**Part 98 - Anonymous methods in c#**

In this video we will discuss, what anonymous methods are with an example.  
  
**What is an anonymous method?**  
In simple terms, anonymous method is a method without a name.

**Let's understand how a method can exist without a name**

usingSystem**;**

usingSystem.Collections.Generic**;**

classProgram

**{**

publicstaticvoidMain**()**

**{**

List<Employee>listEmployees=newList<Employee>**()**

**{**

newEmployee**{**ID=101**,**Name="Mark"**},**

newEmployee**{**ID=102**,**Name="John"**},**

newEmployee**{**ID=103**,**Name="Mary"**},**

**};**

// Step 2: Create an instance of Predicate<Employee>

// delegate and pass the method name as an argument

// to the delegate constructor

Predicate<Employee>predicateEmployee=

newPredicate<Employee>**(**FindEmployee**);**

// Step 3: Now pass the delegate instance as

// the argument for Find() method

Employeeemployee=

listEmployees.Find**(**x=>predicateEmployee**(**x**));**

Console.WriteLine**(**"ID = {0}, Name {1}"**,**

employee.ID**,**employee.Name**);**

// Anonymous method is being passed as an argument to

// the Find() method. This anonymous method replaces

// the need for Step 1, 2 and 3

employee=listEmployees.Find**(**delegate**(**Employeex**)**

**{**returnx.ID==102**; });**

Console.WriteLine**(**"ID = {0}, Name {1}"**,**

employee.ID**,**employee.Name**);**

**}**

// Step 1: Create a method whose signature matches

// with the signature of Predicate<Employee> delegate

privatestaticboolFindEmployee**(**EmployeeEmp**)**

**{**

returnEmp.ID==102**;**

**}**

publicclassEmployee

**{**

publicintID**{**get**;**set**; }**

publicstringName**{**get**;**set**; }**

**}**

**}**

In this example, **Find**() method expects a delegate to be passed as the argument. If you want to look at the signature of the delegate, right click on **Find()**method and select **"Go To Definition"**from the context menu. At this point you should see the following method.  
publicTFind**(**Predicate<T>match**);**  
  
Right click on Predicate<T> and select  **"Go To Definition"**  
  
Now you should see the signature of the Predicate delegate.  
publicdelegateboolPredicate<in**T**>**(T**obj**);**  
  
Notice that the delegate returns bool and expects an object of Type <T>. In our case T is Employee.   
  
So, to the **Find**() method we need to pass an instance of Predicate<Employee> delegate as an argument. Delegates are function pointers. This means when we create an instance of a delegate, we pass the name of the method as an argument to the delegate constructor.

**Step 1:**Create a method whose signature matches with the signature of Predicate<Employee> delegate

privatestaticboolFindEmployee**(**EmployeeEmp**)**

**{**

returnEmp.ID==102**;**

**}**

**Step 2:** Create an instance of Predicate<Employee> delegate and pass the method name as an argument to the delegate constructor

Predicate<Employee>predicateEmployee=

newPredicate<Employee>**(**FindEmployee**);**

**Step 3:** Now pass the delegate instance as the argument for Find() method

Employeeemployee=

listEmployees.Find**(**x=>predicateEmployee**(**x**));**

**Anonymous methods were introduced in C# 2** and they eliminate the need for Step 1, 2 & 3, that is they provide us a way of creating delegate instances without having to write a separate method.  
  
**Now let us see, how to pass anonymous method as an argument to Find() method.**  
employee=listEmployees.Find**(**delegate**(**Employeex**) {**returnx.ID==102**; });**  
  
**Subscribing for an event handler is another example**

privatevoidForm1\_Load**(**objectsender**,**EventArgse**)**

**{**

ButtonButton1=newButton**();**

Button1.Text="Click Me"**;**

Button1.Click+=newEventHandler**(**Button1\_Click**);**

this.Controls.Add**(**Button1**);**

**}**

voidButton1\_Click**(**objectsender**,**EventArgse**)**

**{**

MessageBox.Show**(**"Button Clicked"**);**

**}**

**The code above can be replaced with the following code**

privatevoidForm1\_Load**(**objectsender**,**EventArgse**)**

**{**

ButtonButton1=newButton**();**

Button1.Text="Click Me"**;**

Button1.Click+=delegate**(**objectobj**,**EventArgseventArgs**)**

**{**

MessageBox.Show**(**"Button Clicked"**);**

**};**

this.Controls.Add**(**Button1**);**

**}**

**With anonymous Methods delegate parameters are optional.** This means the below code

Button1.Click+=delegate**(**objectobj**,**EventArgseventArgs**)**

**{**

MessageBox.Show**(**"Button Clicked"**);**

**};**

**can be rewritten as shown below**

Button1.Click+=delegate

**{**

MessageBox.Show**(**"Button Clicked"**);**

**};**