Pointers to virtual functions with adjustors

devblogs.microsoft.com/oldnewthing/20050324-00

March 24, 2005

Raymond Chen

As a mental exercise, let’s combine two mind-numbing facts about pointers to member functions, namely that all pointers to virtual functions look the same and that pointers to member functions are very strange animals. The result may make your head explode.

Consider:

```
class Class1 {
 public: virtual int f() { return 1; }
};
class Class2 {
 public: virtual int g() { return 2; }
};
class Class3 : public Class1, public Class2 {
};
int (Class3::*pfn)() = Class3::g;
```

Here, the variable `pfn` consists of a code pointer and an adjustor. The code pointer gives you the virtual call stub:

```
 mov eax, [ecx]             ; first vtable
 jmp dword ptr [eax]        ; first function
```

and the adjustor is `sizeof(Class1)` (which in our case would be 4 on a 32-bit machine). The result, then, of compiling a function call `(p->*pfn)()` might look something like this:

```
 mov ecx, p
 lea eax, pfn
 add ecx, dword ptr [eax+4] ; adjust
 call dword ptr [eax]       ; call
-- transfers to
 mov eax, [ecx]             ; first vtable
 jmp dword ptr [eax]        ; first function
-- transfers to
 mov eax, 2                 ; return 2
 ret
```

Okay, I lied. It’s really not all that complicated after all. But you can probably still impress your friends with this knowledge. (If you have really geeky friends.)

Raymond Chen

Follow