**Preprocessor Directives and Guards**

**Preprocessor Directives**

Preprocessor directives are lines included in the code of our programs that are not program statements but directives for the preprocessor. These lines are always preceded by a hash sign (#). The preprocessor is executed before the actual compilation of code begins, therefore the preprocessor digests all these directives before any code is generated by the statements

**Conditional Inclusions**

**e.g. #ifndef / #endif or #pragma once**

These directives allow the discarding of part of the code of a program if a certain condition is met.In other words, if you have **included** this stuff already, **do not include it again**. This avoids redefinition errors.

The intent is to include the contents of the header file in the translation unit only once, even if the physical header file is included more than once. This can happen, for example, if you include the header directly in your source file, and it's also indirectly included via another header.

Putting the #ifndef wrapper around the contents means the compiler only parses the header's contents once, and avoids redefinition errors.

Some compilers allow "#pragma once" to do the same thing, but the #ifndef construct works everywhere. Pragmas are machine- or operating system-specific by definition, and are usually different for every compiler.

**Best Practice**

Not only should you use a unique and predictable (internal) include guard but you should also consider using (external) include guards around each preprocessor include directive in header files. It makes little difference on a small project, but a large difference on big ones.