How do I decode a #pragma detect_mismatch error?

devblogs.microsoft.com/oldnewthing/20220427-00

April 27, 2022

Raymond Chen

Some time ago, I mentioned that you can use `#pragma detect_mismatch` to get the linker to verify that everybody agreed on a setting. This is typically used to avoid violations of the C++ One Definition Rule (ODR), which often fall into the category known in the standard as "ill-formed, no diagnostic required" (IFNDR), meaning "You broke the rules, and the compiler is not required to tell you that you broke the rules."

Okay, so you use the `#pragma detect_mismatch` trick as described in the earlier article to detect these mismatches at link time. But how do you decode the error message?

```
widget.lib(viewer.obj) : error LNK2038: mismatch detected for 'Contoso threading':
value 'Single' doesn't match value 'Multi' in gadget.lib(printer.obj)
```

Let's take apart the error message.

- `widget.lib(viewer.obj)` : This is identifying one of the files involved in the conflict. If the file is part of a library, the name is given as the library name, with the file in parentheses. If the file is being consumed directly, it will just be the file itself. In this case, the file is `viewer.obj` (presumably compiled from `viewer.cpp`) in the `widget.lib` library.
- `Contoso threading` : This is the name of the key that was passed as the first argument to the `#pragma detect_mismatch`.
- `Single` : This is the second parameter passed to `#pragma detect_mismatch`, as provided in `viewer.obj`.
- `Multi` : This is the second parameter passed to `#pragma detect_mismatch`, as provided in `printer.obj`.
- `gadget.lib(printer.obj)` : This is the other file involved in the conflict. It takes the same form as above.

Let me annotate the error message.

`widget.lib(viewer.obj):`	first party to the conflict

`error LNK2038: mismatch detected for`	
`'Contoso threading':`	conflicting property
`value 'Single'`	first party value
`doesn't match`	
`value 'Multi'`	second party value
`in gadget.lib(printer.obj)`	second party to the conflict

In other words, when you compiled `printer.cpp` to produce `printer.obj` in `gadget.lib`, somebody did a `#pragma detect_mismatch("Contoso threading", "Single")`. Looking at the implementation of the header file, we see that this happens if `SINGLE_THREADED` is defined.

On the other hand, when you compiled `viewer.cpp` to produce `viewer.obj` in `widget.lib`, somebody did a `#pragma detect_mismatch("Contoso threading", "Multi")`. Looking at the implementation of the header file, we see that this happens if `SINGLE_THREADED` is *not* defined.

This conflict results in an inconsistency between the two versions of the `Contoso` class, and that's what the error message is telling you.

Your job now is to resolve the conflict. Maybe you need to edit the `printer.cpp` and `viewer.cpp` files so that they agree on whether the `Contoso` object is single-threaded or multi-threaded.

Or maybe you split the `Contoso` class into two classes, one single-threaded and one multi-threaded, and let `Contoso` be an alias for one or the other. This lets each component use the `Contoso` they prefer, but it also means that they cannot pass `Contoso` objects between each other, since they aren't the same object any more.

How to resolve the conflict is your call. I'm just here to interpret the error message for you.

Raymond Chen

Follow