Restoring support for pre-standardization coroutine free awaiters for even older code bases

devblogs.microsoft.com/oldnewthing/20211108-00

November 8, 2021

Raymond Chen

A customer had a large code base that was originally written in C++/CX, and they have been gradually converting it to C++/WinRT. They managed to get most of it converted, but the XAML binding is a major obstacle because you have to do it all at once, and they have a lot of XAML.

They found that with the upgrade to version 2.0.200729.8 of C++/WinRT, they lost the ability to `co_await` an `IAsyncAction^`. What they discovered is that C++/WinRT dropped support for pre-standardization coroutine free awaiters. They tried the modernizer from the article, but couldn't get it to work.

Okay, the first mistake is mine. I inadvertently relied upon a C++20 feature in the `make_cpp20_await_adapter` function:

```
template<typename Awaitable>
auto make_cpp20_await_adapter(Awaitable& awaitable)
{
    return cpp20_await_adapter<Awaitable>(awaitable);
}
```

This takes advantage of the treatment of a parenthesized list as an aggregate initializer if a class lacks a constructor. To get this code to work on C++17, we'll have to use the braced constructor instead.

```
template<typename Awaitable>
auto make_cpp20_await_adapter(Awaitable& awaitable)
{
    return cpp20_await_adapter<Awaitable>{ awaitable };
}
```

I've retroactively updated the article to incorporate this fix.

The last piece of the puzzle is realizing that some old versions of `pplawait.h` use free awaiters and therefore require modernization. (Version 14.16.27023 uses free awaiters, but version 14.29.30133 uses standard awaiters. I don't know exactly when the change

happened.)

If you can't upgrade to a newer version of `pplawait.h`, you can add modernizers:

```
namespace Windows::Foundation
{
    auto operator co_await(IAsyncAction^ x)
    {
        return ::modernizer::make_cpp20_await_adapter(x);
    }

    template<typename T>
    auto operator co_await(IAsyncOperation<T>^ x)
    {
        return ::modernizer::make_cpp20_await_adapter(x);
    }

    template<typename P>
    auto operator co_await(IAsyncActionWithProgress<P>^ x)
    {
        return ::modernizer::make_cpp20_await_adapter(x);
    }

    template<typename T, typename P>
    auto operator co_await(IAsyncOperationWithProgress<T, P>^ x)
    {
        return ::modernizer::make_cpp20_await_adapter(x);
    }
}
```

Raymond Chen

Follow