How do I create a Windows Runtime IRandomAccess-Stream around a bunch of bytes or a classic COM IStream?

devblogs.microsoft.com/oldnewthing/20241023-00

October 23, 2024

There are some Windows Runtime methods which expect a random access stream in the form of a `IRandomAccessStream`. On the other hand, you might have your data in the form of a memory block. For example, you might have a memory block that represents a bitmap, and you want to wrap it in a `IRandomAccessStream` so you can use it as a `SvgImageSource`.

One option is to create an `InMemoryRandomAccessStream` and use the `WriteAsync()` method to write the bytes (in the form of an `IBuffer`), then rewind the stream back to the start.

```
// C++/WinRT

winrt::Buffer BufferFromBytes(winrt::array_view<uint8_t> bytes)
{
    winrt::Buffer buffer(bytes.size());
    memcpy(buffer.data(), bytes.data(), bytes.size());
    buffer.Length(bytes.size());
    return buffer;
}

winrt::IAsyncOperation<winrt::IRandomAccessStream>
    BytesToRandomAccessStream(winrt::array_view<uint8_t> bytes)
{
    winrt::InMemoryRandomAccessStream stream;
    co_await stream.WriteAsync(BufferFromBytes(bytes));

    stream.Seek(0);
    co_return stream;
}
```

This is rather annoying because `WriteAsync()` is an async method, which means we have to `co_await` it, which in turn forces us to be a coroutine as well.

But there's a shortcut: `CreateRandomAccessStreamOverStream`.

The `CreateRandomAccessStreamOverStream` function takes a classic COM `IStream` and creates a Windows Runtime `IRandomAccessStream` around it.

So this will actually take two steps. First we need to put the bytes into an `IStream`. Then we can wrap the `IStream` inside an `IRandomAccessStream`.

```
// C++/WinRT

winrt::com_ptr<IStream> StreamFromBytes(winrt::array_view<uint8_t> bytes)
{
    winrt::com_ptr<IStream> stream{
        SHCreateMemStream(reinterpret_cast<const BYTE*>(bytes.data()),
                          bytes.size()),
        winrt::take_ownership_from_abi };
    winrt::check_pointer(stream.get());
    return stream;
}

winrt::IRandomAccessStream
    BytesToRandomAccessStream(winrt::array_view<uint8_t> bytes)
{
    return winrt::capture<winrt::IRandomAccessStream>(
        CreateRandomAccessStreamOverStream,
        StreamFromBytes(bytes).get(),
        BSOS_DEFAULT);
}
```

Bonus reading: The difference between assignment and attachment with ATL smart pointers, exacerbated by `SHCreateMemStream` not following standard COM patterns.

For C#, somebody did part of the work for you: `WindowsRuntimeStreamExtensions.AsRandomAccessStream` converts a `System.IO.Stream` to a Windows Runtime `IRandomAccessStream`.

```
// C#
IRandomAccessStream
    BytesToRandomAccessStream(byte* bytes, long length)
{
    var stream = new UnmanagedMemoryStream(bytes, length);
    return stream.AsRandomAccessStream();
}
```

The above function looks simple, but it's also dangerous because you don't know when it's safe to free the `bytes`. We can copy them into a `MemoryStream` so that its lifetime is properly managed.

```
// C#
MemoryStream
    StreamFromBytes(byte* bytes, int length)
{
    var stream = new MemoryStream(length);
    stream.Write(new ReadOnlySpan<byte>(bytes, length));
    stream.Seek(0, SeekOrigin.Begin);
    return stream;
}

IRandomAccessStream
    BytesToRandomAccessStream(byte* bytes, int length)
{
    return StreamFromBytes(bytes, length).AsRandomAccessStream();
}
```