

note

Everything in an ASP.NET application is a class

You might remember that C#'s code-behind files end in *.cs*, the same file extension as *MyClass.cs*. That's because they are classes too!

An ASP.NET application consists of many classes co-operating with each other to produce the web site you see in your browser.

note

Protection levels

In this lesson you created both your property and your method using the prefix *public*.

public means that the property or method is available within any instance created from the class. For example the *Text* property of a *Label* control is a public property of the *Label* class.

If you'd created a property with the prefix *private*, it would only be available to code inside the class. In other words it would be hidden in any instance created from the class.

public and *private* are called *protection levels*.

You'll learn more about protection levels in: *Lesson 6-8: Create a private method*.

If you are not completing the course incrementally use the sample file: **Lesson 6-1** to begin this lesson.

Sample files with the starting point for each lesson are also provided for all of the other lessons in this session.

Lesson 6-1: Create a class

C# code is organized into classes, which can have properties, methods and events.

Classes can be thought of as templates from which any number of instances of the class (also called objects) may be created.

In previous lessons you created many variables. You were actually creating many instances of the relevant variable's class.

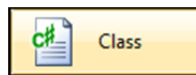
For example, the *DateTime* class defines all of the properties, methods and events of a *DateTime* variable.

In this lesson you'll create a brand new class of your own.

1 Open *My Project* from your sample files folder.

2 Create a new class.

1. Right-click *My Project* in the *Solution Explorer* and then click *Add*→*New Item...* from the shortcut menu.
2. Click *Web* under the *Installed Templates* list on the left-hand side.
3. Click *Class* from the central list of item types.



4. Name the class: **MyClass.cs**



5. Click *Add*.



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace My_Project
{
    public class MyClass
    {
    }
}
```

Your new class will appear in the *Solution Explorer* and its code will be automatically displayed.

3 Add a property to your class.

Add the following line of code to your class:

public int IntProperty;

```
public class MyClass
{
    public int IntProperty;
}
```

This will create a new property called *IntProperty* in your class, using the *int* data type.

You will be able to access this new property using:
MyClass.IntProperty

4 Add a method to your class.

1. Add the following code to your class:

```
public void AddToProperty()
{
}
```

```
public class MyClass
{
    public int IntProperty;
    public void AddToProperty()
    {
    }
}
```

This will create a new method called *AddToProperty*.

You will be able to call the new method using:
MyClass.AddToProperty()

2. Add the following code inside the *AddToProperty* method:

```
IntProperty = IntProperty + 1;
```

```
public void AddToProperty()
{
    IntProperty = IntProperty + 1;
}
```

This method will add 1 to the value of the *IntProperty* property when it is called.

You can see how you can use classes to create your own objects when none of the built in classes provide the functionality you need.

5 Close Visual Studio.