Lab 5: GNU Make

Comp Sci 1585

Data Structures Lab:
Tools for Computer Scientists

Outline

1 What is make?
Example makefile
Dependencies
Targets and .PHONY
Wildcards and special variables
Example
Patterns

What is `$ make`?

- `$ make` executes a program that can be used to create files (such as executables).
- It can detect what has changed between builds and only rebuild what is necessary.
- Makefiles are very common in large C and C++ projects.
- They can also be used to store project-related commands.
- GNU Make is a tool which controls the generation of executables, and other non-source files of a program, from the program's source files.
- `https://www.gnu.org/software/make/`

Outline

1 What is make?
Example makefile
Dependencies
Targets and .PHONY
Wildcards and special variables
Example
Patterns

Your First Makefile: `$ makefile-1`


```
program:
        g++ *.cpp -o program
```

Outline

1 What is make?
Example makefile
Dependencies
Targets and .PHONY
Wildcards and special variables
Example
Patterns

Dependencies: `$ makefile-2`


```
program: main.cpp funcs.h funcs.cpp
        g++ *.cpp -o program
```

Outline

1 What is make?
Example makefile
Dependencies
Targets and .PHONY
Wildcards and special variables
Example
Patterns

Multiple Targets: `$ makefile-3`


```
program: main.o funcs.o
        g++ main.o funcs.o -o program

main.o: main.cpp funcs.h
        g++ -c main.cpp

funcs.o: funcs.cpp funcs.h
        g++ -c funcs.cpp
```

Phony Targets: `$ makefile-4`


```
.PHONY: clean

program: main.o funcs.o
        g++ main.o funcs.o -o program

# %< --- SNIP --- >%

# - means "ignore errors from"
# @ means "don't print command"
clean:
        -@rm -f program
        -@rm -f *.o
```

Outline

1 What is make?
Example makefile
Dependencies
Targets and .PHONY
Wildcards and special variables
Example
Patterns

Store variable to use as arguments

- `var=value` sets values.
- `${var}` uses the value of the variable.
- `var=$(wildcard *.cpp)` stores the name of every file ending in `.cpp` in the variable `var`.
- `foo=$(var:%.cpp=%.o)` substitutes `.o` for `.cpp` in all the files in `var`.
- `target: var=thing` assigns `thing` to `var` when building `target` and its dependencies.

Outline

1 What is make?
Example makefile
Dependencies
Targets and .PHONY
Wildcards and special variables
Example
Patterns

Variables: $ makefile-5


```
CFLAGS = -Wall --pedantic-errors -O2

program: main.o funcs.o
	g++ ${CFLAGS} main.o funcs.o -o program

.PHONY:debug
debug: CFLAGS = -g -Wall --pedantic-errors
debug: program

main.o: main.cpp funcs.h
	g++ ${CFLAGS} -c main.cpp

funcs.o: funcs.cpp funcs.h
	g++ ${CFLAGS} -c funcs.cpp
```

Outline

1 What is make?
Example makefile
Dependencies
Targets and .PHONY
Wildcards and special variables
Example
Patterns

Patterns

- You can make pattern targets that describe how to build more than one file.
- As with substitution, you use `%` for the variable part of the target name.
- For example: `%.o: %.cpp` describes how to build any `.o` file from its matching `.cpp` file.
- `$@` holds the name of the target.
- `$<` holds the name of the first dependency.
- `$^` holds the names of all the dependencies.
- Automatic Variables: `https://www.gnu.org/software/make/manual/html_node/Automatic-Variables.html`

Patterns: $ makefile-6


```
SOURCES = $(wildcard *.cpp)
HEADERS = $(wildcard *.h)
OBJECTS = $(SOURCES:%.cpp=%.o)

CPP = g++
CFLAGS = -Wall --pedantic-errors -O2

program: ${OBJECTS}
	${CPP} ${CFLAGS} ${OBJECTS} -o program

%.o: %.cpp ${HEADERS}
	${CPP} ${CFLAGS} -c $<

.PHONY: clean
clean:
	-@rm -f program
	-@rm -f ${OBJECTS}
```

Miscellany

Command-Line Options:

- `$ make -j3` runs up to 3 jobs in parallel
- `$ make -B` makes targets even if they seem up-to-date.

Related programs:

- makedepend is a command for auto-generating dependencies in C and C++ projects.
- CMake can generate Makefiles and various IDE configurations for projects.