

Delegates

Tomáš Matoušek

tm.matfyz.cz/teaching/clr.htm

Delegates

- type-safe pointer to a method
- declaration defines target method signature
- can “point” to both static and instance methods

```
class A
{
    static void f(string x) { }
    public virtual void g(string x) { }

    static void Foo(MyDelegate d)
    {
        if (d != null) d("bar"); // invokes the target method
    }

    static void Main(string[] args)
    {
        Foo(new MyDelegate(f));
        Foo(new MyDelegate(new A().g));
    }
}
```

Delegates: MSIL Level

- C# delegate declaration turned into class declaration by the compiler

```
.class sealed MyDelegate extends [mscorlib]System.MulticastDelegate
{
    .method public instance
        void .ctor(object 'object', native int 'method') runtime managed
    {
    }
    .method public virtual instance
        void Invoke(string x) runtime managed
    {
    }
    .method public virtual instance class [mscorlib]System.IAsyncResult
        BeginInvoke(string x,
                    class [mscorlib]System.AsyncCallback callback,
                    object 'object') runtime managed
    {
    }
    .method public virtual instance
        void EndInvoke(class [mscorlib]System.IAsyncResult result) runtime managed
    {
    }
}
```

Emitting Delegate Construction

- delegate construction (static method)

```
ldnull
ldftn    void A::f(string)
newobj   instance void MyDelegate::.ctor(object, native int)
```

- delegate construction (instance non-virtual method)

```
// load a reference to target object
dup
ldftn    instance void A::g(string)
newobj   instance void MyDelegate::.ctor(object, native int)
```

- delegate construction (instance virtual method)

```
// load a reference to target object
dup
ldvirtftn instance void A::g(string)
newobj   instance void MyDelegate::.ctor(object, native int)
```

- delegate invocation

```
// load a reference to the delegate
ldstr    "bar"
callvirt instance void MyDelegate::Invoke(string)
```

vm/class.h

vm/comdelegate

vm/*CPU*/istublinker*CPU*.cpp

Delegates: EE Level

- .ctor(object, native int) implemented by COMDelegate::DelegateConstruct
 - the same method for all delegates
- special Invoke emitted for delegate classes
 - target method is instance
 - “this” of the delegate replaced by the target instance
 - target method is static
 - must be shuffled by a stub
 - the stub must get rid of this extra parameter
 - hardcore optimizations
 - see StubLinker*CPU*::EmitMulticastInvoke, StubLinker*CPU*::EmitShuffleThunk, ...

New Delegate Features in v2.0

- **co-/contra- variance of arguments**
 - input arguments
 - type in delegate signature can be more restrictive than in method
⇒ object passed to the Invoke can be passed to the method
 - return type and output arguments
 - type in method signature can be more restrictive than in delegate
⇒ object returned from the method can be returned from the Invoke
- **special handling of the 1st argument**
 - target and the 1st arg are interchangeable

Open vs. Closed Delegates

Instance vs. Static Target Methods

- in instance methods, "this" is a hidden argument #0
 - recall instruction ldarg.0
- let's call the argument #0 a "target"
 - for instance methods, it is the "this"
 - for static methods, it is the 1st argument
- two possibilities
 1. target is bound to a delegate instance
 - each invocation implicitly uses the target
 - delegate is closed over instance/first argument
 2. target is explicitly specified in each invocation
 - delegate represents open method
- creation
 - closed instance and open static
 - created by delegate .ctor or via reflection (Delegate.CreateDelegate)
 - open instance and closed static
 - requires using reflection

Examples

- delegates
- methods

```
delegate string D1(C c, object s);  
delegate string D2(object s);  
delegate string D3();  
delegate object D4(string s);  
  
class C  
{  
    string M1(object s);  
    static string M2(object s);  
}
```

- possible combinations
 - open instance D1 -> M1
 - open static D2 -> M2
 - closed instance D2 -> M1
 - closed over an instance of C
 - closed static D3 -> M2
 - closed over first argument
 - co-/contra- variance
 - D4 -> M1 or M2