

## SAMPLE CODE: DELEGATES AND EVENTS

---

### EXAMPLE 1: USING THE INVOCATION LIST TO CATCH EXCEPTIONS

```
using System;

namespace DelegateTest
{
    /// <summary>
    /// Summary description for Class1.
    /// </summary>
    ///
    delegate void DemoOp();

    class Class1
    {
        /// <summary>
        /// The main entry point for the application.
        /// </summary>
        [STAThread]
        static void Main(string[] args)
        {
            DemoClass dc1 = new DemoClass("Demo 1", 1);        //Create some objects
            DemoClass dc2 = new DemoClass("Demo 2", 2);
            DemoOp demoDel = null;                            //Declare an empty delegate
            demoDel += new DemoOp(dc1.ShowMyNumber);          //Add a method to the
callback list
            demoDel += new DemoOp(dc2.ShowMyName);            //Add another one

            //The following line of code would cause the program to terminate,
            //due to the exception thrown in ShowMyNumber
            //    demoDel();

            //Here is the way to ensure that all methods in the invocation list
            //are called

            Delegate[] delArray = demoDel.GetInvocationList();
            foreach (DemoOp demoOp in delArray)
            {
                try
                {
                    demoOp();
                }
                catch (InvalidOperationException e)
                {
                    Console.WriteLine("Caught {0} from {1}: {2}",
                                        e.GetType(), e.TargetSite, e.Message);
                }
            }
        }
    }
}
```



```

class DemoClass
{
    private string name;
    private int number;

    public DemoClass(string str, int x)
    {
        name = str;
        number = x;
    }

    //When these methods are called via the delegate, "this" will be the
    //object used in the creation of the delegate.
    public void ShowMyName()
    {
        Console.WriteLine("My name is " + this.name);
    }

    public void ShowMyNumber()
    {
        //Console.WriteLine("My number is " + this.number);
        throw new InvalidOperationException("No I won't!");
    }
}
}

```

## EXAMPLE 2: ACCESSING VARIABLES FROM AN ANONYMOUS METHOD

```

using System;
using System.Collections.Generic;
using System.Text;

namespace OuterVariables
{
    class Outer
    {
        static void Main(string[] args)
        {
            Test t = new Test();
            t.F();
            t.d();
        }
    }

    public delegate void D();

    class Test
    {
        int _x = 1;
        public D d = null;
        public void F()
        {
            int y = 123;
            int z = y + _x;
            d = delegate { Console.WriteLine("{0}, {1}, {2}",
                _x, y, z); };
            d();
        }
    }
}

```

}

## EXAMPLE 3: EVENTS

```
using System;

namespace CustomEventConsole
{
    class Class1
    {
        static void Main()
        {
            InfoProvider ip = new InfoProvider();
            ip.InfoEvent += GotInfo;
            ip.Start();
        }

        public static void GotInfo(object sender, InfoEventArgs e)
        {
            Console.WriteLine(e.Date.ToString());
        }
    }

    public class InfoEventArgs : EventArgs
    {
        private DateTime date;

        public InfoEventArgs(DateTime date)
        {
            this.date = date;
        }

        public DateTime Date
        {
            get { return date; }
        }
    }

    public class InfoProvider
    {
        public event EventHandler<InfoEventArgs> InfoEvent;

        //Note: the generic System.EventHandler is defined like this:
        // public delegate void EventHandler<TEventArgs>
        //     (object sender, TEventArgs e) where TEventArgs : EventArgs;

        public void Start()
        {
            DateTime start;
            TimeSpan span;

            for (int i = 0; i < 5; i++)
            {
                start = DateTime.Now;
                do
                {
                    span = DateTime.Now - start;
                }
                while (span.Seconds < 1);
            }
        }
    }
}
```

```
        DateTime date = DateTime.Now;
        InfoEventArgs args = new InfoEventArgs(date);
        HaveInfo(args);
    }

protected void HaveInfo(InfoEventArgs args)
{
    if (InfoEvent != null)
        InfoEvent(this, args);

    //If you are using threads, search the .NET docs for
    //EventHandler<TEventArgs> delegate
    //to see how to avoid a possible thread safety issue.
}
}
```