A delegate can call more than one method when invoked. This is referred to as multicasting. To add an extra method to the delegate's list of methods—the invocation list—simply requires adding two delegates using the addition or addition assignment operators ('+' or '+='). For example:

C#

MethodClass obj = new MethodClass();

Del d1 = obj.Method1;

Del d2 = obj.Method2;

Del d3 = DelegateMethod;

//Both types of assignment are valid.

Del allMethodsDelegate = d1 + d2;

allMethodsDelegate += d3;

At this point allMethodsDelegate contains three methods in its invocation list—Method1, Method2, and DelegateMethod. The original three delegates, d1, d2, and d3, remain unchanged. When allMethodsDelegate is invoked, all three methods are called in order. If the delegate uses reference parameters, the reference is passed sequentially to each of the three methods in turn, and any changes by one method are visible to the next method. When any of the methods throws an exception that is not caught within the method, that exception is passed to the caller of the delegate and no subsequent methods in the invocation list are called. If the delegate has a return value and/or out parameters, it returns the return value and parameters of the last method invoked. To remove a method from the invocation list, use the decrement or decrement assignment operator ('-' or '-='). For example:

C#

//remove Method1

allMethodsDelegate -= d1;

// copy AllMethodsDelegate while removing d2

Del oneMethodDelegate = allMethodsDelegate - d2;

Because delegate types are derived from System.Delegate, the methods and properties defined by that class can be called on the delegate. For example, to find the number of methods in a delegate's invocation list, you may write:

C#

int invocationCount = d1.GetInvocationList().GetLength(0);

Delegates with more than one method in their invocation list derive from MulticastDelegate, which is a subclass of System.Delegate. The above code works in either case because both classes support GetInvocationList.

System.Delegate

------------🡪MulticastDelegate