The vtable does not always go at the start of the object

devblogs.microsoft.com/oldnewthing/20060120-00

January 20, 2006

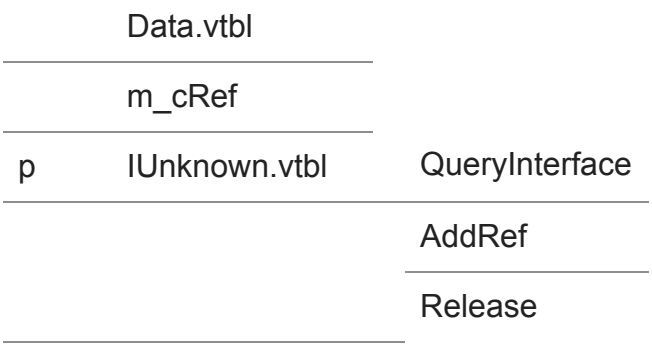
Raymond Chen

Although the diagrams I presented in my discussion of The layout of a COM object place the vtable at the beginning of the underlying C++ object, there is no actual requirement that it be located there. It is perfectly legal for the vtable to be in the middle or even at the end of the object, as long as the functions in the vtable know how to convert the address of the vtable pointer to the address of the underlying object. Indeed, in the second diagram in that article, you can see that the "q" pointer indeed points into the middle of the object.

Here's an example that puts the vtable at the end of the object:

```
class Data {
public:
 Data() : m_cRef(1) { }
 virtual ~Data() { }
 LONG m_cRef;
};
class VtableAtEnd : Data, public IUnknown {
public:
 STDMETHODIMP QueryInterface(REFIID riid, void **ppvOut)
 {
  if (riid == IID_IUnknown) {
   AddRef();
   *ppvOut = static_cast<IUnknown*>(this);
   return S_OK;
  }
  *ppvOut = NULL;
   return E_NOINTERFACE;
 }
 STDMETHODIMP_(ULONG) AddRef()
 {
  return InterlockedIncrement(&m_cRef);
 }
 STDMETHODIMP_(ULONG) Release()
 {
  LONG cRef = InterlockedDecrement(&m_cRef);
  if (!cRef) delete this;
  return cRef;
 }
};
```

The layout of this object may very well be as follows: (Warning: Diagram requires a VML-enabled browser.)



Observe that in this particular object layout, the vtable resides at the end of the object rather than at the beginning. This is perfectly legitimate behavior. Although it is the most common object layout to put the vtable at the beginning, COM imposes no requirement that it be done that way. If you want to put your vtable at the end and use negative offsets to access your object's members, then more power to you.

Raymond Chen

Follow