C# Class Members

❮ Previous Next ❯

Class Members

Fields and methods inside classes are often referred to as "Class Members":

Example

Create a `Car` class with three class members: **two fields** and **one method**.

```
// The class
class MyClass
{
  // Class members
  public void fullThrottle()   // method
  {
    Console.WriteLine("The car is going as fast as it can!");
  }
}
```

Fields

In the previous chapter, you learned that variables inside a class are called fields, and that you can access them by creating an object of the class, and by using the dot syntax (`.`).

The following example will create an object of the `Car` class, with the name `myObj`. Then we print the value of the fields `color` and `maxSpeed`:

Example

```
class Car
{
  string color = "red";
  int maxSpeed = 200;

  static void Main(string[] args)
  {
    Car myObj = new Car();
    Console.WriteLine(myObj.color);
}
```

Run example »

You can also leave the fields blank, and modify them when creating the object:

Example

```
class Car
{
  string color;
  int maxSpeed;

  static void Main(string[] args)
  {
    Car myObj = new Car();
    myObj.color = "red";
    myObj.maxSpeed = 200;
    Console.WriteLine(myObj.color);
    Console.WriteLine(myObj.maxSpeed);
  }
}
```

Run example »

Example

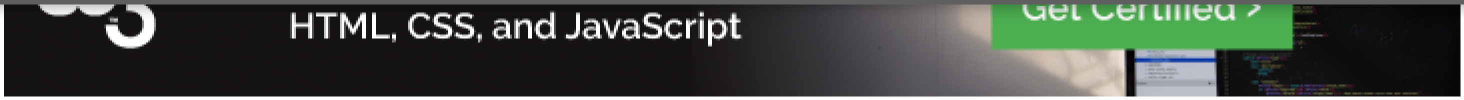
```
class Car
{
  string model;
  string color;
  int year;

  static void Main(string[] args)
  {
    Car Ford = new Car();
    Ford.model = "Mustang";
    Ford.color = "red";
    Ford.year = 1969;

    Car Opel = new Car();
    Opel.model = "Astra";
    Opel.color = "white";
    Opel.year = 2005;

    Console.WriteLine(Ford.model);
    Console.WriteLine(Opel.model);
  }
}
```

Run example »



Object Methods

You learned from the C# Methods chapter that methods are used to perform certain actions.

Methods normally belongs to a class, and they define how an object of a class behaves.

Just like with fields, you can access methods with the dot syntax. However, note that the method must be `public`. And remember that we use the name of the method followed by two parantheses `()` and a semicolon `;` to call (execute) the method:

Example

```
class Car
{
  string color;                 // field
  int maxSpeed;                 // field
  public void fullThrottle()    // method
  {
    Console.WriteLine("The car is going as fast as it can!");
  }

  static void Main(string[] args)
  {
    Car myObj = new Car();
}
```

Run example »

Why did we declare the method as `public`, and not `static`, like in the examples from the C# Methods Chapter?

The reason is simple: a `static` method can be accessed without creating an object of the class, while `public` methods can only be accessed by objects.

Use Multiple Classes

Remember from the last chapter, that we can use multiple classes for better organization (one for fields and methods, and another one for execution). This is recommended:

Car.cs

```
class Car
{
  public string model;
  public string color;
  public int year;
  public void fullThrottle()
  {
}
```

Program.cs

```
class Program
{
  static void Main(string[] args)
  {
    Car Ford = new Car();
    Ford.model = "Mustang";
    Ford.color = "red";
    Ford.year = 1969;

    Car Opel = new Car();
    Opel.model = "Astra";
    Opel.color = "white";
    Opel.year = 2005;

    Console.WriteLine(Ford.model);
    Console.WriteLine(Opel.model);
  }
}
```

Run example »

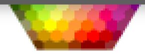
You will learn more about Access Modifiers in a later chapter.

Tip: As you continue to read, you will also learn more about other class members, such as constructors and properties.

❮ Previous Next ❯

HTML CSS JAVASCRIPT MORE ▾

COLOR PICKER



HOW TO

Tabs
Dropdowns
Accordions
Side Navigation
Top Navigation
Modal Boxes
Progress Bars
Parallax
Login Form
HTML Includes
Google Maps
Range Sliders
Tooltips
Slideshow
Filter List
Sort List

SHARE

☰ HTML CSS JAVASCRIPT MORE ▾

HTML
CSS
JavaScript
SQL
Python
PHP
jQuery
Bootstrap
XML

Read More »

HTML CSS JAVASCRIPT MORE ▾

REPORT ERROR PRINT PAGE FORUM ABOUT

Top Tutorials

Top References

JavaScript Tutorial
How To Tutorial
SQL Tutorial
Python Tutorial
W3.CSS Tutorial
Bootstrap Tutorial
PHP Tutorial
jQuery Tutorial
Java Tutorial
C++ Tutorial

Top Examples

HTML Examples
CSS Examples
JavaScript Examples
How To Examples
SQL Examples
Python Examples
W3.CSS Examples
Bootstrap Examples
PHP Examples
jQuery Examples
Java Examples
XML Examples

JavaScript Reference
SQL Reference
Python Reference
W3.CSS Reference
Bootstrap Reference
PHP Reference
HTML Colors
jQuery Reference
Java Reference
Angular Reference

Web Certificates

HTML Certificate
CSS Certificate
JavaScript Certificate
SQL Certificate
Python Certificate
jQuery Certificate
PHP Certificate
Bootstrap Certificate
XML Certificate

Get Certified »

W3Schools is optimized for learning, testing, and training. Examples might be simplified to improve reading and basic understanding. Tutorials, references, and examples are constantly reviewed to avoid errors, but we cannot warrant full correctness of all content. While using this site, you agree to have read and accepted our terms of use, cookie and privacy policy.
Powered by W3.CSS.

w3schools.com