Make and CMake : Automating C++ Build Process

Deepak Aggarwal in Coding Blocks Follow
Dec 13, 2017 · 3 min read

Many of my students struggle to understand the C++ compilation when applied to multiple files that belong to same project.
A C++ project, *something that will give you one executable*, can consists of multiple files. The number could be as large as 1000 for complex projects like Adobe Photoshop or Google Chrome.
Now, if I give you this source code, would you be able to compile it and get an executable that you can double click; and the software starts running??? This would be painful *(if not difficult)* even for an expert.
What about if I give you a list of instructions and a program, that can execute these instructions and do this compilation for you. The end product would be an executable. Isn't it great. No! Its superb.
That is what we are going to learn here. How to write the instructions in a file that will be executed by a program to automate the build (or loosely compilation) process.

> *The file that we are going to write is named* Makefile.
> *The program we are going to use is called* make.

C++ Multi-file Programming

Let us consider a single program to divide 2 numbers. Files that we will require are :

1. div.h

```
#ifndef DIVISON_H
#define DIVISON_H

double divison(int, int);

#endif
```

medium-900f569a75db-div.h hosted with ❤ by GitHub view raw

2. div.cpp

```
#include "div.h"
double divison(int dividend, int divisor){
    if (divisor == 0){
        return -1;
    }

    return (double)dividend / divisor;
}
```

```

int main(){
    int x;  cin >> x;
    int y;  cin >> y;

    int quotient = division(x, y);
    cout << quotient;
    return 0;
}
```

medium-900f569a75db-main.cpp hosted with ❤ by **GitHub** view raw

If I want to compile these files and want to transform them into an exectable, i would write the following in terminal :

```
# Just translate the files.
# Result would be div.o main.o in the present working directory
g++ -c div.cpp
g++ -c main.cpp
# Do all the linking stuff
g++ div.o main.o -o divisonExecutable
```

. . .

Writing our Makefile

That's lot to remember. Now let's write an equivalent *Makefile*
Important : The statements starting with # are for humans and shall not be typed in the terminal. But do read them.

```
# Makefile
# Specify what I need in the end. One single executable
divisonExecutable : main.o div.o
# Read this as divisionExecutable depends on main.o div.o
# But how is it produced??? Hmm...using the below statement
    g++ main.o div.o -o divisonExecutable
# starts with tab, I repeat tab
#-----------------------------------------------------------------
# But main.o is not there? So specify how it is produced.
main.o : main.cpp div.h
    g++ -c main.cpp
# Same for test.o
test.o : test.cpp test.h
    g++ -c test.cpp
```

Yup! Just `make` . *make* looks for `makefile` , if not found, `Makefile` .
If you need to specify your own file name, you need to specify

```
make -f name-of-my-file-that-make-will-use
```

When you will run `make` , you can see the following in your pwd.

> *test.o*
> *main.o*
> *divisonExecutable*

If you want to learn more about `make` do visit these two articles which I found really helpful.

Makefile in Linux: An Overview
Small C/C++ applications with a couple of modules are easy to manage. Developers can recompile them easily by calling…
www.codeproject.com

How to create our own Makefile | Creating your own Makefile
Although this blog is not especially oriented to the area of development, but there are several concepts that are…
www.milesweb.com

. . .

CMake

Now what if writing `Makefile` is also tedious. For that we have `cmake` that will generate `Makefile` for us.

. . .

Hope this article helped you understand the `make` and `cmake` beyond commands.

Programming C Programming Build Concept

Deepak Aggarwal

Follow

Mentoring people to become better software developers.

Coding Blocks

Follow

Daily Tidbits on Android, Javascript and Machine Learning

See responses (2)

More From Medium

More from Coding Blocks

Related reads

Related reads

One stop guide to Google Summer of Code

Harshit... Jul 18,... 5K

Splitting a string in C++

Ben Key Oct 24,... 57

How to optimize C and C++ code in 2018

Iurii... Nov 30, 201... 573