Makefile note:

Header guards: prevent duplication on declarations and definition in the header file

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | #ifndef SOME\_UNIQUE\_NAME\_HERE  #define SOME\_UNIQUE\_NAME\_HERE    // your declarations (and certain types of definitions) here    #endif |

SOME\_UNIQUE\_NAME\_HERE: the convention is set to the full filename of the header file, in all caps, using underscores for space

Ls (no executable code)

G++ main.cpp message.cpp

Ls (see a.out)

./a.out

If thousands of files need compiled, time is also very long.

Key idea: Only compile something that changed.

Makefile structure: white space sensitive

Target: dependencies

(tab) action

1 output: main.o message.o

2 g++ main.o message.o -o output

3

4 main.o: main.cpp

5 g++ -c main.cpp

6

7 message.o: message.cpp message.h

8 g++ -c message.cpp

9

10 clean:

11 rm \*.o output

-c means do not generate executable, just object code.

Ls

Make

Ls (.o files and executable)

./output

Make(again, no change)

Clean

Ls

Make(again)

Ls

./output

Change message.cpp: Hello replace Makefile example

Make (no change of main)

./output

Change main.cpp: touch main.cpp (timestamp is changed)