# Delegates

And what are **Delegates** in C#? A 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. Delegates are used to pass methods as arguments to other methods. Event handlers are nothing more than methods that are invoked through delegates. You create a custom method, and a class such as a windows control can call your method when a certain event occurs. Any method from any accessible class or struct that matches the delegate type can be assigned to the delegate. The method can be either static or an instance method. This flexibility means you can programmatically change method calls, or plug new code into existing classes.

Στην C#, οι **delegates** είναι τύποι αναφοράς που επιτρέπουν την αποθήκευση αναφορών σε μεθόδους. Χρησιμοποιούνται κυρίως για να εκτελούν μεθόδους asyncronously ή να περνούν μεθόδους ως παραμέτρους. Οι delegates μπορούν να θεωρηθούν σαν pointers για μεθόδους.

Βασικά χαρακτηριστικά των delegates:

* Επιτρέπουν τη χρήση callbacks.
* Μπορούν να συνδυαστούν (multicast delegates).
* Υποστηρίζουν τις ανώνυμες μεθόδους (anonymous methods) και τις εκφράσεις lambda.

# Παραδείγματα χρήσης:

## Βασικός Delegate:

public delegate void MyDelegate(string message);

public class Program

{

public static void DisplayMessage(string message)

{

Console.WriteLine(message);

}

static void Main()

{

MyDelegate del = DisplayMessage;

del("Γειά σου, Κόσμε!");

}

}

## Multicast Delegate:

public delegate void MyMultiDelegate(string message);

public class Program

{

public static void ShowMessage1(string message)

{

Console.WriteLine("Message 1: " + message);

}

public static void ShowMessage2(string message)

{

Console.WriteLine("Message 2: " + message);

}

static void Main()

{

MyMultiDelegate del = ShowMessage1;

del += ShowMessage2; // Προσθήκη δεύτερης μεθόδου

del("Γειά σου, Κόσμε!"); // Καλεί και τις δύο μεθόδους

}

}

## Anonymous Method:

public delegate void MyDelegate(int number);

public class Program

{

static void Main()

{

MyDelegate del = delegate(int n) {

Console.WriteLine("Ο αριθμός είναι: " + n);

};

del(5); // Καλεί την ανώνυμη μέθοδο

}

}

## Lambda Expressions

public delegate int MathOperation(int x, int y);

public class Program

{

static void Main()

{

MathOperation add = (x, y) => x + y;

MathOperation multiply = (x, y) => x \* y;

Console.WriteLine("Άθροισμα: " + add(5, 4));

Console.WriteLine("Πολλαπλασιασμός: " + multiply(5, 4));

}

}