CIS241

System-Level Programming and Utilities

C

Erik Fredericks, frederer@gvsu.edu

Fall 2025

> **Based on material provided by Erin Carrier, Austin Ferguson, and Katherine Bowers**

But why?

- **Help develop a better understanding of what high (and low) level langauges are doing!**
- **Understand memory management**
  - i.e., Java/Python isn't handling it "for you"
- **System programming - interact with OS**

C vs. Java

Both are compiled, but to different things

- C: compiled to **assembly language**
- Java: compiled to Java bytecode (executed by JVM)
- Python: interpreted (unless if you're compiling to `pyc`)

Portability:

- **Java: more portable**
  - Bytecode executed by same JVM version on any machine
- **C: less portable**
  - Compiled code is **architecture dependent**

C vs. Java

Speed:

- C is generally faster than Java and Python
  - Java can be fast (JIT makes optimizations, Cython helps Python, etc.)

Compiled vs. Interpreted?

Compiled:

- C: `gcc program.c`
- Java: `javac program.java`

Interpreted (run without compiling)

- bash: `bash script.sh`
- Python: `python program.py`
- awk: `gawk program.awk`

Getting started with C

Hello world!

```c
#include <stdio.h>

int  main(void)
{
        printf("Hello world\n");
        return  0;
}
```

NOTE THE SEMI-COLONS!!!

C preprocessor

Performs source code substitution!

`#` indicates a preprocessing directive

`include` - replaces a line with contents of file

- Typically used for header files
- Essentially "importing"

```
define PI 3.14159
```

- Anywhere you have `PI`, replace with value
- Convention - use all caps

Running C code

Compiling:

- `gcc program_file.c`
  - Generates an `a.out` file
- `gcc -o myprogram program_file.c`
  - Generates a `myprogram` file

Running

- `./a.out`
- `./myprogram`

DON'T DO THIS

```
gcc -o project.c project.c
```

Why?

If you don't have a C compiler

(Including update/upgrade)

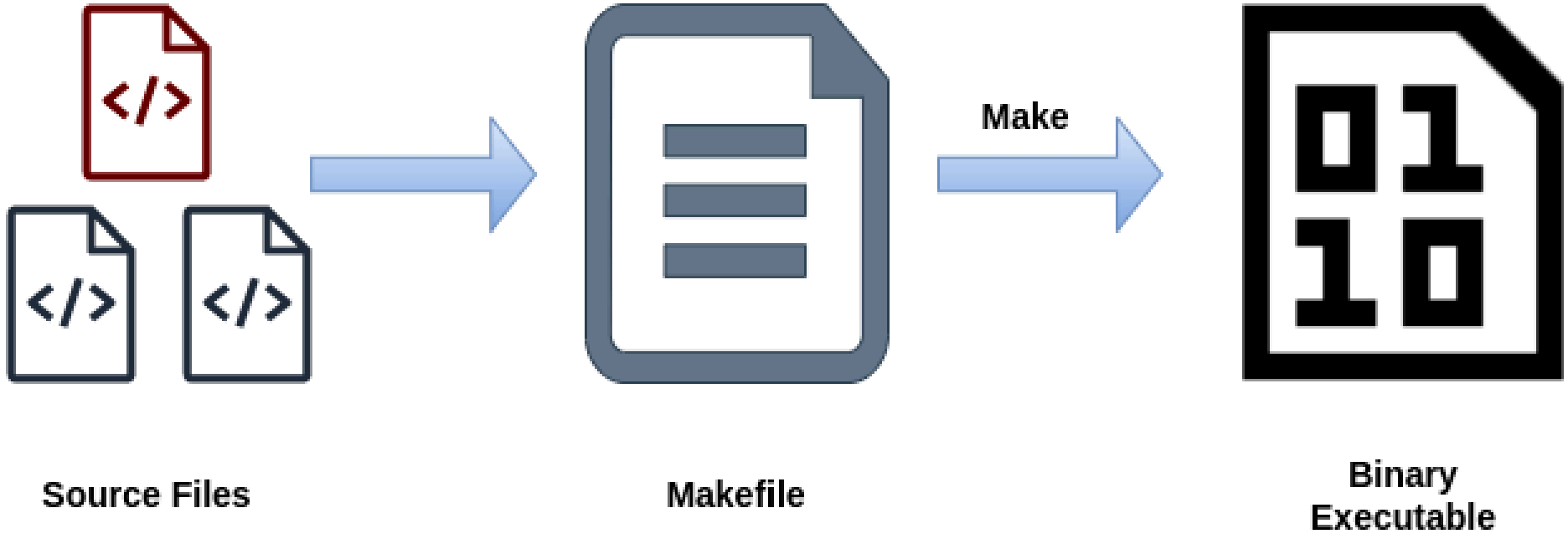
WSL (Ubuntu): `sudo apt update && sudo apt upgrade && sudo apt install gcc`

- (or more broadly, `sudo apt install build-essential`)

Potentially:

Mac (Homebrew):

- `brew update`
- `brew upgrade`
- `brew info gcc`
- `brew install gcc`
- `brew cleanup`
- https://discussions.apple.com/thread/8336714?sortBy=rank


Make
Source Files
Makefile
Binary
Executable

Compiler flags!

- `-o` - specify output file
- `-O` - specify optimization level when compiling
  - Compiling goes to machine code
  - Some machine code is faster/uses less space (i.e., optimized)
  - Careful - high optimization levels can **break** code!
- `-Wall` - enables warning messages

https://gcc.gnu.org/onlinedocs/gcc/Optimize-Options.html

Sample `O` flags

- `-O` or `-O1` means "optimize some"
- `-O2` means "optimize more"
- `-O3` means "optimize as much as possible"
- `-Os` and `-Oz` means "optimize for space"

C++?

C++ is an object-oriented version of C

- Not stricly an "extension"

C is procedural

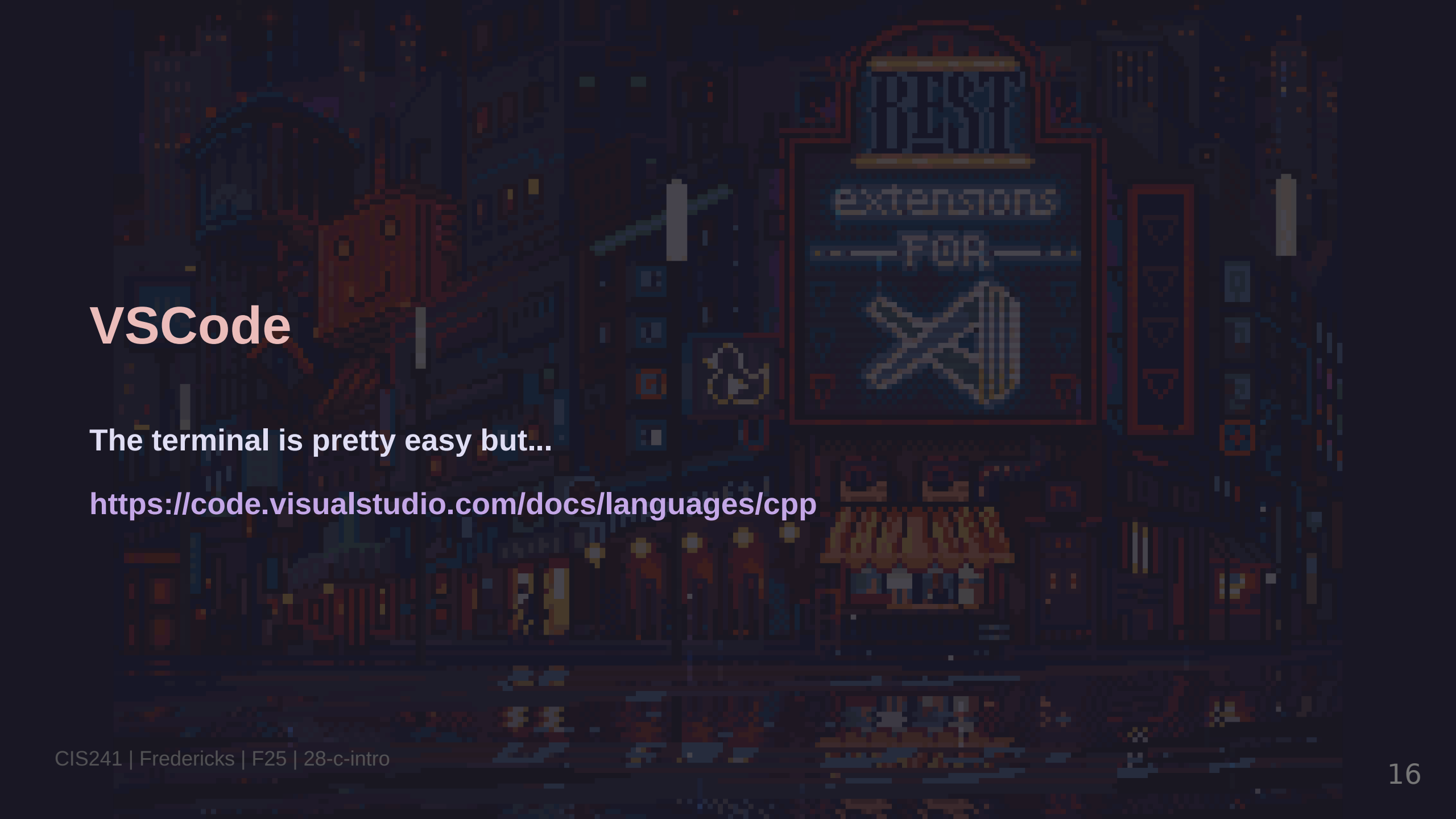
If you know C, you can do C++ (with extra effort)

Why is it called C?

Wikipedia says:

> **A successor to the program language B.**

https://en.wikipedia.org/wiki/C_(programming_language)



VSCode

The terminal is pretty easy but...

https://code.visualstudio.com/docs/languages/cpp