The Resource Compiler's preprocessor is not the same as the C preprocessor

devblogs.microsoft.com/oldnewthing/20171004-00

October 4, 2017

Raymond Chen

A customer had a project written in Visual C++, and the Build step failed with an error from the Resource Compiler:

```
Fatal error RC1015: cannot open include file 'vcruntime.h'
```

For reasons known (or perhaps not) only to the original developer of the project, this particular RC file had an enormous number of `#include`d header files, with many levels of nesting. Most of the header files employed the `#pragma once` directive to avoid multiple inclusion. But upon reading of the documentation for the Resource Compiler, it seems that the `RC.EXE` compiler doesn't recognize `#pragma once`.

The customer was able to replicate the problem by creating a pair of mutually-including header files:

```
// file1.h
#include "file2.h"

// file2.h
#include "file1.h"
```

which produced the same "cannot open include file" error message.

From this investigation, we are fairly confident that this is the root cause of the original error message. If this had been a problem with include files arise in C++ code, we could have used the `/showincludes` command line switch to troubleshoot the problem, but the `RC.EXE` compiler does not provide any switches for diagnosing problems with include files.

The customer wanted us to confirm their conclusions, and also to indicate whether the behavior with `RC.EXE` is by design, or whether they should file a defect report.

The Resource Compiler's preprocessor is not the same as the C preprocessor, even though they superficially resemble each other. In particular, `#pragma` is conspiciously missing from the table of supported preprocessor directives.

In order to get the effect of `#pragma once` in the Resource Compiler, you need to use the old-fashioned include guard technique.

Or better would be to avoid including so much junk. Many header files use `#ifdef RC_INVOKED` to detect whether they are being included by the Resource Compiler. When included by the Resource Compiler, they define only the identifiers needed for resource files and skip over all the other junk.

Bonus chatter: The Resource Compiler differs from the C preprocessor in another significant way: As noted on the same page that has the list of preprocessor directives, if a file has the extension `.c` or `.h`, then the Resource Compiler ignores all lines that aren't preprocessor directives. If you want to include another file that has resource content, you need to give it another extension. Typically, that extensions is `.rc`, although you may find the extension `.dlg` in older code.

Bonus bonus chatter: The Resource Compiler has *two* expression evaluators. The one used by the preprocessor follows the C language rules. But the one used by the resource parser follows its own weird rules. For example, in resource files, you can say `3 | NOT 2`, which is equivalent to `3 & ~2`. It means that in resource files, the `|` operator is not symmetric!

Raymond Chen

Follow