Does the Resource Compiler have a separate preprocessor or doesn't it?

devblogs.microsoft.com/oldnewthing/20240910-00

September 10, 2024

Some time ago, I noted that the Resource Compiler's preprocessor is not the same as the C preprocessor. Michal Necasek at the OS/2 Museum (highly recommended) took issue with that statement, noting that the strings in the `rcpp.exe` program reveal that it seems to be basically the same C preprocessor book with a different cover. So who's right?

We're both right.

The standalone `rcpp.exe` program may very well have begun as the C preprocessor, but it existed only in the 16-bit SDK. The 32-bit Resource Compiler uses a built-in preprocessor; there is no 32-bit `rcpp.exe`.

Even though a lot of the preprocessing machinery is the same, the two have diverged. For example, the only `#pragma` directive supported by the Resource Compiler is `#pragma code_page(n)`, which is a pragma not supported by the C preprocessor at all.

Bonus chatter: Even though you can ask the C compiler to produce a preprocessed file, the C compiler internally doesn't write the output to a file and then read it back. That is just a waypoint in the overall process of compiling. Internally, the tokens in the preprocessed output have hidden attributes (known informally in standards circles as "blue paint") that don't show up in the output, so it is not strictly the case that taking the preprocessed output and feeding it back into the compiler is a full fidelity representation of the output of the preprocessor step.