What are Delegates?

In this lesson, you'll come to know about delegates with the help of the illustrated example.

WE'LL COVER THE FOLLOWING

A **delegate** is a way of telling C# which method to call when an event is triggered.

For example, if you click a `Button` on a form, the program would call a specific method. It is this pointer that is a delegate.

Delegates form the basis of event handling in C#.

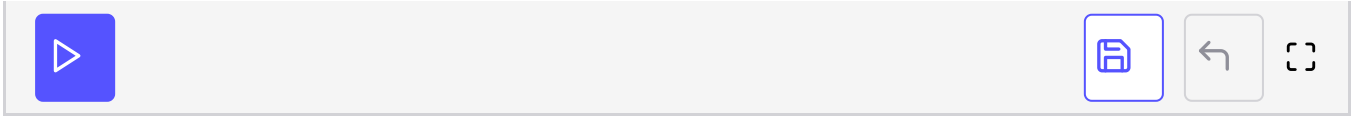
> They are a construct for abstracting and creating objects that reference methods and can be used to call those methods.

A delegate declaration specifies a particular method signature. References to one or more methods can be added to a **delegate instance**. The delegate instance can then be *"called"*, which effectively calls all the methods that have been added to the delegate instance.

A Simple Example

```
using System;
delegate void Procedure();

class DelegateDemo
{
    public static void Method1()
    {
        Console.WriteLine("Method 1");
```

```
    }

    public static void Method2()

    {
        Console.WriteLine("Method 2");
    }

    public void Method3()
    {
        Console.WriteLine("Method 3");
    }

    static void Main()
    {
        Procedure someProcs = null;

        someProcs = new Procedure(DelegateDemo.Method1);
        someProcs();

        someProcs = new Procedure(Method2);  // Example with omitted class name
        someProcs();
    }
}
```



Explanation

In this example, the delegate is declared by the line number **2** `delegate void Procedure()`. This statement is a complete *abstraction*. It does not result in executable code that does any work, but merely declares a delegate type called `Procedure` that takes no arguments and returns nothing.

Next, in the `Main()` method, the statement `Procedure someProcs = null;` in line **23**, instantiates a delegate. The assignment means that the delegate is not initially referencing any methods.

The statements `someProcs = new Procedure(DelegateDemo.Method1)` (line **25**) and `someProcs = new Procedure(Method2)` (line **28**) add two *static methods* to the delegate instance.

> Note that the class name can also be left off, as the statement is occurring inside `DelegateDemo`.

Finally, the statement `someProcs()` (line **26** and **29**) calls the delegate instance.

Cool, right? Let us now shed some light on “anonymous delegates”!