Compiler error message metaprogramming: Helping to find the conflicting macro definition

devblogs.microsoft.com/oldnewthing/20211206-00

December 6, 2021

Raymond Chen

Say you want to require that a preprocessor macro is set a particular way:

```
#include <contoso.h>
#if CONTOSO_VERSION != 314
#error This header file requires version 314.
#endif
```

Okay, if the version isn't set correctly, you will indeed get the error, but that doesn't help the user much with figuring out why the version number is incorrect.

You might try this:

```
#include <contoso.h>
#if CONTOSO_VERSION != 314
#error This header file requires version 314 \
(got CONTOSO_VERSION instead)
#endif
```

Unfortunately, it doesn't work:

```
// MSVC
fatal error C1189: #error:  This header file requires version 314 (got
CONTOSO_VERSION instead)

// gcc
error: #error This header file requires version 314 (got CONTOSO_VERSION instead)

// clang
error: This header file requires version 314 (got CONTOSO_VERSION instead)

// icc
error: #error directive: This header file requires version 314 (got CONTOSO_VERSION
instead)
```

None of them substitute the macro in the error message, so you don't see what version you actually got.

Here's the trick: Just redefine the symbol.

```
#include <contoso.h>
static_assert(CONTOSO_VERSION == 314,
              "This header file requires version 314.");
#define CONTOSO_VERSION 314

// MSVC
error C2338: This header file requires version 314.
warning C4005: 'CONTOSO_VERSION': macro redefinition
C:\contoso\v271\contoso.h(5): note: see  previous definition of 'CONTOSO_VERSION'

// gcc
warning: "CONTOSO_VERSION" redefined
  3 | #define CONTOSO_VERSION 314

in file included from widget.h:1:
/contoso/v271/contoso.h:5: note: this is the location of the previous definition
  5 | #define CONTOSO_VERSION 271

error: static assertion failed: This header file requires version 314.
  2 | static_assert(CONTOSO_VERSION == 314,

// clang
warning: 'CONTOSO_VERSION' macro redefined [Wmacro-redefined]
#define CONTOSO_VERSION 314

/contoso/v271/contoso.h:5:9: note: previous definition is here
#define CONTOSO_VERSION 271

error: static_assert failed due to requirement '271 == 314' "This header file
requires version 314."
static_assert(CONTOSO_VERSION == 314,

// icc
error: static assertion failed with "This header file requires version 314."
static_assert(CONTOSO_VERSION == 314,

warning #47: incompatible redefinition of macro "CONTOSO_VERSION" (declared at line 5
of "/contoso/v271/contoso.h")
#define CONTOSO_VERSION 314
```

All of the major compilers provide the courtesy of telling you where the previous conflicting definition was made. From the error message, it is evident that we are including the `contoso.h` header file from the `v271` directory, when we presumably meant to include from the `v314` directory. The thing to investigate, therefore, is the project configuration where the include directories are specified, and fix it so that the correct version of `contoso.h` gets included.

This trick takes advantage of the fact that the C and C++ languages both permit a macro symbol to be defined twice, provided that the second definition is identical to the first. If not, then the program is *ill-formed* and requires a diagnostic.

So we just redefine the macro to the value we want. If it has that value, then everything is great. If not, then a diagnostic is required. And all of the major compilers point you at the previous definition, which will help you figure out why there is a conflict.

Raymond Chen

Follow