Programming in C++

Build Basics

Curtis Larsen

Utah Tech University—Computing

Spring 2025

Objectives

Objectives:

- Understand File Contents
- Compile Source to Object
- Build Object Archive
- Link Object to Executable
- Understand Makefile Format
- Create Makefile Contents

File Contents

File Types

Extension	Type	Purpose	Contents
`.h`	Header	Declarations	text
`.cpp`	Implementation	Definitions	text
`.o`	Object	Machine instructions	binary
`.a`	Archive/Library	Collections of object files	binary
	Executable	Complete program	binary

Header File

```
#ifndef _DECLARATIONS_H_
#define _DECLARATIONS_H_

int add(int a, int b);

class Score {
public:
  int getPoints() const;
  void setPoints(const int points);
private:
  int mPoints{};
};

#endif /* _DECLARATIONS_H_ */
```

Implementation File

```
#include "declarations.h"

int add(int a, int b) {
  return a + b;
}

int Score::getPoints() const {
  return mPoints;
}

void Score::setPoints(const int points) {
  mPoints = points;
}
```

Object File

```
...
0000000000000000 <_Z3addii>:
   0:        f3 0f 1e fa    endbr64
   4:        55             push   %rbp
   5:        48 89 e5       mov    %rsp,%rbp
   8:        89 7d fc       mov    %edi,-0x4(%rbp)
   b:        89 75 f8       mov    %esi,-0x8(%rbp)
   e:        8b 55 fc       mov    -0x4(%rbp),%edx
  11:        8b 45 f8       mov    -0x8(%rbp),%eax
  14:        01 d0          add    %edx,%eax
  16:        5d             pop    %rbp
  17:        c3             ret

0000000000000018 <_ZNK5Score9getPointsEv>:
...
```

Archive File

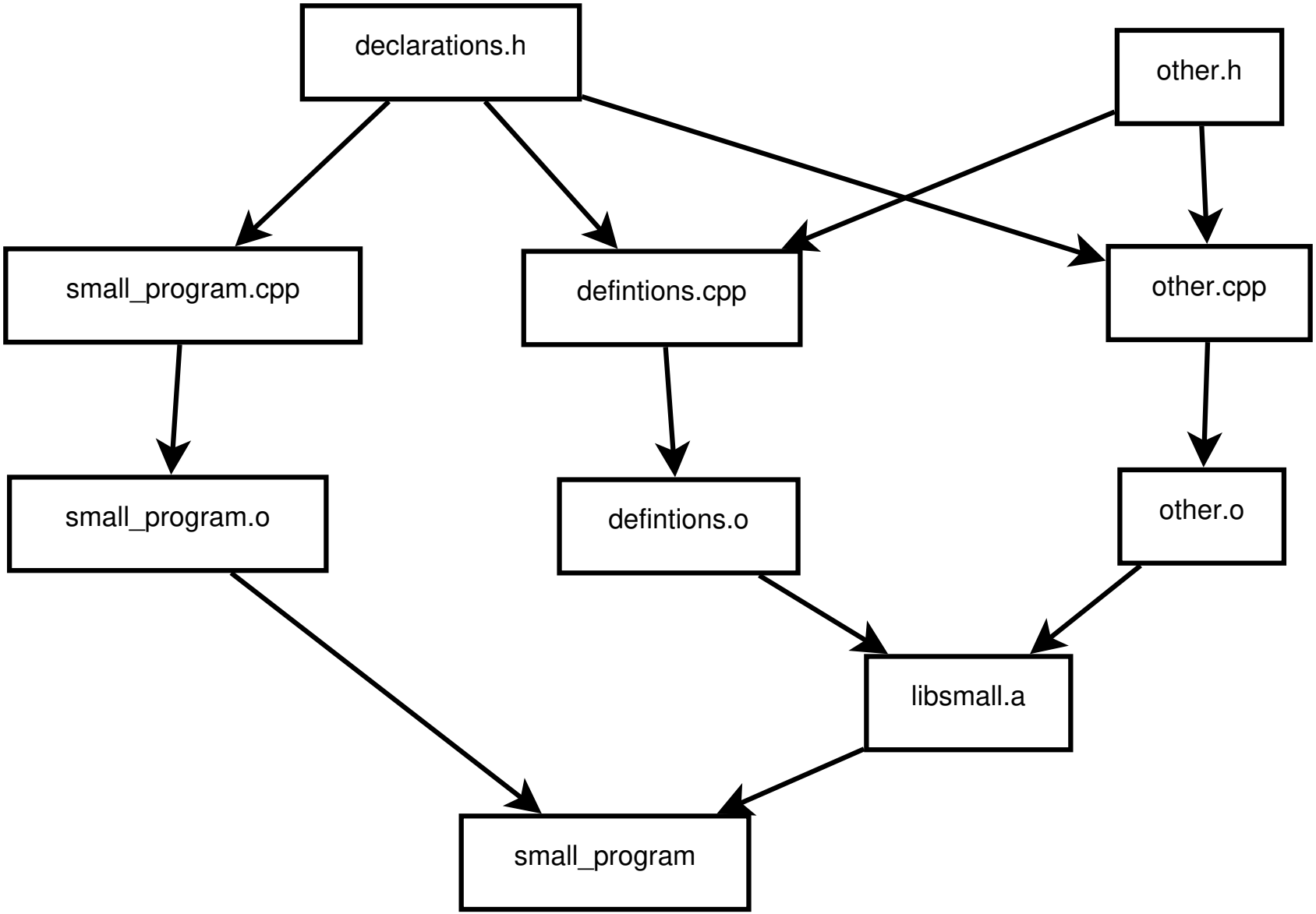
```
In archive libsmall.a:

definitions.o:     file format elf64-x86-64
rw-r--r-- 0/0   1496 Dec 31 17:00 1969 definitions.o
```

Executable File

```
...
0000000000001100 <_start>:
    1100:          f3 0f 1e fa endbr64
    1104:          31 ed       xor    %ebp,%ebp
    1106:          49 89 d1    mov    %rdx,%r9
    1109:          5e          pop    %rsi
    110a:          48 89 e2    mov    %rsp,%rdx
    110d:          48 83 e4 f0 and    $0xfffffffffffffff0,%rsp
    1111:          50          push   %rax
    1112:          54          push   %rsp
    1113:          45 31 c0    xor    %r8d,%r8d
    1116:          31 c9       xor    %ecx,%ecx
    1118:          48 8d 3d ca 00 00 00 lea    0xca(%rip),%rdi
    111f:          ff 15 b3 2e 00 00    call   *0x2eb3(%rip)
    1125:          f4          hlt
...
```

Build Executable

Build Process



Build Commands

Process	Command
Include	`#include "declarations.h"`
Compile	`g++ -o definitions.o -c definitions.cpp`
Archive	`ar crus libsmall.a definitions.o`
Link	`g++ -o small_program small_program.o -L. -lsmall`

make

Purpose

- Build flowchart is complicated.
- Build flowchart can get very large.
- Record flowchart for future reference.
- Automatically run commands based on flowchart.
- Uses time stamps to detect changes in flowchart.

Makefile Rule Structure

Generic Rule Format

```
target: dependency1 dependency2 ... dependencyN
<tab-indent>command1
<tab-indent>command2
<tab-indent>...
<tab-indent>commandN
```

Example Rule

```
definitions.o: definitions.cpp declarations.h
        g++ -o definitions.o -c definitions.cpp
```

`Makefile` Shortcuts

In the command section, you can use the following shortcuts:

- `$@` - target
- `$<` - first dependency
- `$^` - all dependencies

Default Target

- The first target in the `Makefile` is the default target.
- This is customarily set in the `Makefile` to `all`.

Example Rule

```
all: small_program
```

Clean Target

- We often define a `clean` target to clean up the build directory.
- This is used to remove any derived files created by the build process.
- Objects, libraries, and executables are all derived.

Example Rule

```
clean:
        -rm -f *~
        -rm -f *.o
        -rm -f small_program
        -rm -f libsmall.a
```

git Update

Derived Files and git

We don't add/commit any derived files in the git repository. Doing so can cause `make` to misbehave. Executables can be unexpectedly broken.

.gitignore

Creating a `.gitignore` file and listing the derived files will prevent `git` from adding and committing derived files.

Example `.gitignore`

```
small_program
*.o
*.a
```

Small Program Example

Tour Example Code

Review Sample Program