Header files

What is header files

A Header files is a file with extension **`.h`** containing C declarations, global variables, data type definitions and macro definitions to be shared between several source files. They can be inserted into other files by the preprocessing directive **`#include`**.

Why do we need header files

In some large programs, there are always some global variables, type definitions, functions and macro definitions shared by several source files. Before we use these things, we should declared them in order to judge whether we use it correctly. But declaring it in all files brings some troubles when we build a large program. Using header files is a convenient mechanism to solve those troubles and it has several advantages.

1. Using header files can easily change the programs when needed (only one master file to change)

Suppose we have a function named **`func`** in one file and we use it in several source files. We should declare it before using the function in each file.

```c
// file1.c
void func(void);

void f1() { func(); }
```

```c
// file2.c
void func(void);

void f2() { func(); }
```

So we should present the prototype in all the files that use the function. If the function is modified, we must change all the declarations in those files which may cost much time. A solution to this problem is using header files. We gather the prototypes in a header file. By this convenient mechanism, we only need to change in one place.

```c
// func.h
void func(void);
```

```c
// file1.c
#include "func.h"
void f1() { func(); }
```

```c
// file2.c
#include "func.h"

void f2() { func(); }
```

2. header files separate the interfaces and the implementations, which reduces the compilation time.

How can we use header files

The preprocessor directive **#include**

Both user and system header files are included using the preprocessor directive **#include**. Including a header file is equal to copy the code of header file. It has two variants:

- **#include <file>** :

  This variant is used for system files. It searches for a file named file in a standard list of system directories.

- **#include "file"** :

  This one is used for header files of your own program. It searches for a file named file first in the directory containing the current file, then in the quote directories and then the same directories used for **<file>**.

Include only once

Including a header file in several files may cause the result of copying the code in this header file several times. In order to avoid this situation, we need some measures.

Suppose we have a header file name **foo.h** whose contents are

```c
// foo.h
void f1();
void f2();
```

We can include it only once by using the preprocessor directive **#ifndef** as follow:

```c
// foo.h
#ifndef FOO_H_
#define FOO_H_ // define the macro

void f1();
void f2();

#endif // FOO_H_
```

One of the drawbacks of `#ifndef` is that it may have some troubles when the same macro defined in several different header files. Except the first header file, the contents in other header files will be ignore.

Another way to avoid multi-include is to use `#pragma once`, but it also has some disadvantages. This directive can only for the full file because it cannot work for a code fragment. It's not a cross-platform directive. It is not supported for the early compiler.

Reference

- https://en.wikipedia.org/wiki/Include_directive
- https://www.tutorialspoint.com/cprogramming/c_header_files.htm
- https://gcc.gnu.org/onlinedocs/cpp/Header-Files.html
- https://stackoverflow.com/questions/333889/why-have-header-files-and-cpp-files?noredirect=1
- https://stackoverflow.com/questions/1305947/why-does-c-need-a-separate-header-file
- http://www.cnblogs.com/Braveliu/archive/2012/12/29/2838726.html
- http://www.umich.edu/~eecs381/handouts/CHeaderFileGuidelines.pdf