How do I prevent my ATL class from participating in COM aggregation? DECLARE_NOT_AGGREGATABLE didn't work

devblogs.microsoft.com/oldnewthing/20240101-00

January 1, 2024

Raymond Chen

Consider the following ATL class declaration:

```
class Widget :
    public CComObjectRootEx<CComMultiThreadModel>,
    public CComCoClass<Widget>,
    public IAgileObject
{
public :
    DECLARE_NOT_AGGREGATABLE(Widget)

    BEGIN_COM_MAP(Widget)
        COM_INTERFACE_ENTRY(IAgileObject)
    END_COM_MAP()

    [[ ... ]]
};
```

This class says that it is not aggregatable.

Hold my beer.

```
HRESULT CreateAggregatedWidget(
    IUnknown* punkOuter, REFIID riid, void** ppv)
{
    return
        CComCreator<CComAggObject<Widget>>::
            CreateInstance(punkOuter, riid, pppv);
}
```

There, I aggregated your allegedly non-aggregatable object.

The `DECLARE_NOT_AGGREGATABLE` macro is an instruction to the class factory to disallow aggregation, but if you create the object by means other than the class factory, then the macro has no effect. Creating the object without using the class factory is common for objects created internally.

So how do you mark your class so that any attempt to aggregate the object encounters a compiler error?

I found a sneaky trick.

The `CComAggObject` template sets up aggregation by creating the inner object and overriding its `IUnknown` methods to forward to the controlling unknown, kept in the member variable `m_pOuterUnknown`. The connection is made here:

```
template <class Base>
class CComContainedObject :
    public Base
{
public:
    typedef Base _BaseBlass;

    CComContainedObject(void* pv)
    {
        this->m_pOuterUnknown = (IUnknown*)pv;
    }
```

If we can make this line of code produce an error, then we can prevent people from instantiating `CComContainedObject<Widget>` and thereby prevent them from putting it in a `CComAggObject`.

I had multiple ideas for how to accomplish this, but the simplest was this:

```
class Widget :
    public CComObjectRootEx<CComMultiThreadModel>,
    public CComCoClass<Widget>,
    public IAgileObject
{
public :
    DECLARE_NOT_AGGREGATABLE(Widget)

    static constexpr void* m_pOuterUnknown = nullptr;

    BEGIN_COM_MAP(Widget)
        COM_INTERFACE_ENTRY(IAgileObject)
    END_COM_MAP()

    ⟦ ... ⟧
};
```

This shadows the `m_pOuterUnknown` member from `CComObjectRootEx`, so when `CComContainedObject` tries to do a `this->m_pOuterUnknown`, it gets the `Widget` one. And since the `Widget` one is marked `constexpr`, it cannot be modified.

```
error C3892: 'm_pOuterUnknown': you cannot assign to a variable that is const
```

If you are compiling with a version of C++ that doesn't support `constexpr` (which is not entirely out of the question given that you're using ATL, and ATL was written in 1996, over a decade before `constexpr` was a twinkle in Gabriel and Bjarne's eyes), you can use this alternate shadowing definition:

```
    static void m_pOuterUnknown() {}
```

This makes `m_pOuterUnknown` a function, and you cannot assign to a function.

```
error C2659: '=': function as left operand
```