Delegate Class

Delegate Class

- Overrides Object's Equals() method;
- It checks to see if Target and Method are the same.

```
MyString ms1 = new MyString(PrintLower)
MyString ms2 = new MyString(PrintLower)
Console.WriteLine(ms1.Equals(ms2));
// displays "true"
```

- Can also use != for "not equal"

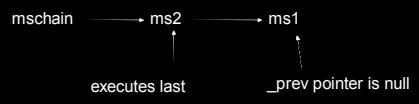
Delegate Chains

- Recall that each Delegate object has a field called _prev that can refer to another delegate object in a linked list.
- The Delegate class defines 3 **static** methods that you can use to manipulate a linked list of delegates:

Delegate Chains

1. public static Delegate Combine(Delegate tail, Delegate head)

```
MyString ms1 = new MyString(PrintLower)
MyString ms2 = new MyString(PrintUpper)
MyString msChain = (MyString) Delegate.Combine(ms1, ms2);
```



Delegate Chains

- We have seen that C# simplifies adding callback methods.

```
MyString ms = null;
ms += new MyString(PrintLower) // calls Combine
Ms -= new MyString(PrintUpper)  // calls Remove
```

Delegate Chains

2. public static Delegate Combine(Delegate[] delegateArray)

```
MyString[] strArr = new MyString[2];
strArr[0] = new MyString(PrintLower);
strArr[1] = new MyString(PrintUpper);
MyString msChain = (MyString) Delegate.Combine(strArr);
```

Delegate Chains

- Invoke() method in pseudocode:

```
public void virtual Invoke(string msg)
{
// if any delegates in chain that should be called first, call
    them recursively
if (_prev != null)
  _prev.Invoke(msg);
// now call the callback method on the specified target
_target.methodPtr(msg);
}
```

Delegate Chains

- You invoke the head delegate with code such as this:

```
if (msChain != null)
  msChain("Hello From Delegate World");
```

Delegate Chains

3. public static Delegate Remove(Delegate source, Delegate value)

- To **remove** a delegate from the delegate chain, use the **static Remove method of the Delegate Class.**

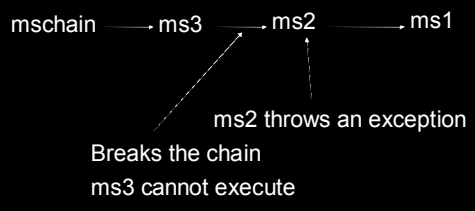
```
msChain = (MyString) Delegate.Remove(msChain, new MyString(PrintLower));
```

- 1st argument refers to the head of the delegate chain
- 2nd argument refers to the delegate object to remove

Delegate Chains

- Scans list looking for something equal to newed up delegate.
- Remember how method Equals() works?
- If one is found, it's removed and new head is returned to caller.
- If no match, no harm, no foul and it returns same head as was passed to the method.

Delegate Chains



Delegate Chains

- In such cases, there is an **instance** method in MulticastDelegate class that you can use to get some more control over this processing and do your own calls to invoke the list of delegates.
- This method is called GetInvocationList()

Delegate Chains

- public virtual Delegate[] GetInvocationList()
- This method is called via a reference to a delegate chain and returns an array of references to delegate objects.
- Returns a **clone** of each delegate object with its _prev pointer nulled out.
- Thus, each is isolated and can be called without invoking the others.

Delegate Chains

```
MyString[] arrayOfDelegates = MyString.GetInvocationList();

foreach(MyString ms in arrayOfDelegates)
{
  ms("some message")
}
```