If you wished a language supported the preprocessor, you know, you can fix that

devblogs.microsoft.com/oldnewthing/20090724-00

July 24, 2009

Raymond Chen

A customer had the following question about the message compiler, something that I had noted almost nobody uses. Well how do you do, we found somebody who actually uses it. Anyway, the question went like this (paraphrased, as always):

> Can I use symbolic constants in my .mc file? For example, I have a message file that goes like this:
>
> ```
> SymbolicName=ERROR_XYZ_TOO_LARGE
> The XYZ parameter cannot exceed 100.
> .
> SymbolicName=ERROR_ABC_TOO_SMALL
> The ABC parameter must be at least 1.
> .
> SymbolicName=ERROR_CANT_COMBINE_ABC_WITH_XYZ
> You cannot specify values for both ABC and XYZ.
> .
> ```
>
> I have symbols defined in a header file `#define MINIMUM_ABC_VALUE 1` and `#define MAXIMUM_XYZ_VALUE 100` that I, of course, have to keep in sync with the error messages. One way to do this is to change the messages:
>
> ```
> SymbolicName=ERROR_XYZ_TOO_LARGE
> The XYZ parameter cannot exceed %1!d!.
> .
> SymbolicName=ERROR_ABC_TOO_SMALL
> The ABC parameter must be at least %1!d!.
> .
> ```
>
> And in my function that prints error messages, I can insert these magic parameters:
>
> ```
> error = DoMyThing(...);
> if (error != ERROR_SUCCESS) {
>  switch (error) {
>  case ERROR_ABC_TOO_SMALL:
>   Insertion = MINIMUM_ABC_VALUE;
>   break;
>  case ERROR_XYZ_TOO_LARGE:
>   Insertion = MAXIMUM_XYZ_VALUE;
>   break;
>  case ERROR_CANT_COMBINE_ABC_WITH_XYZ:
>   Insertion = 0; // not used
>   break;
>  ... repeat for other error messages...
>  }
>  DWORD_PTR Parameters[1] = { Insertion };
>  FormatMessage(FORMAT_MESSAGE_ARGUMENT_ARRAY ...
>      ..., error, ..., (va_list*)&Parameters)...
> }
> ```
>
> This is obviously a rather high-maintenance approach. Is there some way I could just write, say,

> ```
> SymbolicName=ERROR_XYZ_TOO_LARGE
> The XYZ parameter cannot exceed {MAXIMUM_XYZ_VALUE}.
> .
> SymbolicName=ERROR_ABC_TOO_SMALL
> The ABC parameter must be at least {MINIMUM_ABC_VALUE}.
> .
> ```
>
> and have the message compiler do the substitution? It would be great if it could even take the values from my header files.

This is a case of standing right next to the answer and not even realizing it.

There’s no law that says that you’re not allowed to use any other tools. It so happens that the preprocessor is a handy tool. If you want the preprocessor to run over your message files before they go into the message table, then why not run the preprocessor over your message files before they go into the message table?

```
#include "qqlimits.h" // pretend the program's name is "qq"
...
SymbolicName=ERROR_XYZ_TOO_LARGE
The XYZ parameter cannot exceed MAXIMUM_XYZ_VALUE.
.
SymbolicName=ERROR_ABC_TOO_SMALL
The ABC parameter must be at least MINIMUM_ABC_VALUE.
.
SymbolicName=ERROR_CANT_COMBINE_ABC_WITH_XYZ
You cannot specify values for both ABC and XYZ.
.
```

Give this file a name like, say, `qq.mcp`, and add a rule to your makefile:

```
qq.mc: qq.mcp qqlimits.h
  cl /EP qq.mcp >qq.mc
```

Make your changes to `qq.mcp`, and when you build, the makefile will preprocess it and generate the `qq.mc` file, which you can then compile with the message compiler just like you were doing before.