**C# Delegates**

A [delegate](https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/delegate) is a type that represents references to methods with a particular parameter list and return type. When you instantiate a delegate, you can associate its instance with any method with a compatible signature and return type. You can invoke (or call) the method through the delegate instance.

C# delegates are same as pointers to functions, in C or C++. A delegate Object is a reference type variable that use to holds the reference to a method. The reference can be changed at runtime which is hold by an object of delegate, a delegate object can hold many functions reference which is also known as Invocation List that refers functions in a sequence FIFO, we can new functions ref in this list at run time by += operator and can remove by -= operator.   
  
Delegates are especially used for implementing events and the call-back methods. All delegates are implicitly derived from the System.Delegate class.

## Delegates Overview

Delegates have the following properties:

* Delegates are like C++ function pointers but are type safe.
* Delegates allow methods to be passed as parameters.
* Delegates can be used to define callback methods.
* Delegates can be chained together; for example, multiple methods can be called on a single event.
* Methods do not have to match the delegate type exactly. For more information, see [Using Variance in Delegates](https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/concepts/covariance-contravariance/using-variance-in-delegates).
* C# version 2.0 introduced the concept of [Anonymous Methods](https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/statements-expressions-operators/anonymous-methods), which allow code blocks to be passed as parameters in place of a separately defined method. C# 3.0 introduced lambda expressions as a more concise way of writing inline code blocks. Both anonymous methods and lambda expressions (in certain contexts) are compiled to delegate types. Together, these features are now known as anonymous functions. For more information about lambda expressions, see [Anonymous Functions](https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/statements-expressions-operators/anonymous-functions).