Peeking inside C++/CX delegates

devblogs.microsoft.com/oldnewthing/20200124-00

January 24, 2020

Raymond Chen

Let's hope you never need to do this, but if you are forced to debug code written in C++/CX, and you have a C++/CX delegate and want to know what it is, well, here goes.

```
0:005> dps 0x00000295`0d3a0ab0
00000295`0d3a0ab0  00007ff9`c0238fd8 contoso!RoutedEventHandler::`vftable'
00000295`0d3a0ab8  00007ff9`c0238f98 contoso!RoutedEventHandler::`vftable'
00000295`0d3a0ac0  00007ff9`c0238f68 contoso!RoutedEventHandler::`vftable'
00000295`0d3a0ac8  00000295`7d9f8ee0
00000295`0d3a0ad0  ffffffff`ffffffff
00000295`0d3a0ad8  00007ff9`c0252f08 contoso!`RoutedEventHandler<Widget,
                                      void (Widget::*)(Object ^,RoutedEventArgs
^)>'::

`2'::__abi_PointerToMemberWeakRefCapture::`vftable'
00000295`0d3a0ae0  00000295`469826e0
00000295`0d3a0ae8  00007ff9`bfff2e80 contoso!Widget::OnColorChanged
```

The object starts with some vtables and other bookkeeping. But the interesting thing is the next vtable, because that one tells you what kind of delegate you have.

In this case, it's a vtable for a "pointer to member weak ref capture", which tells us that our delegate is a weak pointer plus a pointer to member function.

The next two pointers are the weak reference and the member function pointer.

Most C++/CX delegates are of the "weak pointer plus method pointer" variety, but the other flavor is the "functor", where the handler is an arbitrary object that supports the function call operator.

For example, a delegate that refers to a static method looks like this:

```
0:000> dps 08062190
08062190  0116a2f4 contoso!RoutedEventHandler::`vftable'
08062194  0116a310 contoso!RoutedEventHandler::`vftable'
08062198  0116a334 contoso!RoutedEventHandler::`vftable'
0806219c  08068a10
080621a0  ffffffff
080621a4  0116a378 contoso!__abi_FunctorCapture<
                   void (__cdecl*)(Object ^,RoutedEventArgs ^),
                   void,
                   Object ^,
                   RoutedEventArgs ^>::`vftable'
080621a8  00f9371c contoso!OnColorChanged
```

The vtable in position 5 says that this is a functor that captured a plain old function pointer, and the plain old function pointer comes immediately after: It's `contoso!OnColorChanged`.

For a class that supports the function call operator, you get a customized function for that class. In this example, the class is a lambda:

```
08062240  0116a2f4 contoso!RoutedEventHandler::`vftable'
08062244  0116a310 contoso!RoutedEventHandler::`vftable'
08062248  0116a334 contoso!RoutedEventHandler::`vftable'
0806224c  080689b0
08062250  ffffffff
08062254  0116a388 contoso!__abi_FunctorCapture<
                   <lambda_484c5c4112f278281e0294b32780c5b6>,
                   void,
                   Object ^,
                   RoutedEventArgs ^>::`vftable'
08062258  08053480 // lambda contents start here
0806225c  00000000
```

If the lambda is large (bigger than than 16 pointers), then it is stored in a separate memory allocation. You can find a pointer to the wrapper for the captured lambda 16 pointers later:

```
07a2c2f0  00e57324 contoso!RoutedEventHandler::`vftable'
07a2c2f4  00e57340 contoso!RoutedEventHandler::`vftable'
07a2c2f8  00e57364 contoso!RoutedEventHandler::`vftable'
07a2c2fc  07a36278
07a2c300  ffffffff
07a2c304  00000000 (unused slot 0)
07a2c308  00000000 (unused slot 1)
07a2c30c  00000000 (unused slot 2)
07a2c310  00000000 (unused slot 3)
07a2c314  00000000 (unused slot 4)
07a2c318  00000000 (unused slot 5)
07a2c31c  00000000 (unused slot 6)
07a2c320  00000000 (unused slot 7)
07a2c324  00000000 (unused slot 8)
07a2c328  00000000 (unused slot 9)
07a2c32c  00000000 (unused slot 10)
07a2c330  00000000 (unused slot 11)
07a2c334  00000000 (unused slot 12)
07a2c338  00000000 (unused slot 13)
07a2c33c  00000000 (unused slot 14)
07a2c340  00000000 (unused slot 15)
07a2c344  01212a48 // pointer to functor

01212a48  00e57388 contoso!__abi_FunctorCapture<
                   <lambda_51d3f9ef4e98c0ae34c7116adc4d714e>,
                   void,
                   Object ^,
                   RoutedEventArgs ^>::`vftable'
01212a4c  07a1f3d0 // lambda contents start here
01212a50  00000000
```

Raymond Chen

Follow