C# delegates are similar to pointers to functions, in C or C++. A **delegate** is a reference type variable that holds the reference to a method. The reference can be changed at runtime. A delegate holds the address of one function or addresses of many functions. Delegates are especially used for implementing events and the call-back methods. All delegates are implicitly derived from the **System.Delegate** class. Delegates encapsulate (hide) actual information like class name and method names.

Declaring Delegates

Delegate declaration determines the methods that can be referenced by the delegate. A delegate can refer to a method, which have the same signature as that of the delegate.
For example, consider a delegate:

public delegate int MyDelegate (string s);

The preceding delegate can be used to reference any method that has a single *string* parameter and returns an *int* type variable.
Syntax for delegate declaration is:

delegate <return type> <delegate-name> <parameter list>

Instantiating Delegates

Once a delegate type has been declared, a delegate object must be created with the **new** keyword and be associated with a particular method. When creating a delegate, the argument passed to the **new** expression is written like a method call, but without the arguments to the method. For example:

```
public delegate void MyDelegate(string s);
...
MyDelegate md1 = new MyDelegate(WriteToScreen);
MyDelegate  md2 = new MyDelegate(WriteToFile);
```

Note: The return type of function and the return type of the delegate must be one and the same.

Types of Delegations

Delegations are divided into two types

1. Single cast Delegates
   Single cast delegate holds the address of single method.
2. Multicast Delegates
   Multi cast delegate holds the address of multiple methods in single delegate.

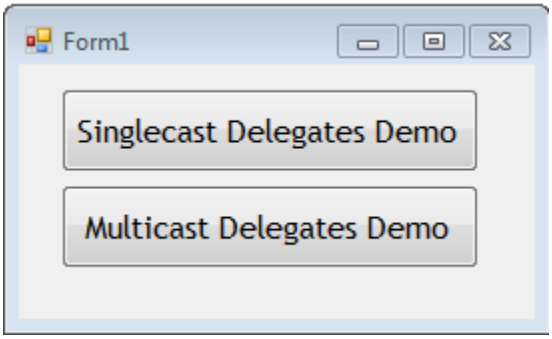
Example

Step 1: Create a class called DelegateDemo.cs

```
class DelegateDemo
    {
        int num = 10;
        public int AddNum(int p)
        {
            num += p;
            return num;
        }

        public int MultNum(int q)
        {
            num *= q;
            return num;
        }
        public int getNum()
        {
            return num;
        }
    }
    delegate int NumberChanger(int n);
```

Step 2: Create the form as following




```
//Multicast Delegate Demo
            DelegateDemo dd = new DelegateDemo();

            //create delegate instances
            NumberChanger nc;
            NumberChanger nc1 = new NumberChanger(dd.AddNum);
            NumberChanger nc2 = new NumberChanger(dd.MultNum);
            nc = nc1;
            nc += nc2;
            //calling multicast
            nc(5);
            MessageBox.Show("Value of Num :" + dd.getNum());
```