Other uses for the -library flag of the C++/WinRT compiler

devblogs.microsoft.com/oldnewthing/20201119-00

November 19, 2020

Raymond Chen

Last time, we saw that you can add the `-library` flag to the `cppwinrt.exe` command line if you intend to combine static libraries into a single C++/WinRT module.

But the flag can be used even if there are no static libraries involved.

Once you use the flag to become in charge of implementing `winrt_ get_ activation_ factory` and `winrt_ can_ unload_ now`, you can add other bonus code to the functions to add custom behavior.

If we use the enigmatic name `-library I`, then the original version will be prefixed with `I`, which is a bit of an English grammar hack, like the Visual Basic language keywords `Me` and `My`.

```
bool __stdcall winrt_can_unload_now() noexcept
{
  bool unload = I_can_unload_now();
  if (unload) {
    FlushLoggingBuffers();
  }
  return unload;
}
```

In this example, we flush the logging buffers when it looks like we're about to be unloaded. This is code that we would have to do anyway in `DLL_ PROCESS_ DETACH`, but we can do it here, so that the work is done outside the loader lock.

Note that returning `true` from `winrt_ can_ unload_ now` doesn't guarantee that you will be unloaded, so any work you do is speculative and may be a false alarm. You can flush the logging buffers, and possibly even close the handles, but you need to be ready for the case that the unload doesn't actually happen, and your logging code needs to be reopen the handles if they had been closed.

Injecting code into the `winrt_ get_ activation_ factory` can also be handy if you need to do some filtering or rerouting.

```
void* __stdcall winrt_get_activation_factory(
    std::wstring_view const& name)
{
  if (use_old_media_player &&
      name == L"Contoso.MediaPlayer") {
    return v1_get_activation_factory(name);
  } else {
    return I_get_activation_factory(name);
  }
}
```

In the above example, the `Contoso. MediaPlayer` was completely rewritten between versions 1 and 2, and the implementation wants to preserve the old version as a compatibility fallback. If the DLL has decided to use the old media player, then requests for the media player will produce an object from the old implementation. Otherwise, the DLL is using the new media player, or the request is not even for the media player at all, in which case we get the object from the current implementation.

Raymond Chen

Follow