The case of the make_shared on a C++/WinRT type

devblogs.microsoft.com/oldnewthing/20230621-00

June 21, 2023

Raymond Chen

A customer asked for some debugging assistance with their C++/WinRT object. It seemed that the code was crashing in an assignment statement. They were able to capture a Time Travel Trace, so let's join our investigation already in progress.

```
(819c.6e48): Access violation - code c0000005 (first/second chance not available)
First chance exceptions are reported before any exception handling.
This exception may be expected and handled.
Time Travel Position: 1010F3:0
Contoso!winrt::Windows::Foundation::IUnknown::unconditional_release_ref+0x31:
00007ffc`08e03b15 mov     rax,qword ptr [rax+10h]
ds:00000000`0000029d=????????????????
```

We are crashing on a garbage pointer. (The effective address is given as `00000000`0000029d`, which implies that the current value in `rax` is `00000000`0000028d`, which is not a valid pointer.)

Let's see where that pointer came from.

```
0:013> u .-31 .
Contoso!winrt::Windows::Foundation::IUnknown::unconditional_release_ref:
00007ffc`08e03ae4 mov     qword ptr [rsp+8],rcx
00007ffc`08e03ae9 sub     rsp,48h
00007ffc`08e03aed mov     qword ptr [rsp+28h],0
00007ffc`08e03af6 mov     rax,qword ptr [rsp+50h]
00007ffc`08e03afb lea     rdx,[rsp+28h]
00007ffc`08e03b00 mov     rcx,rax
00007ffc`08e03b03 call    Contoso!std::exchange (00007ffc`08d0c8dc)
00007ffc`08e03b08 mov     qword ptr [rsp+20h],rax
00007ffc`08e03b0d mov     rax,qword ptr [rsp+20h]
00007ffc`08e03b12 mov     rax,qword ptr [rax]
00007ffc`08e03b15 mov     rax,qword ptr [rax+10h]
```

We see that `rsp+20h` holds the exchanged-out `IUnknown` pointer, which means that the `mov rax, [rax]` loads the vtable, and the `mov rax, [rax+10h]` is trying to load a function from the vtable.

Let's see how we got here.

```
0:013> g- 00007ffc`08e03ae4
Time Travel Position: 1010C3:DB
Contoso!winrt::Windows::Foundation::IUnknown::unconditional_release_ref:
00007ffc`08e03ae4 mov     qword ptr [rsp+8],rcx ss:000000d5`fa05e500=00007ffc00000000
0:013> t-
Time Travel Position: 1010C3:DA
Contoso!winrt::Windows::Foundation::IUnknown::release_ref+0x19:
00007ffc`08e01819 call Contoso!winrt::...::unconditional_release_ref
(00007ffc`08e03ae4)
0:013> k
RetAddr               Call Site
00007ffc`092314b0     Contoso!winrt::Windows::Foundation::IUnknown::release_ref+0x19
00007ffc`09231459     Contoso!winrt::Windows::Foundation::IUnknown::operator=+0x24
00007ffc`09231569
Contoso!winrt::Windows::Foundation::IInspectable::operator=+0x1d
00007ffc`092315e1     Contoso!winrt::Contoso::IGadget::operator=+0x1d
00007ffc`0924d31b     Contoso!winrt::Contoso::Gadget::operator=+0x1d
00007ffc`0925b1f2
Contoso!WidgetManager::SetPrimaryGadgetAsync$_ResumeCoro$1+0x103b
00007ffc`09250273     Contoso!std::experimental::coroutine_handle<void>::resume+0x4a
00007ffc`09223039     Contoso!winrt::impl::await_adapter<...>::await_suspend+0xdb
00007ffc`0924cfdb     Contoso!winrt::impl::notify_awaiter<...>::await_suspend+0x31
00007ffc`0925b1f2
Contoso!WidgetManager::SetPrimaryGadgetAsync$_ResumeCoro$1+0xcfb
00007ffc`092326b7     Contoso!std::experimental::coroutine_handle<void>::resume+0x4a
00007ffc`0925b2aa
Contoso!std::experimental::coroutine_handle<void>::operator()+0x13
00007ffc`0923515f     Contoso!winrt::impl::resume_apartment+0xb2
00007ffc`092215ad
Contoso!winrt::impl::disconnect_aware_handler<...>::Complete+0x9f
00007ffc`09246c9e
Contoso!winrt::impl::disconnect_aware_handler<...>::operator()+0x2d
00007ffc`8c7493fd     Contoso!winrt::impl::delegate<...>::Invoke+0x3e
```

We are running due to the completion of an asynchronous operation.

```
0:013> u 00007ffc`0924d31b-80 00007ffc`0924d31b
Contoso!WidgetManager::SetPrimaryGadgetAsync$_ResumeCoro$1+0xfbb:
...
00007ffc`0924d2f8 mov     rax,qword ptr [rsp+600h]
00007ffc`0924d300 mov     rax,qword ptr [rax+281h]
00007ffc`0924d307 add     rax,58h
00007ffc`0924d30b mov     rdx,qword ptr [rsp+470h]
00007ffc`0924d313 mov     rcx,rax
00007ffc`0924d316 call    Contoso!winrt::Contoso::Gadget::operator=
(00007ffc`092315c4)
00007ffc`0924d31b mov     eax,118h
```

We're at this assignment statement:

```
IAsyncAction WidgetManager::SetPrimaryGadgetAsync()
{
    auto strongThis = get_strong();
    co_await winrt::resume_background();

    try
    {
        auto result = co_await GadgetFinder::GetGadgetAsync(GadgetKind::Primary);
        if (result.Status() == GadgetFinderStatus::Success)
        {
            m_primaryGadget = result.Gadget(); // ⇐ here
        }
    }
    catch(...)
    {
        LOG("Error finding primary gadget", winrt::to_hresult());
    }
}
```

Let's look at the state of our object just before calling the assignment operator.

```
0:013> g- 00007ffc`0924d316
Time Travel Position: 1010C3:B9
Contoso!WidgetManager::SetPrimaryGadgetAsync$_ResumeCoro$1+0x1036:
00007ffc`0924d316 call    Contoso!winrt::Contoso::Gadget::operator=
(00007ffc`092315c4)
0:013> dv
               result = struct winrt::Contoso::FindGadgetResult
           strongThis = struct winrt::com_ptr<WidgetManager>
      __coro_frame_ptr = 0x00007ffc`0924c2e0
0:013> ?? strongThis
struct winrt::com_ptr<WidgetManager>
   +0x000 m_ptr            : 0x0000028d`1207f0c0 WidgetManager
0:013> ?? strongThis.m_ptr
class WidgetManager * 0x0000028d`1207f0c0
   +0x000 __VFN_table : ????
   +0x018 vtable           : winrt::impl::produce<WidgetManager,...>
   +0x008 __VFN_table : ????
   +0x010 m_references     : std::atomic<unsigned __int64>
   +0x020 m_widgetList     : winrt::IVectorView<winrt::Contoso::Gadget>
   +0x028 m_initCompletedEvent : winrt::handle_type<winrt::handle_traits>
   +0x030 m_mainWidgetName : std::wstring
   +0x050 m_primaryGadget  : winrt::Contoso::Gadget
```

Those question marks look suspicious. What does this object look like?

```
0:013> dps 0x0000028d`1207f0c0
0000028d`1207f0c0  00000000`00000000
0000028d`1207f0c8  00000000`00000000
0000028d`1207f0d0  00700041`005c0001
0000028d`1207f0d8  00000000`00000000
0000028d`1207f0e0  00000000`00000000
0000028d`1207f0e8  00000000`00000000
0000028d`1207f0f0  0000028d`10ddb880
0000028d`1207f0f8  0000028d`10ddd4e8
0000028d`1207f100  0000028d`10ddd4e8
0000028d`1207f108  0000028d`06f81940
0000028d`1207f110  0000028d`06f81c34
0000028d`1207f118  0000028d`06f81c34
0000028d`1207f120  00000000`00000000
0000028d`1207f128  00000000`00000000
0000028d`1207f130  00000000`00000000
0000028d`1207f138  00000000`00000000
```

Yeah, that doesn't look good. The object is corrupted.

Let's see if it was corrupted when the coroutine began.

```
0:000> bp Contoso!WidgetManager::SetPrimaryGadgetAsync
0:000> g-
Breakpoint 0 hit
Time Travel Position: 6EAA3:50
Contoso!WidgetManager::SetPrimaryGadgetAsync:
00007ffc`0924d838 mov     qword ptr [rsp+10h],rdx
ss:000000d5`f95fc838=0000028d0f8ab990
0:002> ?? ((Contoso!WidgetManager*)@rcx)
class WidgetManager * 0x0000028d`1207f0c0
   +0x000 __VFN_table : 0x00007ffc`09c2c048
   +0x018 vtable           : winrt::impl::produce<WidgetManager,...>
   +0x008 __VFN_table : 0x00007ffc`09c2c098
   +0x010 m_references     : std::atomic<unsigned __int64>
   +0x020 m_widgetList     : winrt::IVectorView<winrt::Contoso::Gadget>
   +0x028 m_initCompletedEvent : winrt::handle_type<winrt::handle_traits>
   +0x030 m_mainWidgetName : std::wstring
   +0x050 m_primaryGadget  : winrt::Contoso::Gadget
0:002> ?? ((Contoso!WidgetManager*)@rcx)->m_primaryGadget
struct winrt::Contoso::Gadget
   +0x000 m_ptr            : (null)
```

Looks good. Let's see when it gets corrupted.

```
0:013> ba w4 @rcx
0:002> g
Breakpoint 1 hit
Time Travel Position: F253A:BC
ucrtbase!memcpy+0x2b1:
00007ffc`aa2914a1 vmovdqu ymm1,ymmword ptr [rdx+r9-0A0h] ds:0000028d`119673c0=61
0:013> k
RetAddr               Call Site
00007ffc`8c72ab79     ucrtbase!memcpy+0x2b1
(Inline Function)     Contoso!std::_Char_traits<unsigned short,unsigned
short>::copy+0x11
(Inline Function)     Contoso!std::wstring::assign::__l2::
<lambda_...>::operator()+0x11
(Inline Function)     Contoso!std::wstring::_Reallocate_for+0x6c
(Inline Function)     Contoso!std::wstring::assign+0x72
(Inline Function)     Contoso!std::wstring::assign+0xa8
(Inline Function)     Contoso!std::wstring::{ctor}+0xc1
00007ffc`8c72a9af     Contoso!winrt::Contoso::implementation::

GadgetFinder::BuildGadgetQuery+0x179
...
00007ffc`0924ccf1     Contoso!winrt::Contoso::GadgetFinder::GetGadgetAsync+0x3c
00007ffc`0925b1f2
Contoso!WidgetManager::SetPrimaryGadgetAsync$_ResumeCoro$1+0xa11
00007ffc`092326b7     Contoso!std::experimental::coroutine_handle<void>::resume+0x4a
00007ffc`0925b2aa
Contoso!std::experimental::coroutine_handle<void>::operator()+0x13
00007ffc`0923515f     Contoso!winrt::impl::resume_apartment+0xb2
00007ffc`092215ad
Contoso!winrt::impl::disconnect_aware_handler<...>::Complete+0x9f
00007ffc`09246c9e
Contoso!winrt::impl::disconnect_aware_handler<...>::operator()+0x2d
00007ffc`8c7493fd     Contoso!winrt::impl::delegate<...>::Invoke+0x3e
```

That's strange. It's being overwritten by a `std::wstring`.

Wait a second, is the object already destroyed, and we're operating on an a dead object? Let's execute backward and see if the destructor gets hit.

```
0:008> bd1
0:008> x Contoso!WidgetManager::~WidgetManager
00007ffc`08d1d678 Contoso!WidgetManager::~WidgetManager (void)
0:008> g- 00007ffc`08d1d678
Time Travel Position: F21CF:46
Contoso!WidgetManager::~WidgetManager:
00007ffc`08d1d678 mov     qword ptr [rsp+8],rcx ss:000000d5`f95fe660=0000028d0f9fd0f0
0:002> r
rax=00007ffc08d22ea0 rbx=0000000000000000 rcx=0000028d1207f0c0 ⇐ destructing our
object
rdx=0000000000000000 rsi=0000028d11950b48 rdi=0000028d11950b70
rip=00007ffc08d1d678 rsp=000000d5f95fe658 rbp=0000028d11950b40
 r8=000000d5f95fe8d0  r9=0000000000000000 r10=00000fff811a45d5
r11=0202020202222000 r12=0000028d0f876af0 r13=0000000000000000
r14=0000028d11dfaa20 r15=0000000000000000
iopl=0         nv up ei pl nz na po nc
cs=0033  ss=002b  ds=002b  es=002b  fs=0053  gs=002b             efl=00000206
Contoso!WidgetManager::~WidgetManager:
00007ffc`08d1d678 mov     qword ptr [rsp+8],rcx ss:000000d5`f95fe660=0000028d0f9fd0f0
0:002>
```

How about that, the breakpoint on the destructor was hit before we reached the start of the `WidgetManager::SetPrimaryGadgetAsync` method. Since we are executing backward, this means that the `WidgetManager::SetPrimaryGadgetAsync` method started, and then the `WidgetManager` was destroyed. Who is destroying the `WidgetManager`? I thought we had a strong reference to it. That strong reference should prevent the `WidgetManager` from destructing.

```
0:002> k
Call Site
Contoso!WidgetManager::~WidgetManager
Contoso!WidgetManager::`scalar deleting destructor'+0x17
Contoso!std::_Destroy_in_place<WidgetManager>+0x4f
Contoso!std::_Ref_count_obj2<WidgetManager>::_Destroy+0x1a
Contoso!std::_Ref_count_base::_Decref+0x4d
Contoso!std::_Ptr_base<WidgetManager>::_Decref+0x23
Contoso!std::shared_ptr<WidgetManager>::~shared_ptr+0x13
Contoso!winrt::Contoso::implementation::DoodadPageViewModel::~DoodadPageViewModel+0x71

Contoso!winrt::impl::heap_implements<implementation::DoodadPageViewModel>::~heap_imple

Contoso!winrt::impl::heap_implements<implementation::DoodadPageViewModel>::`scalar
deleting destructor'+0x17
Contoso!winrt::impl::root_implements<implementation::DoodadPageViewModel>::NonDelegati

Contoso!winrt::impl::root_implements<implementation::DoodadPageViewModel>::Release+0x5

Windows_UI_Xaml!DirectUI::TrackerTargetReference::Clear+0x1ee
...
```

You might notice something unusual about the stack trace that leads to the destructor. If you don't, we can go back to the constructor:

```
0:002> x Contoso!WidgetManager::WidgetManager
00007ffc`08d167f8 Contoso!WidgetManager::WidgetManager (void)
0:002> g- 00007ffc`08d167f8
Time Travel Position: 6EA29:114
Contoso!WidgetManager::WidgetManager:
00007ffc`08d167f8 mov     qword ptr [rsp+18h],r8
ss:000000d5`f95fd7a0=00007ffc09c67028
0:002> r
rax=00007ffc09c66fa0 rbx=0000000000000000 rcx=0000028d1207f0c0 ⇐ constructing our
object
rdx=00007ffc09c66fa0 rsi=0000028d0f89b4b0 rdi=0000028d1207f0c0
rip=00007ffc08d167f8 rsp=000000d5f95fd788 rbp=0000000000000000
 r8=000000d5f95fd7d8  r9=00000000ffffffff r10=00007ffcaa240000
r11=0000000000000000 r12=0000000000000000 r13=000000d5f95fe2b0
r14=000000d5f95fe090 r15=0000000000000000
iopl=0         nv up ei pl nz na po nc
cs=0033  ss=002b  ds=002b  es=002b  fs=0053  gs=002b             efl=00000206
Contoso!WidgetManager::WidgetManager:
00007ffc`08d167f8 mov     qword ptr [rsp+18h],r8
ss:000000d5`f95fd7a0=00007ffc09c67028
0:002> k
Call Site
Contoso!WidgetManager::WidgetManager
Contoso!std::_Construct_in_place<WidgetManager;>+0x93
Contoso!std::_Ref_count_obj2<WidgetManager>::_Ref_count_obj2<WidgetManager>+0x7e
Contoso!std::make_shared<WidgetManager>+0x85
Contoso!Contoso::implementation::DoodadPageViewModel::DoodadPageViewModel+0x2b8
```

From the stack, we see that the `WidgetManager` is being constructed via `std::make_shared`.

Too bad C++/WinRT objects aren't supposed to be constructed via `std::make_shared`.

C++/WinRT objects are reference-counted and manage their reference count internally. Putting them inside a `shared_ptr` creates a conflict between two reference counts: There's a reference count in the C++/WinRT object, and another reference count in the `shared_ptr`.

When the `WidgetManager::SetPrimaryGadgetAsync` method performs a `auto strongThis = get_strong();`, it is getting a `com_ptr<WidgetManager>` whose lifetime is tracked in the reference count embedded in the `WidgetManager`.

Too bad the `shared_ptr` doesn't know about that reference count.

When the last `shared_ptr` reference goes away, the strong reference count in the `shared_ptr` control block goes to zero, and the `WidgetManager` destructs. The reference count hiding inside the `WidgetManager` does not participate in this decision.

This explains why the `get_strong()` call in the coroutine is ineffective in extending the lifetime of the `WidgetManager`.

At this point, you have to decide whether you want `WidgetManager` to be a plain C++ object managed by `shared_ptr` or a fancy C++/WinRT object managed by `winrt::implements`.

	`std::shared_ptr`	`winrt::implements`
Base class	`std::enable_shared_from_this`	`std::implements`
Create with	`std::make_shared`	`winrt::make`
Get strong reference	`shared_from_this()`	`get_strong()`

In this particular case, the `WidgetManager` implements some COM interfaces, so it needs to use `winrt::implements` as its base class, and the correct function for creating the object is `winrt::make<WidgetManager>()`.

Bonus chatter: If the `_DEBUG` symbol is defined, then C++/WinRT will trigger a build break[1] if you construct a `winrt::implements`-derived object by some means other than `winrt::make`:

```
xmemory(1893,9): error C2259: 'WidgetManager': cannot instantiate abstract class
sample.cpp(601): message : see declaration of 'WidgetManager'
xmemory(1893,9): message : due to following members:
xmemory(1893,9): message : 'void winrt::impl::root_implements<D,
winrt::Windows::Foundation::IInspectable>::use_make_function_to_create_this_
object(void)': is abstract
        with
        [
            D=WidgetManager
        ]
base.h(7308): message : see declaration of 'winrt::impl::root_implements<D,
winrt::Windows::Foundation::IInspectable>::use_make_function_to_create_this_object'
        with
        [
            D=WidgetManager
        ]
memory(2027): message : see reference to class template instantiation
'std::_Wrap<_Ty>' being compiled
        with
        [
            _Ty=WidgetManager
        ]
memory(2725): message : see reference to class template instantiation
'std::_Ref_count_obj2<_Ty>' being compiled
        with
        [
            _Ty=WidgetManager
        ]
sample.cpp(643): message : see reference to function template instantiation
'std::shared_ptr<WidgetManager> std::make_shared<WidgetManager,>(void)' being
compiled
```

If the customer had turned on the `_DEBUG` symbol in their debug build, they would have found this problem much sooner.

[1] This is another example of compiler error meta-programming: structuring the code to influence the nature of the compiler error, in the hopes that the error will be more self-explanatory.