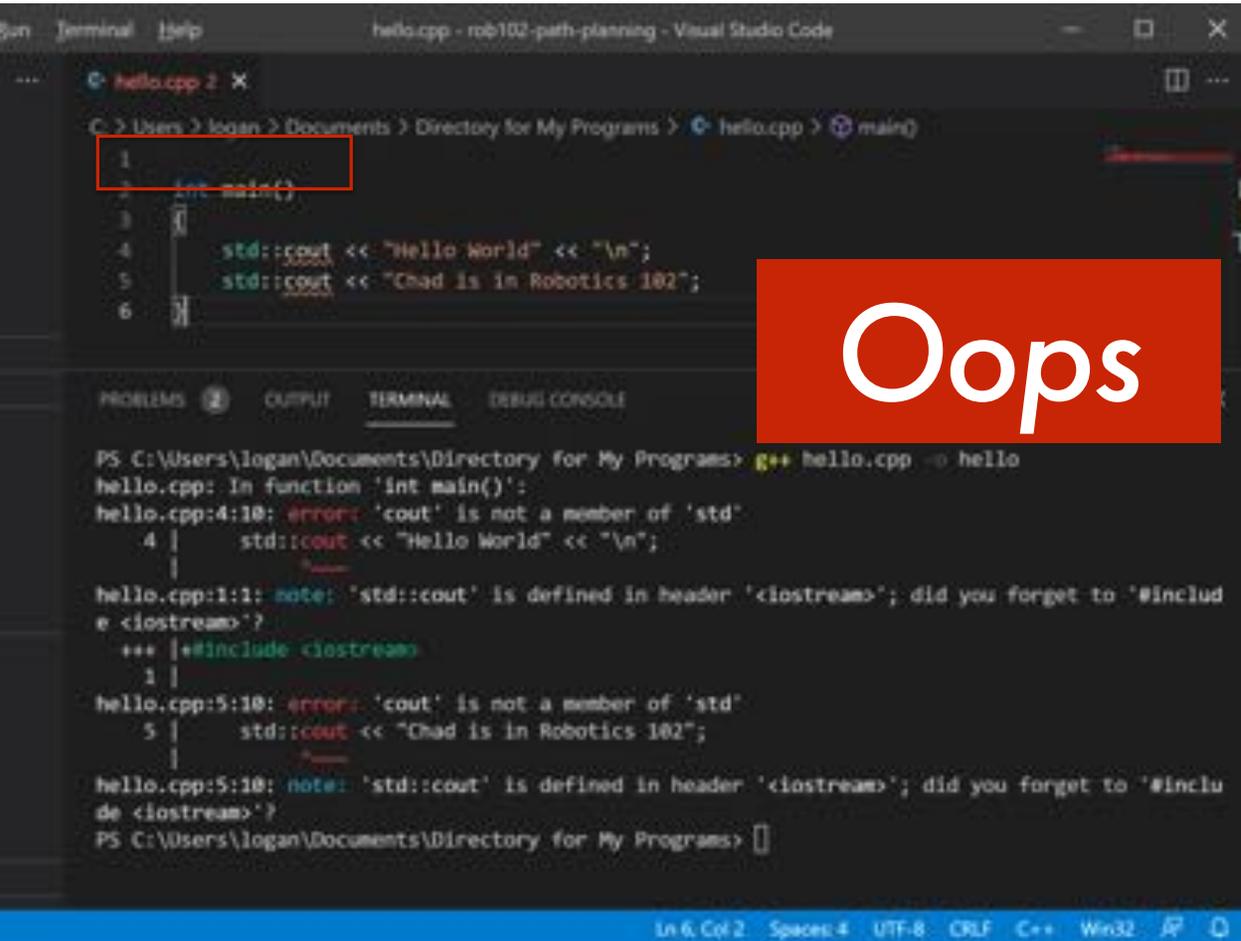
**Oops**

```
PS C:\Users\logan\Documents\Directory for My Programs> g++ hello.cpp -o hello
hello.cpp:3:1: error: expected unqualified-id before '{' token
 3 | {
  | ^
PS C:\Users\logan\Documents\Directory for My Programs>
```

## Compilation will fail with an error



# Suppose needed library is forgotten

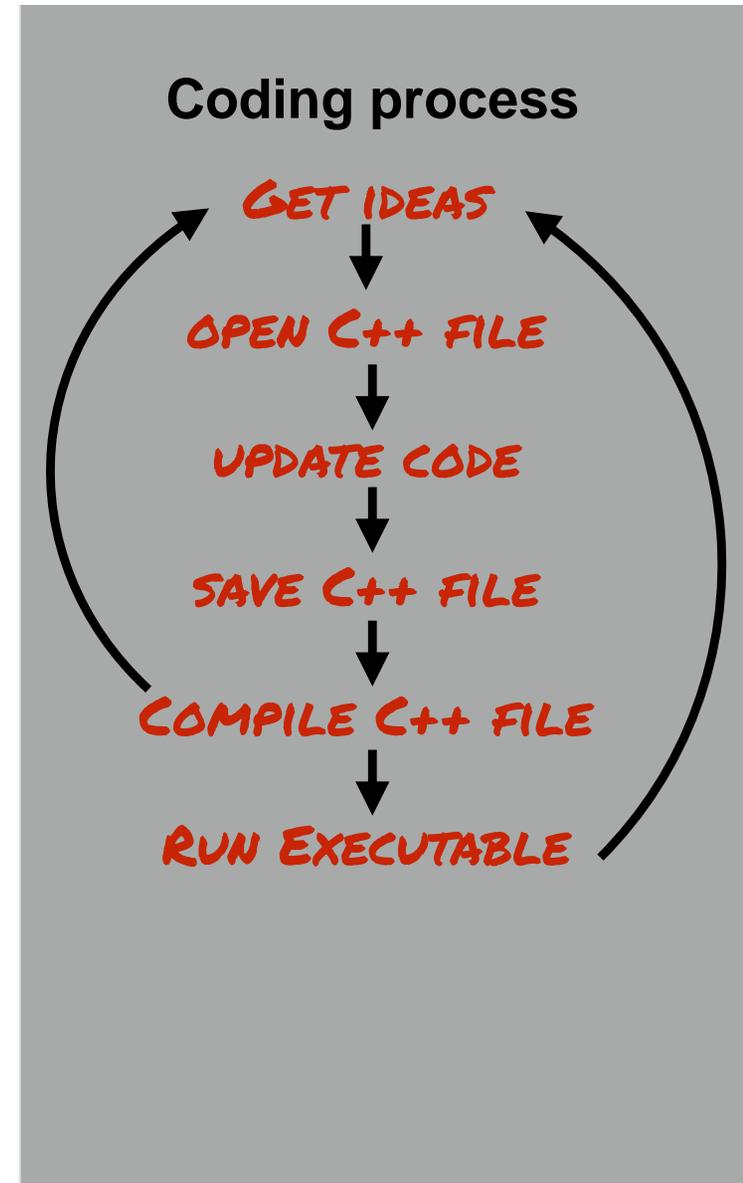


```
1  
2 int main()  
3  
4     std::cout << "Hello World" << "\n";  
5     std::cout << "Chad is in Robotics 102";  
6
```

**Oops**

```
PS C:\Users\logan\Documents\Directory for My Programs> g++ hello.cpp -o hello  
hello.cpp: In function 'int main()':  
hello.cpp:4:10: error: 'cout' is not a member of 'std'  
   4 |     std::cout << "Hello World" << "\n";  
     |           ^~~~~  
hello.cpp:1:1: note: 'std::cout' is defined in header '<iostream>'; did you forget to '#include <iostream>'?  
+++ |##include <iostream>  
   1 |  
hello.cpp:5:10: error: 'cout' is not a member of 'std'  
   5 |     std::cout << "Chad is in Robotics 102";  
     |           ^~~~~  
hello.cpp:5:10: note: 'std::cout' is defined in header '<iostream>'; did you forget to '#include <iostream>'?  
PS C:\Users\logan\Documents\Directory for My Programs>
```

## Compilation will fail with errors



Suppose I am just careless

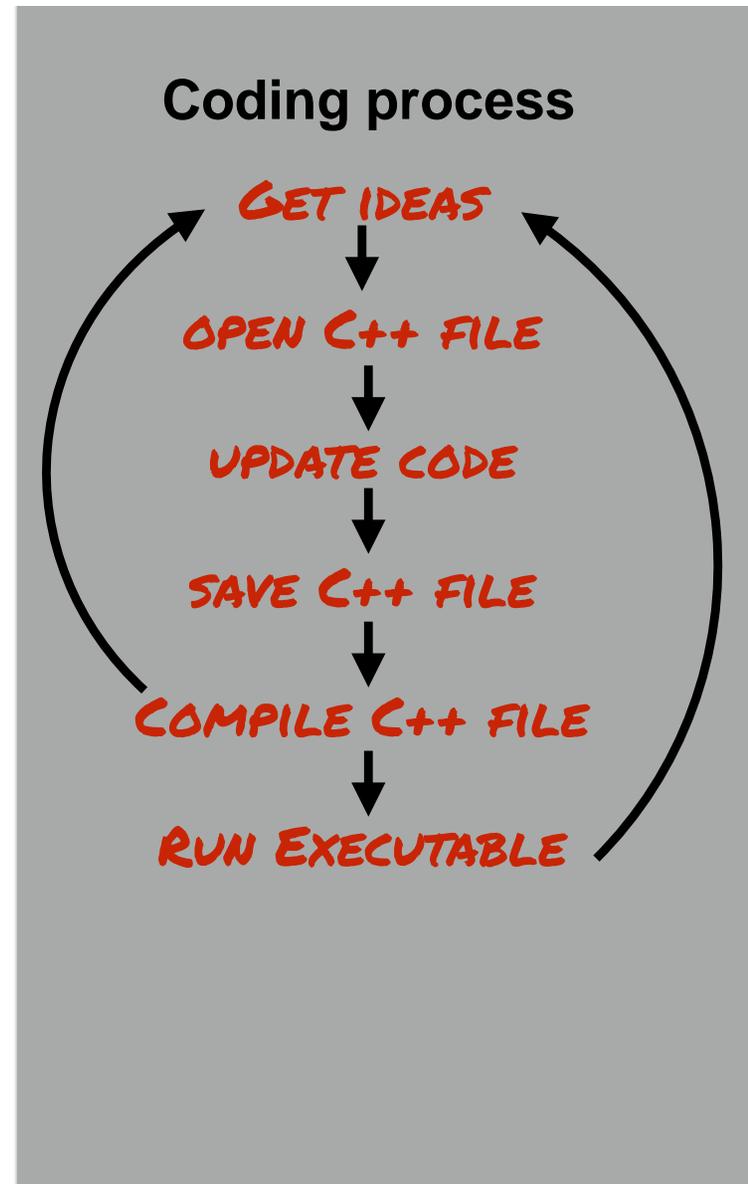
The screenshot shows a Visual Studio Code editor window with a C++ file named `hello.cpp`. The code is as follows:

```
1 #include <iostream>
2 int main()
3 {
4     whatever text because I feel like it
5     std::cout << "Hello World" << "\n";
6     std::cout << "Chad is in Robotics 102";
7 }
```

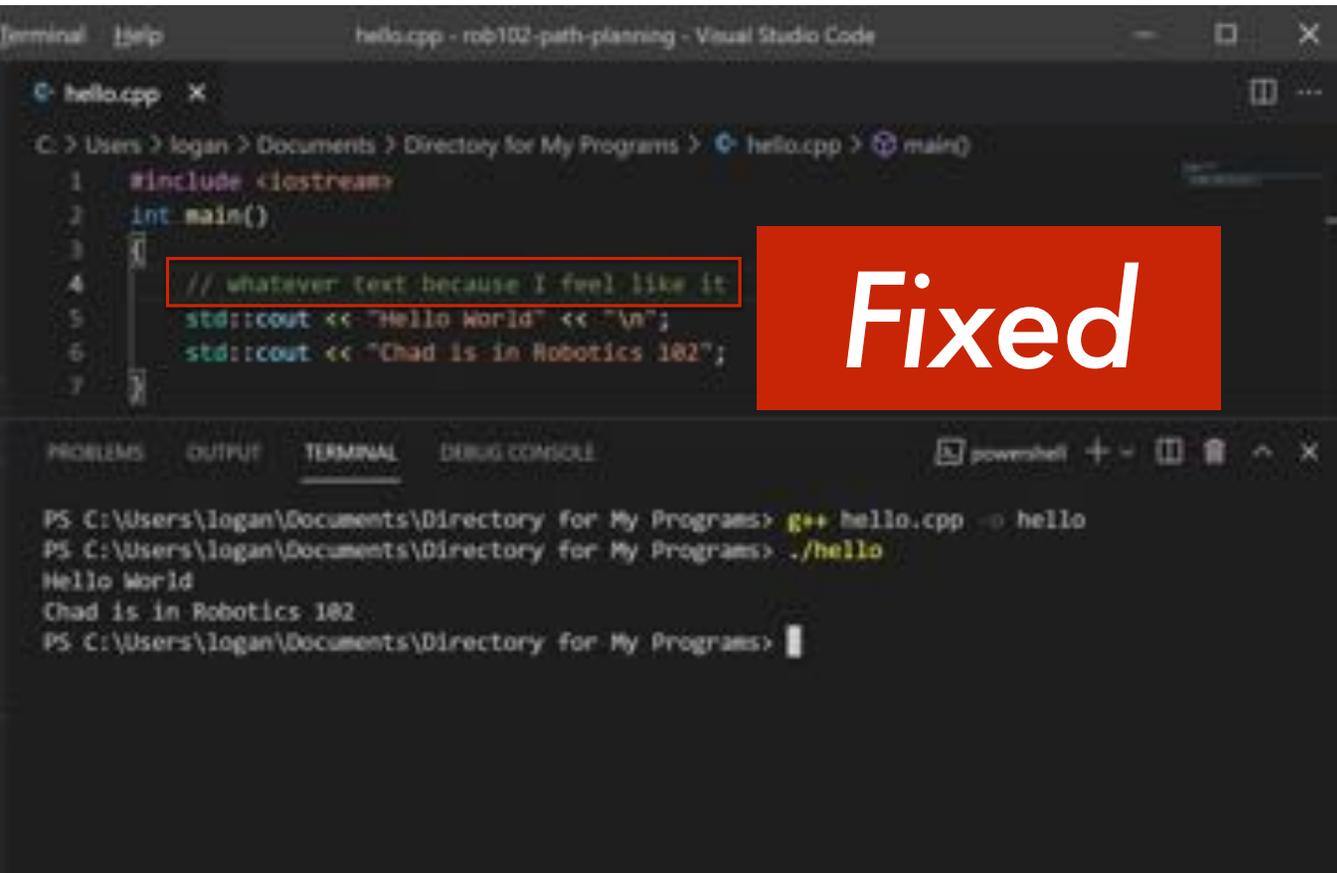
The word `whatever` on line 4 is highlighted with a red box. To the right of the code, a red box contains the word `Oops` in white text. Below the code editor, the terminal shows the command `g++ hello.cpp -o hello` and the resulting error message:

```
hello.cpp: In function 'int main()':
hello.cpp:4:5: error: 'whatever' was not declared in this scope
  4 |     whatever text because I feel like it
    |     ^~~~~~
PS C:\Users\logan\Documents\Directory for My Programs>
```

Compilation will fail with an error



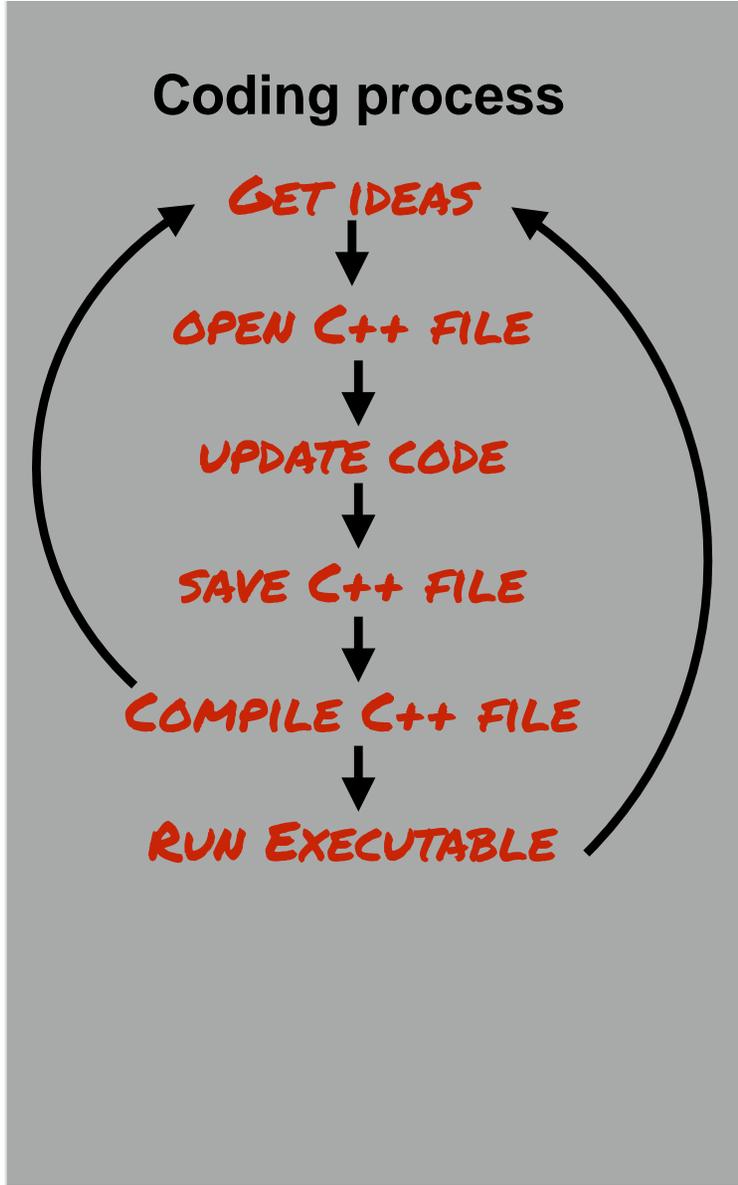
Suppose I am just careless



```
Terminal | help | hello.cpp - rob102-path-planning - Visual Studio Code
hello.cpp x
C: > Users > logan > Documents > Directory for My Programs > hello.cpp > main()
1 #include <iostream>
2 int main()
3
4 // whatever text because I feel like it
5 std::cout << "Hello World" << "\n";
6 std::cout << "Chad is in Robotics 102";
7

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
PS C:\Users\logan\Documents\Directory for My Programs> g++ hello.cpp -o hello
PS C:\Users\logan\Documents\Directory for My Programs> ./hello
Hello World
Chad is in Robotics 102
PS C:\Users\logan\Documents\Directory for My Programs> |
```

**Fixed**



That can be “commented out”

# C++ Comments

***Comments are ignored by the compiler  
and not included in the program***

```
#include <iostream>
/*
  This is a multi-line comment. It is ignored by my program.
  This is our first program.
  The program code below prints messages to the screen.
*/

int main()
{
  std::cout << "Hello World" << "\n";  A single-line comment
  std::cout << "Chad is in Robotics 102"; // "\n" creates a new line
}
```

hello.cpp

***Anything after double slashes on a line  
is ignored as a comment***

# C++ Comments

*Comments are ignored by the compiler and not included in the program*

*Anything in between delimiters `/*` and `*/` is ignored as a comment*

```
#include <iostream>
/*
 This is a multi-line comment. It is ignored by my program.
 This is our first program.
 The program code below prints messages to the screen.
*/

int main()
{
    std::cout << "Hello World" << "\n"; // A single-line comment
    std::cout << "Chad is in Robotics 102"; // "\n" creates a new line
}
```

hello.cpp

*Anything after double slashes on a line is ignored as a comment*

# *Your code is a battleground*

```
#include <iostream>
/*
  This is a multi-line comment. It is ignored by my program.
  This is our first program.
  The program code below prints messages to the screen.
*/

int main()
{
  std::cout << "Hello World" << "\n"; // A single-line comment
  std::cout << "Chad is in Robotics 102"; // "\n" creates a new line
}
```

**hello.cpp**

# *Your code is a battleground*

```
#include <iostream>
/* Hello World - A first C++ Program
   Copyright 2021 Odest Chadwicke Jenkins at the University of Michigan
   Licensed under Michigan Honor License in the LICENSE file and
   available to view at https://autorob.org/MichiganHonorLicense.txt
*/

int main()
{
    std::cout << "Hello World" << "\n"; // A single-line comment
    std::cout << "Chad is in Robotics 102"; // "\n" creates a new line
}
```

**hello.cpp**

```
#include <iostream>
/* Hello World - A first C++ Program
   Copyright 2021 Odest Chadwicke Jenkins at the University of Michigan
   Licensed under Michigan Honor License in the LICENSE file and
   available to view at https://autorob.org/MichiganHonorLicense.txt
*/

int main()
{
    std::cout << "Hello World" << "\n"; // A single-line comment
    std::cout << "Chad is in Robotics 102"; // "\n" creates a new line
}
```

## hello.cpp

## LICENSE

The Michigan Honor License

This unvetted license below is called the "Michigan Honor License" as the 3-Clause BSD License plus two clauses for academic integrity.

Copyright <YEAR> <COPYRIGHT HOLDER>

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

```

#include <iostream>
/* Hello World - A first C++ Program
   Copyright 2021 Odest Chadwicke Jenkins at the University of Michigan
   Licensed under Michigan Honor License in the LICENSE file and
   available to view at https://autorob.org/MichiganHonorLicense.txt
*/

int main()
{
    std::cout << "Hello World" << "\n"; // A single-line comment
    std::cout << "Chad is in Robotics 102"; // "\n" creates a new line
}

```

## hello.cpp

## LICENSE

The Michigan Honor License

This unvetted license below is called the "Michigan Honor License" as the 3-Clause BSD License plus two clauses for academic integrity.

**Copyright <YEAR> <COPYRIGHT HOLDER>**

***This year and your name***

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

# LICENSE

The Michigan Honor License

This unvetted license below is called the "Michigan Honor License" as the 3-Clause BSD License plus two clauses for academic integrity.

## **BSD 3-Clause License**

<https://opensource.org/licenses/BSD-3-Clause>

modification, are

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.
4. Manuscripts and publications using this source code or its binary forms must properly cite the this project and its author(s).
5. Redistributions of source code, if used for credit in an academic program, must retain the academic integrity honor pledge below.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE

# LICENSE

The Michigan Honor License

This unvetted license below is called the "Michigan Honor License" as the 3-Clause BSD License plus two clauses for academic integrity.

Copyright <YEAR> <COPYRIGHT HOLDER>

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

## **Michigan Engineering Honor Code**

<https://bulletin.engin.umich.edu/rules/>

4. Redistributions of source code, if used for credit in an academic program, must retain the following honor pledge, and only append the names all individuals who contributed modifications.

the academic integrity honor pledge below.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE

names of its contributors may be used to  
without specific prior written

# LICENSE

The Michigan Honor License

This unvetted license below is called the "Michigan Honor License" as the 3-Clause BSD License plus two clauses for academic integrity.

Copyright <YEAR> <COPYRIGHT HOLDER>

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

**Attribution similar to [CC BY 4.0 License](https://creativecommons.org/licenses/by/4.0/)**

**<https://creativecommons.org/licenses/by/4.0/>**

5. Manuscripts and publications using this source code or its binary forms must properly cite this project and its author(s).

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE

*Share your code with the world*

*and your future self*

```
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
PS C:\Users\logan\Documents\Directory for My Programs> g++ hello.cpp -o hello
PS C:\Users\logan\Documents\Directory for My Programs> ./hello
Hello World
PS C:\Users\logan\Documents\Directory for My Programs> g++ hello.cpp -o hello
PS C:\Users\logan\Documents\Directory for My Programs> ./hello
Hello WorldChad is in Robotics 102
PS C:\Users\logan\Documents\Directory for My Programs> g++ hello.cpp -o hello
PS C:\Users\logan\Documents\Directory for My Programs> ./hello
Hello World
Chad is in Robotics 102
PS C:\Users\logan\Documents\Directory for My Programs> |
```

Compiler  
Executable

*IDEAS*

Text editor

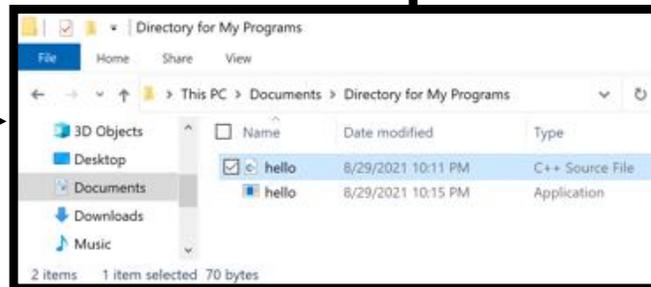
```
1 int main()
2 {
3     //
4     std::cout << "Hello World" << "\n";
5     std::cout << "Chad is in Robotics 102";
6 }
```

hello.cpp

*OPEN FILE*

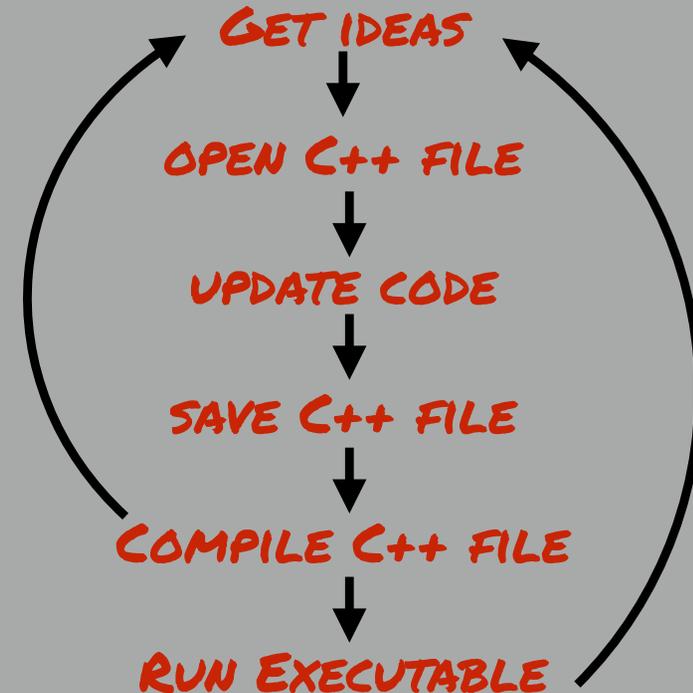
*SAVE FILE*

Source code



*COMPILE  
AND EXECUTE*

## Coding process



```
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
PS C:\Users\logan\Documents\Directory for My Programs> g++ hello.cpp -o hello
PS C:\Users\logan\Documents\Directory for My Programs> ./hello
Hello World
PS C:\Users\logan\Documents\Directory for My Programs> g++ hello.cpp -o hello
PS C:\Users\logan\Documents\Directory for My Programs> ./hello
Hello WorldChad is in Robotics 102
PS C:\Users\logan\Documents\Directory for My Programs> g++ hello.cpp -o hello
PS C:\Users\logan\Documents\Directory for My Programs> ./hello
Hello World
Chad is in Robotics 102
PS C:\Users\logan\Documents\Directory for My Programs> |
```

Compiler  
Executable

*IDEAS*

Text editor

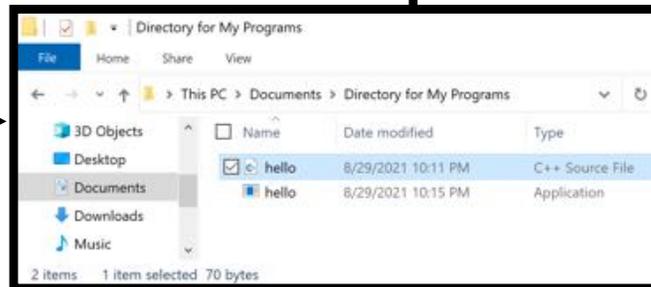
```
2 int main()
3 {
4     std::cout << "Hello World" << "\n";
5     std::cout << "Chad is in Robotics 102";
6 }
```

hello.cpp

*OPEN FILE*

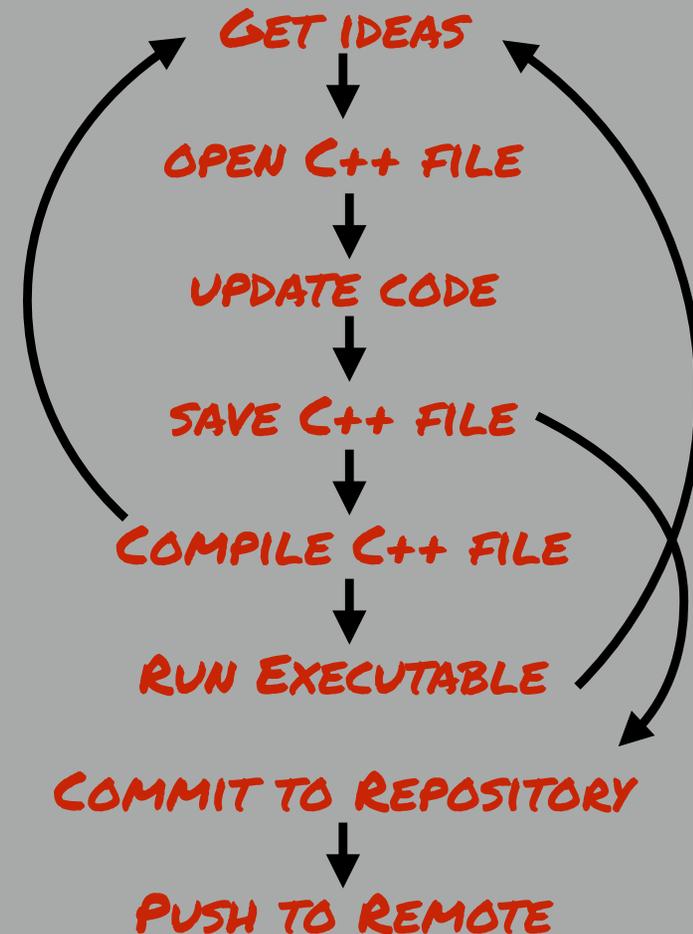
*SAVE FILE*

Source code



*COMPILE  
AND EXECUTE*

## Coding process



```
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
PS C:\Users\logan\Documents\Directory for My Programs> g++ hello.cpp -o hello
PS C:\Users\logan\Documents\Directory for My Programs> ./hello
Hello World
PS C:\Users\logan\Documents\Directory for My Programs> g++ hello.cpp -o hello
PS C:\Users\logan\Documents\Directory for My Programs> ./hello
Hello WorldChad is in Robotics 102
PS C:\Users\logan\Documents\Directory for My Programs> g++ hello.cpp -o hello
PS C:\Users\logan\Documents\Directory for My Programs> ./hello
Hello World
Chad is in Robotics 102
PS C:\Users\logan\Documents\Directory for My Programs> |
```

Compiler  
Executable

*IDEAS*

Text editor

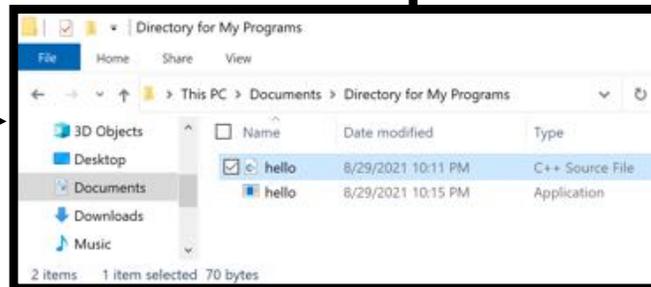
```
1 int main()
2 {
3     // ...
4     std::cout << "Hello World" << "\n";
5     std::cout << "Chad is in Robotics 102";
6 }
```

hello.cpp

*OPEN FILE*

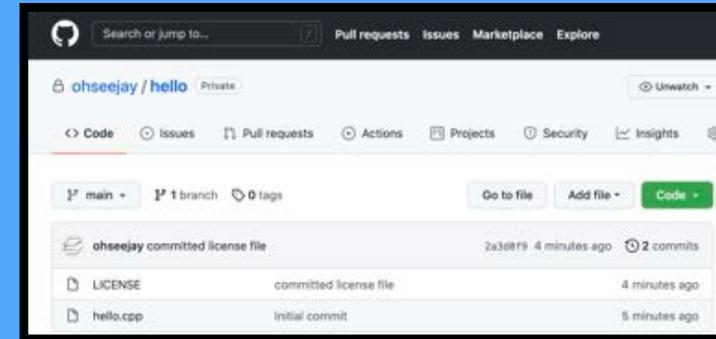
*SAVE FILE*

Source code



*COMPILE  
AND EXECUTE*

**git repository**  
store history of  
code changes  
and collaborate with others



*PUSH REPO*

*PULL REPO*

# Version Control Using git

# What is Version Control?

- Maintains a past history of changes for your code (or any project)
- History of changes (or “commits”) maintained in a repository
- Basic workflow
  - Code is “checked out” (or “pulled”) from a repository, then modified
  - These updates are then “checked in” (or “committed”) to the repository
  - Repository maintains history as “diffs”, the changes between before and after checking in a commit

For example... ocj's TED talk

Henry Evans and Chad Jenkins:

# Meet the robots for humanity

TEDxMidAtlantic · 10:21 · Filmed Oct 2013

28 subtitle languages

View interactive transcript



Watch later



Favorite



Download



Rate

Share this idea



1,145,408 Total views

Share this talk and track your influence

TED Talks are free thanks to support from





Henry Evans and Chad Jenkins:

# Meet the robots for humanity

TEDxMidAtlantic · 10:21 · Filmed Oct 2013

28 subtitle languages

View interactive transcript



Watch later



Favorite



Download



OCT IN WASHINGTON DC

Share this idea



1,145,408 Total views

Share this talk and track your influence

TED Talks are free thanks to support from





Watch Discover Attend

Search...

Log in



HENRY EVANS IN PALO ALTO CA

# Meet the robots for humanity



TEDxMidAtlantic · 10:21 · Filmed Oct 2013

28 subtitle languages  
View interactive transcript



OCT IN WASHINGTON DC

- Watch later
- Favorite
- Download
- Rate

Share this idea



1,145,408 Total views

Share this talk and track your influence!

TED Talks are free thanks to support from **Infosys**



**IN-BROWSER DRONE CONTROL**

The screenshot shows the GitHub repository page for 'odestic/tutorial\_rosbridge\_ardrone'. The repository description is: 'Front-end web interface code associated with teleoperation tutorial for AR.Drone using rosbridge/ROS. This is a simple interface to illustrate basic concepts, and not a maintained release. Please refer to robotwebtools.org for the latest and greatest. <http://rosbridge.org/doku.php?do=search&id=ar.drone> — Edit'. The repository has 7 commits, 1 branch, 0 releases, and 2 contributors. The commit history table is as follows:

Commit	Message	Time
alicef	Create Fly ar.drone through ROS Hydro	3 years ago
	Initial commit	3 years ago
	added actual drone_browser_idleop.html with buttons	3 years ago
	Create Fly ar.drone through ROS Hydro	2 years ago
	Added rosbridge tutorial for ROS Fuerte	2 years ago
	added launch file	3 years ago

The repository name 'tutorial\_rosbridge\_ardrone' is displayed at the bottom of the page.

View history of changes

GitHub, Inc. 6US9 https://github.com/odestcj/tutorial\_rosbridge\_ardrone/loc Search

odestcj / tutorial\_rosbridge\_ardrone Unwatch 0 Star 3 Fork 3

Code Issues 0 Pull requests 0 Wiki Pulse Graphs Settings

Branch: master ▾

Commits on May 15, 2014

-  **Create Fly ar.drone through ROS Hydro** [M] alicef committed on May 15, 2014 7 3eb5113 ↔

Commits on May 6, 2014

-  **Added rosbridge tutorial for ROS Fuerte** odestcj committed on May 6, 2014 bd33463 ↔
-  **Create README.md** odestcj committed on May 6, 2014 8599ede ↔

Commits on Apr 30, 2013

-  **added launch file** odestcj committed on Apr 30, 2013 374fa28 ↔

Commits on Apr 18, 2013

-  **added actual drone\_browser\_teleop.html with buttons** odestcj committed on Apr 18, 2013 645618a ↔
-  **Fixed mappings of buttons and keys to drone commands** odestcj committed on Apr 18, 2013 6893bf ↔

Commits on Apr 5, 2013

cs102.org

**View history  
of changes**

Large open source projects...



**3D INTERACTIONS**  
USING THE LATEST IN WEBGL



**MULTI-PLATFORM SUPPORT**  
HARNESSING THE POWER OF ROS



**TOWARDS COMPATIBILITY**  
MORE BROWSERS, MORE ROBOTS.

## ROBOT WEB ARCHITECTURE

BRIDGING ROBOTS AND THE WEB

A variety of routes are available for architecting a robot web

## ROSBRIDGE AS A TRANSPORT

USING JSON TO SPEAK TO YOUR ROBOT

While ROS works great for applications on the robot, another layer is

GitHub repository page for RobotWebTools / rosbridge\_suite, September 2016.

Repository: RobotWebTools / rosbridge\_suite

523 commits, 7 branches, 37 releases, 33 contributors

Server Implementations of the rosbridge v2 Protocol <http://robotwebtools.org/> — Edit

File	Commit Message	Time Ago
rosapi	Update proxy.py	4 months ago
rosbridge_library	0.7.13	5 months ago
rosbridge_server	enable udp	2 months ago
rosbridge_suite	0.7.13	5 months ago
.gitignore	cleanup of old misc. files from old merge of new features	2 years ago
.travis.yml	ci: test with and without ujson	a year ago
AUTHORS.md	authors and license added	2 years ago
CHANGELOG.md	update the change log	2 years ago
LICENSE	authors and license added	2 years ago
README.md	Update README.md	a year ago
ROSBRIDGE_PROTOCOL...	protocol documented for advertise service functions	a year ago

September 2016

523 commits

33 contributors

RobotWebTools / rosbridge\_suite

Watch 35 Star 106 Fork 114

Code Issues 36 Pull requests 1 Projects 0 Insights

Server implementations of the rosbridge v2 Protocol <http://robotwebtools.org/>

September 2017

623 commits 9 branches 48 releases 47 contributors BSD-3-Clause

623 commits

47 contributors

rosapi	0.8.3	7 hours ago
rosbridge_library	0.8.3	7 hours ago
rosbridge_server	0.8.3	7 hours ago
rosbridge_suite	0.8.3	7 hours ago
.gitignore	Gitignore vim swapfile	a year ago
.travis.yml	Cleaning up travis configuration (#283)	2 months ago
AUTHORS.md	authors and license added	3 years ago
CHANGELOG.md	update the change log	4 years ago
LICENSE	authors and license added	3 years ago
README.md	Update README.md	3 years ago

RobotWebTools / rosbridge\_suite

Unwatch 47 Unstar 159 Fork 155

Code Issues 38 Pull requests 3 Insights Settings

Server Implementations of the rosbridge v2 Protocol <http://robotwebtools.org/>

Edit

September 2018

653 commits 12 branches 52 releases 58 contributors

653 commits

58 contributors

rosapi	Fix a few problems (#350)	25 days ago
rosbridge_library	use package format 2, remove unnecessary dependencies (#348)	25 days ago
rosbridge_server	Fix a few problems (#350)	25 days ago
rosbridge_suite	use package format 2, remove unnecessary dependencies (#348)	25 days ago
.gitignore	Gitignore vim swapfile	2 years ago
.travis.yml	Fix Travis config (#311)	8 months ago
AUTHORS.md	authors and license added	5 years ago
CHANGELOG.md	update the change log	5 years ago

Server Implementations of the rosbridge v2 Protocol <http://robotwebtools.org/> Edit

September 2019

685 commits 12 branches 59 releases 65 contributors View license

685 commits

65 contributors

Clone or download

Latest commit (268) 28 days ago

johnl	0.11.3 release (#424)	
<a href="#">.github</a>	Add GitHub issue template and TROUBLESHOOTING.md (#397)	5 months ago
<a href="#">rosapi</a>	0.11.3 release (#424)	28 days ago
<a href="#">rosbridge_library</a>	0.11.3 release (#424)	28 days ago
<a href="#">rosbridge_msgs</a>	0.11.3 release (#424)	28 days ago
<a href="#">rosbridge_server</a>	0.11.3 release (#424)	28 days ago
<a href="#">rosbridge_suite</a>	0.11.3 release (#424)	28 days ago
<a href="#">.gitignore</a>	Gitignore vim swapfile	3 years ago
<a href="#">.travis.yml</a>	Travis CI: Look for Python syntax errors and undefined name (#420)	2 months ago
<a href="#">AUTHORS.md</a>	authors and license added	6 years ago

# RobotWebTools / rosbridge\_suite

Unwatch 66    Unstar 348    Fork 285

Code    Issues 38    Pull requests 3    Actions    Security    Insights    Settings

## August 2020

develop -    18 branches    68 tags

Go to file    Add file +    Code -    About

727 commits

flynnwa	add ros 1 github actions (#526)	✓	11/26/20	5 days ago
github	add ros 1 github actions (#526)			
rosepi	Fixed filter_globs for noetic (#508)			
rosbridge_library	possible fix for error when working with RosSharp, TypeError: can onl...			2 months ago
rosbridge_msgs	0.11.9			3 months ago
rosbridge_server	Error initialization with tornado. (#510)			3 months ago
rosbridge_suite	0.11.9			3 months ago
.gitignore	Gitignore vim swapfile			4 years ago
.travis.yml	noetic tests (#503)			3 months ago
AUTHORS.md	Add myself to contributors			5 months ago
CHANGELOG.md	update the change log			7 years ago
Dockerfile	noetic tests (#503)			3 months ago
LICENSE	authors and license added			7 years ago

View license

### Releases

68 tags

Create a new release

### Packages

No packages published  
Publish your first package

Contributors (8)

August 2021

jbandes 1.0.8 ✓ e37244a 4 days ago

github	Miscellaneous files: fixed --select=E402,E731,E741,F821 (#621)	10 days ago
roscpl	1.0.8	4 days ago
rosbridge_library	1.0.8	4 days ago
rosbridge_msgs	1.0.8	4 days ago
rosbridge_server	1.0.8	4 days ago
rosbridge_suite	1.0.8	4 days ago
.gitignore	Add cbor raw compression support (#574)	27 days ago
AUTHORS.md	fix: remove json encoding before setting string params (#527)	12 months ago
CHANGELOG.md	Fix types discovered by codespell (#600)	12 days ago
LICENSE	authors and license added	8 years ago
README.md	Remove .travis.yml (#595)	12 days ago

- robotwebtools.org/
- Readme
- View license

Releases

- 78 tags
- Create a new release

GitHub, Inc. [US] https://github.com/RobotWebTools/rosbridge\_suite/com Search

This repository Pull requests Issues Gist

RobotWebTools / rosbridge\_suite Unwatch 28 Unstar 42 Fork 71

Code Issues 15 Pull requests 2 Pulse Graphs Settings

Branch: develop

Commits on Nov 12, 2015

- Merge pull request #197 from xuhao1/UDP retoria committed on Nov 12, 2015 45822ab

Commits on Nov 11, 2015

- enable udp xuhao1 committed on Nov 11, 2015 a4a487a

Commits on Nov 10, 2015

- ? xuhao1 committed on Nov 10, 2015 1693199

Commits on Nov 9, 2015

- Adding UDP xuhao1 committed on Nov 9, 2015 45b189a

Commits on Sep 28, 2015

- Merge pull request #195 from roodddow/patch-1 retoria committed on Sep 28, 2015 db17d7a

**Commit History**

**CAN COPY OR REVERT TO ANY PAST STATE OF THE REPOSITORY**

change log updated Browse files

develop 0.7.13

rtoris committed on Aug 14, 2015 1 parent 3cfc76b commit a2b3f869a67eeab84d17a358d5346b464731544a

Showing 4 changed files with 36 additions and 0 deletions. Unified Split

**Change log**

***YOU CAN VIEW ALL CHANGES  
MADE BETWEEN CONSECUTIVE COMMITS***

```
5 rosapi/CHANGELOG.rst
@@ -23,8 +23,11 @@ Changelog for package rosapi
23 23 0.7.0 (2014-12-02)
24 24 -----
25 25 +
26 26 +0.7.13 (2015-08-14)
27 27 +
28 28 +* Fix catkin_lint issues
29 29 +* Contributors: Matt Vollrath
30 30 +
26 31 0.7.12 (2015-04-07)
27 32 -----
28 33

14 rosbridge_library/CHANGELOG.rst
@@ -34,8 +34,28 @@ Changelog for package rosbridge_library
34 34 * request_id -> id
35 35 * Contributors: Russell Toris
36 36
37 37 +0.7.13 (2015-08-14)
38 38 -----
39 39
40 40 +* Nevermind a_D
41 41 +* Add test_depend too (just in case)
42 42 +* Add dependency on python bson
43 43 +* Get parameter at encode time
```

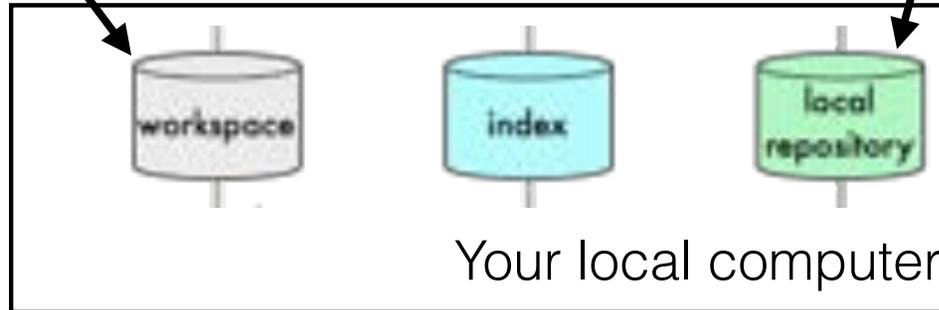
How does git work?

# Git Data Transport Commands

<http://osteale.com>

The directory where you are working

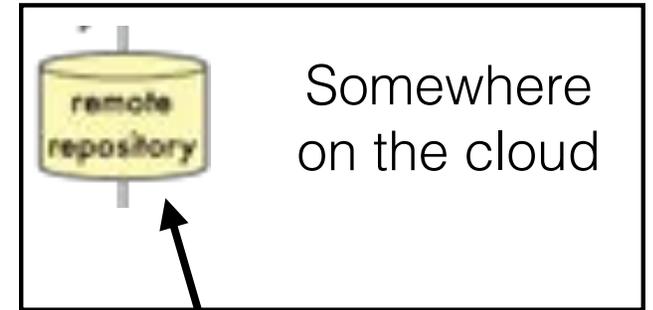
( ~/uname/reponame or  
C:\Users\uname\reponame )



Your local computer

The repository on your local computer

( ~/uname/reponame/.git )



The repository on a remote server

( <http://github.com/username> )

# Git Data Transport Commands

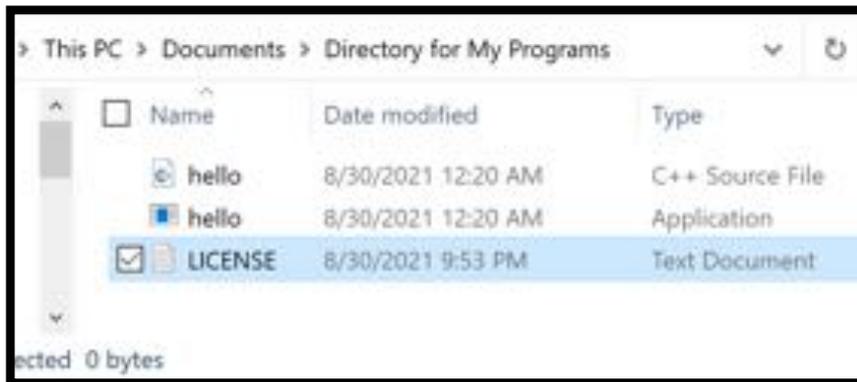
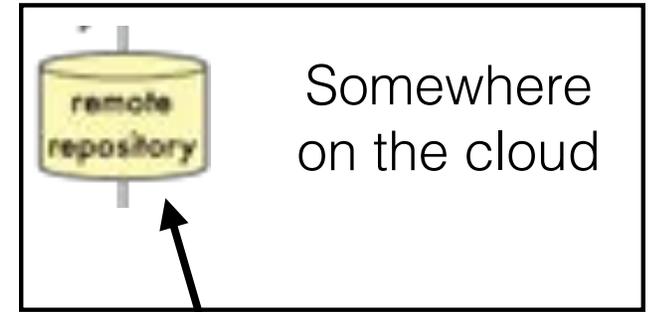
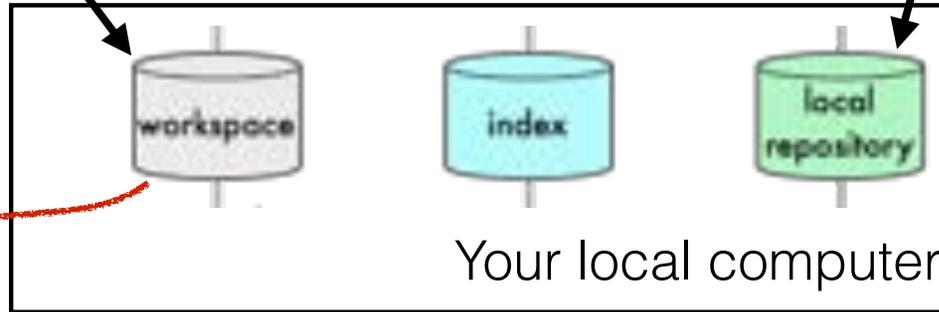
<http://osteele.com>

The directory where you are working

( ~/uname/reponame or C:\Users\uname\reponame )

The repository on your local computer

( ~/uname/reponame/.git )



The repository on a remote server

( <http://github.com/username> )

# Git Data Transport Commands

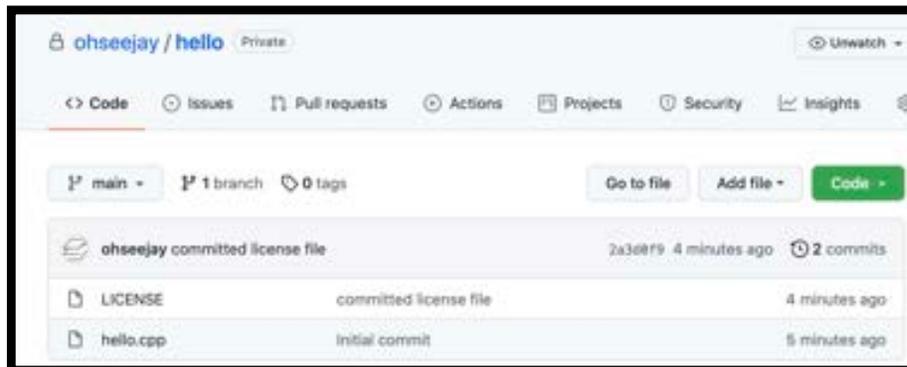
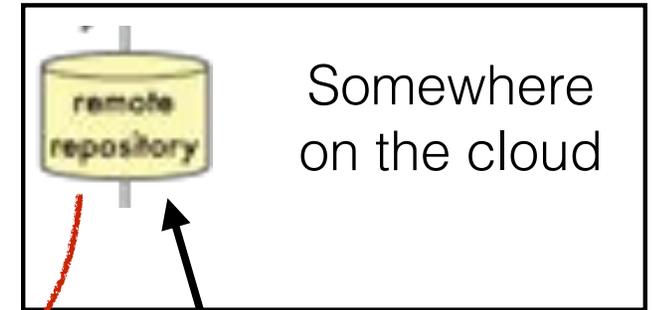
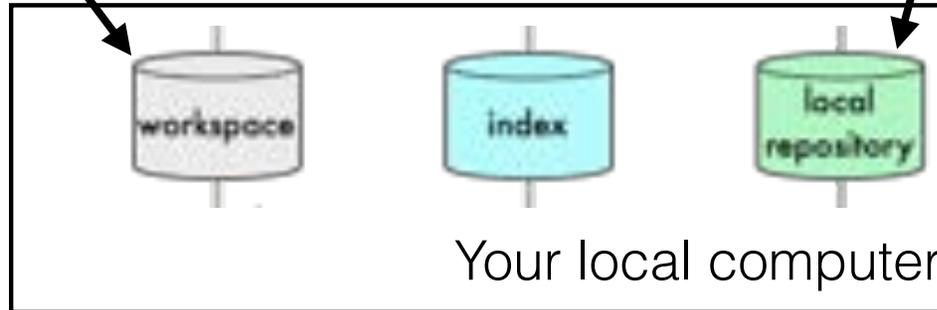
<http://oosteele.com>

The directory where you are working

( ~/uname/reponame or C:\Users\uname\reponame )

The repository on your local computer

( ~/uname/reponame/.git )

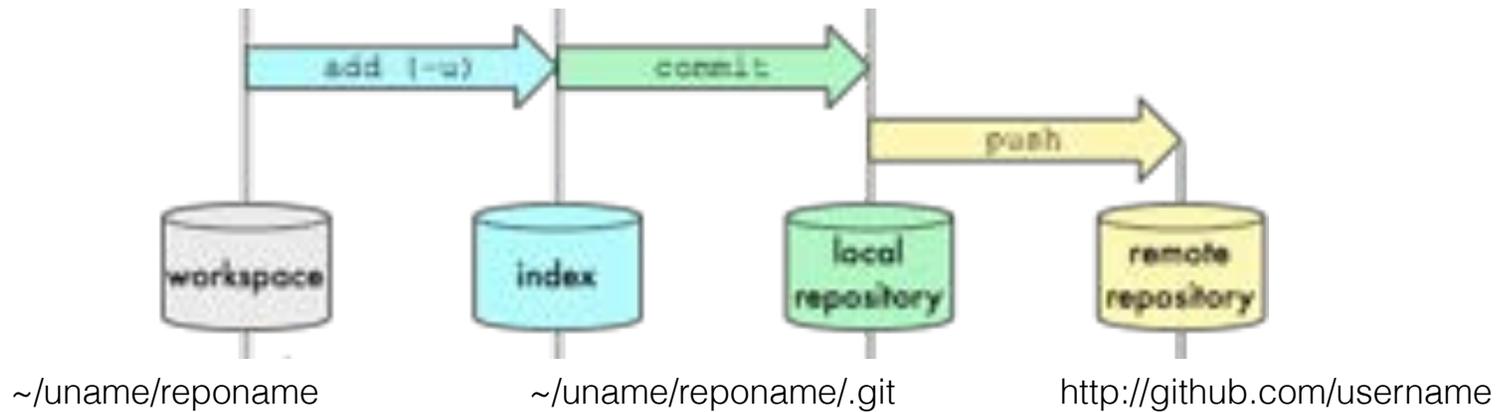


The repository on a remote server

( <http://github.com/username> )

# Git Data Transport Commands

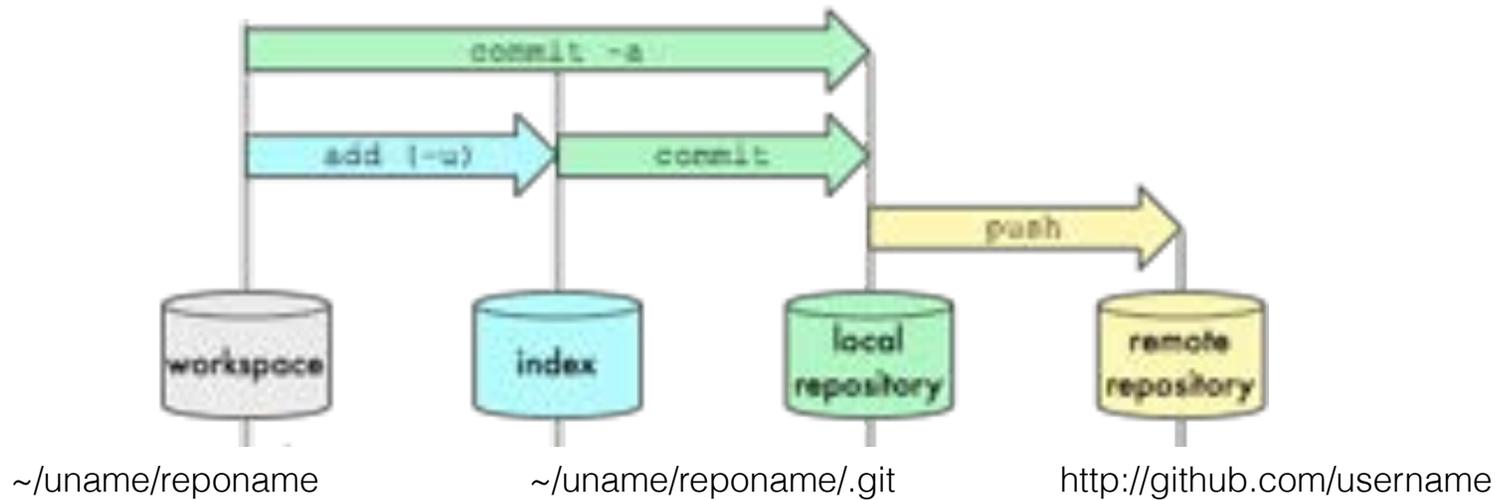
<http://osteale.com>



After making local changes, you can add, commit, and push to your remote repository

# Git Data Transport Commands

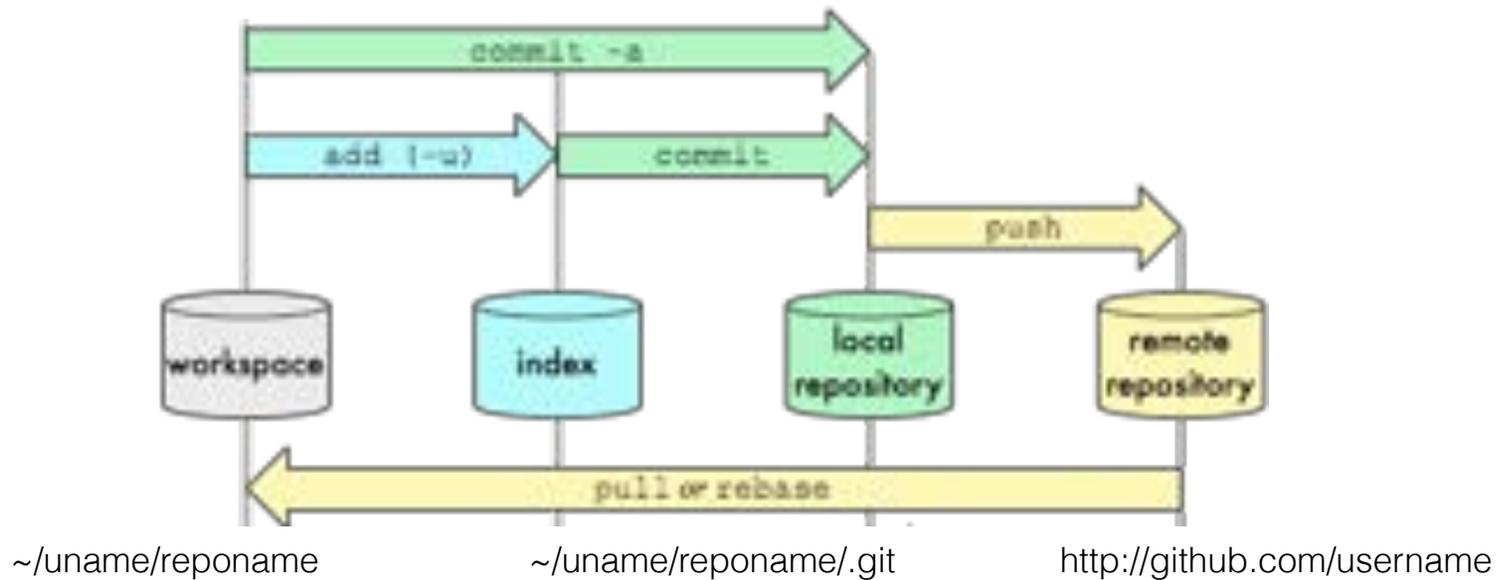
<http://osteale.com>



If there are no files to add, just commit and push

# Git Data Transport Commands

<http://osteale.com>



A pull command updates the local workspace with changes from the remote repository

# git basics: commands

- Push completed project to repository (or just to update)
  - add files to a repository: `git add <file listing>`
  - commit changes to local repo: `git commit -a -m "<msg>"`
  - push local changes to a remote repository: `git push`
- Pull to updates your local repository (and workspace) from remote
  - pull remote changes to a local repository: `git pull`

# **Our first challenge: Project 0**

# Our first challenge: Project 0

## Pocket Calculator



# Our first challenge: Project 0

## Pocket Calculator

```
$ ./calculator  
Please type a number and press enter: 100  
Please type a math operator (one of: + - * /): +  
Please type a number and press enter: 2  
100+2= 102
```

**Actually,  
it will look more like this**

# Next Lecture: Operators and Variables

```
#include <iostream>

/*
   Let's write a calculator program
*/

int main()
{
    std::cout << "What is 100 plus 2?" << "\n";
}
```

**calculator.cpp (Version 00)**