**Delegates**

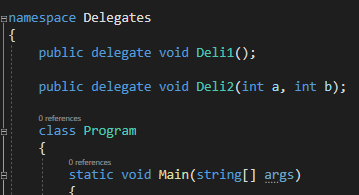
A delegate is an object which refers to a method or you can say it is a reference type variable that can hold a reference to the methods.

In simple words, it is a type that represents references to method**s** with a particular parameter list and return type and then calls the method in a program for execution when it is needed.

Steps of using delegates:

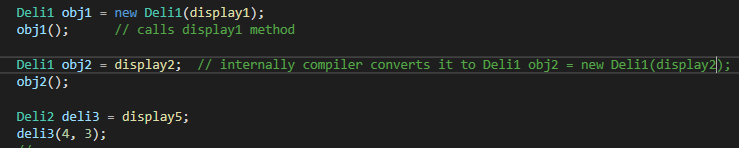
1. Declaration of Delegates:

Delegate type can be declared using the **delegate** keyword.



1. Instantiation & Invocation of Delegates:

Instantiation



Assigning & calling parameterized method

Invocation

**Multi-Cast Delegate:**

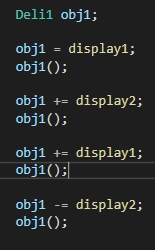
Multicasting of delegate is an extension of the normal delegate (sometimes termed as Single Cast Delegate).

It helps the user to point **more than one** method in a single call.  
**Properties:**

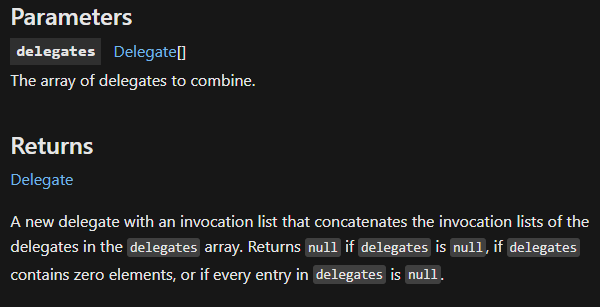
* Delegates are combined and when you call a delegate then a complete list of methods is called.
* All methods are called in First in First Out(FIFO) order.
* ‘+’ or ‘+=’ Operator is used to add the methods to delegates.
* ‘–’ or ‘-=’ Operator is used to remove the methods from the delegates list.

Remember, multicasting of delegate **should have a return type of Void** otherwise it will throw a runtime exception.

Also, the multicasting of delegate will **return the value only from the last method** added in the multicast. Although, the other methods will be executed successfully.

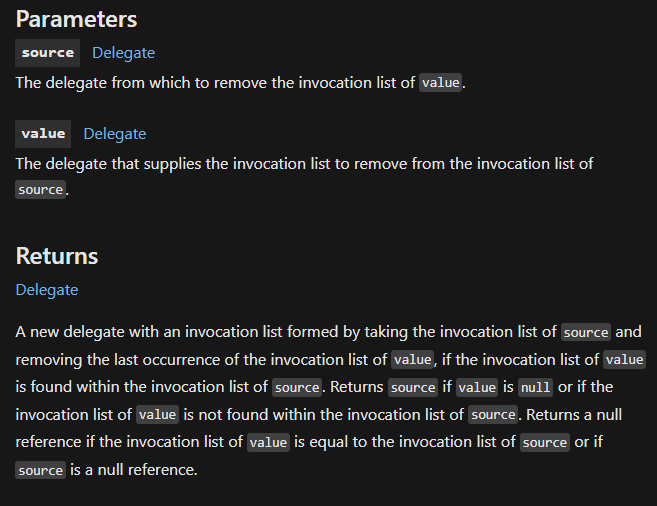


**Combine:**





**Remove:**



i.e. removes that last occurrence of the delegate which is to be removed.



**RemoveAll:**

Removes all the occurrences of delegate which is to be removed.



So, to create & use a custom delegates there are steps which we need to follow.

Following these steps is time consuming, so c# gives us some built in delegates which are as follows:

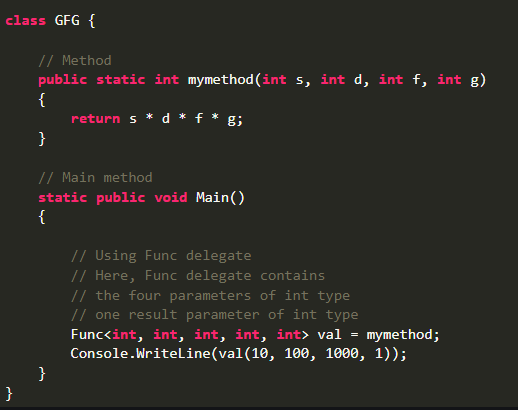
**Func** :

A Func is a built-in generic type delegate.

It can **contain minimum 0 and maximum of 16 input parameters** in it and **contain only one out parameter**.

The *last parameter of the Func delegate is the out parameter which is considered as return type and used for the result*.

Func delegate is used for value returning methods. It can also contain parameters of the same type or of different types.



**Important Points:**

* The last parameter in Func Delegate is always an out parameter which is considered as a return type. It is generally used for the result.
* You can also use a Func delegate with an anonymous method. As shown in the below example:

**Example:**

|  |
| --- |
|  |

* You can also use a Func delegate with the lambda expressions. As shown in the below example:

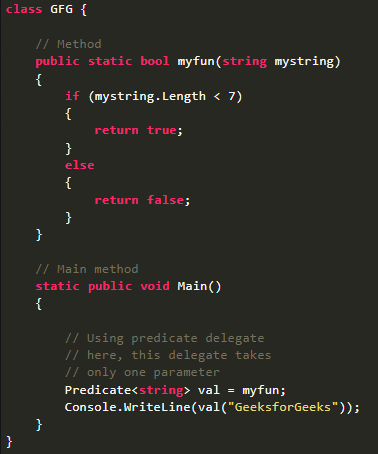
**Example:**



**Predicate:**

A Predicate delegate is an in-built generic type delegate.

This delegate takes only **one input** and returns the value in the form of **true** or **false**.



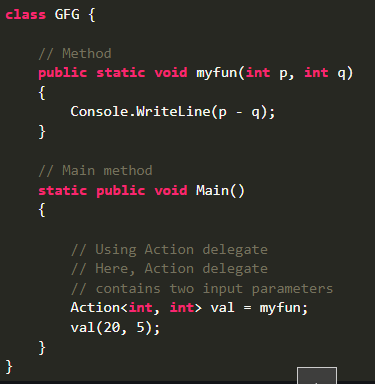
It also can be use with Anonymous Method & lambda function like Func.

**Action:**

Action delegate is an in-built generic type delegate.

It can contain **minimum 1** and **maximum of 16** input **parameters** and **does not contain any output parameter**.

Action delegate is used with those methods whose return type is void. It can also contain parameters of the same type or of different types.



It also can be use with Anonymous Method & lambda function like Func.