Answer to exercise: Pointer to member function cast

devblogs.microsoft.com/oldnewthing/20040210-00

February 10, 2004

Raymond Chen

Yesterday's exercise asked you to predict and explain the codegen for the following fragment:

```
class Base1 { int b1; void Base1Method(); };
class Base2 { int b2; void Base2Method(); };
class Derived : public Base1, Base2
  { int d; void DerivedMethod(); };
class Derived2 : public Base3, public Derived { };
void (Derived::*pfnDerived)();
void (Derived2::*pfnDerived2();
pfnDerived2 = pfnDerived;
```

Well, the codegen might go something like this:

```
  mov  ecx, pfnDerived[0]       ; ecx = address
  mov  pfnDerived2[0], ecx
  mov  ecx, pfnDerived2[4]      ; ecx = adjustor
  add  ecx, sizeof(Base3)       ; adjust the adjustor!
  mov  pfnDerived2[4], ecx
```

Let's use one of our fancy pictures:

p	→	Base3::b3
q	→	Base2::b2
		Base1::b1
		Derived::d

Just for fun, I swapped the order of Base1 and Base2. There is no requirement in the standard about the order in which storage is allocated for base classes, so the compiler is completely within its rights to put Base2 first, if it thinks that would be more efficient.

A pointer to member function for class Derived expects the "this" pointer to be at "q". So when we have a "p", we need to add sizeof(Base3) to it to convert it to "q", on top of whatever other adjustment the original function pointer wanted. That's why we add sizeof(Base3) to the existing adjustor to make a new combined adjustor.

Raymond Chen

Follow