Object-oriented Programming with C#

Tahaluf Training Center 2021

شركـــة تحالـــف الإمـــارات للحـلـــول الـتـقـنـيـــة ذ.م.م.
TAHALUF AL EMARAT TECHNICAL SOLUTIONS L.L.C.

شركـــة تحالـف الإمـــارات للحـــلـــول التـقـنـيـــة ذ.م.م.
TAHALUF AL EMARAT TECHNICAL SOLUTIONS L.L.C.

Day 1

1 **Clean Code**

2 Naming Convention

3 Classes And Objects

4 Access Modifiers

5 Properties, Methods and Constructors

6 Exercises

Clean Code

Clean code is readable and easy to understand by everyone.

Clean code is easy to change and maintainable.

> “It is not enough for code to work.”
>
> -Robert C. Martin

Clean Code

✓ Clean Code

🛇 Dirty Code

Day 1

1 Clean Code

2 **Naming Convention**

3 Classes And Objects

4 Access Modifiers

5 Properties, Methods and Constructors

6 Exercises

Naming Convention

“You should name a variable using the same care with which you name a first-born child.”

—Robert C. Martin, ***Clean Code: A Handbook of Agile Software Craftsmanship***

Naming Convention

There are 2 main types of naming standards that we are using these days:

- ✓ ***Pascal casing***
  A word with the first letter capitalized, and the first letter of each subsequent word-part capitalized.
  Ex - **C**ustomer**N**ame, **E**mployee**D**etails, **S**alary,etc.

- ✓ ***Camel casing***
  A word with the first letter lowercase, and the first letter of each subsequent word-part capitalized.
  EX-**c**ustomer**N**ame, **e**mployee**D**etails, **s**alary

Naming Convention

✓ Always use the **class name** and **method name** in the **Pascal casing.**

✓ Always use **variables** and **parameter** names in **Camel casing**.

✓ Always use the **letter "I" as a prefix** with the name of **interface**. After the letter I, use **Pascal case**.

✓ Use **meaningful** variables and method names while coding. These names should be self-descriptive.

Naming Convention

- ✓ Good Class Name -> Noun, not Verb.
- ✓ Be Specific as Possible.
- ✓ Single Responsibility for the class (SRP).

Naming Convention

شركة تحالف الإمارات للحلول التقنية ذ.م.م.
TAHALUF AL EMARAT TECHNICAL SOLUTIONS L.L.C.

- ✓ Whenever you are writing any method, always try to write the purpose of the method. Use summary or normal comments to specify the purpose of methods with a short description of parameters.

- ✓ Don't use the long method in the project. If by any chance you are using a long method, please use the region to make it easy to understand.

- ✓ Remove unnecessary namespace from your class.

Task

✓ Methods Names

🚫 Get()	✅ GetRegisteredUsers()
🚫 Send()	✅ SendEmail()

✓ Variables Boolean Names

🚫 open	✅ isOpen
🚫 status	✅ isActive
🚫 login	✅ loggedIn

✓ Opposite Words

🚫 on/disable	✅ on/off
🚫 slow/max	✅ min/max
🚫 lock/open	✅ lock/unlock

Day 1

1 Clean Code

2 Naming Convention

3 **Classes And Objects**

4 Access Modifiers

5 Properties, Methods and Constructors

6 Exercises

Classes and Objects

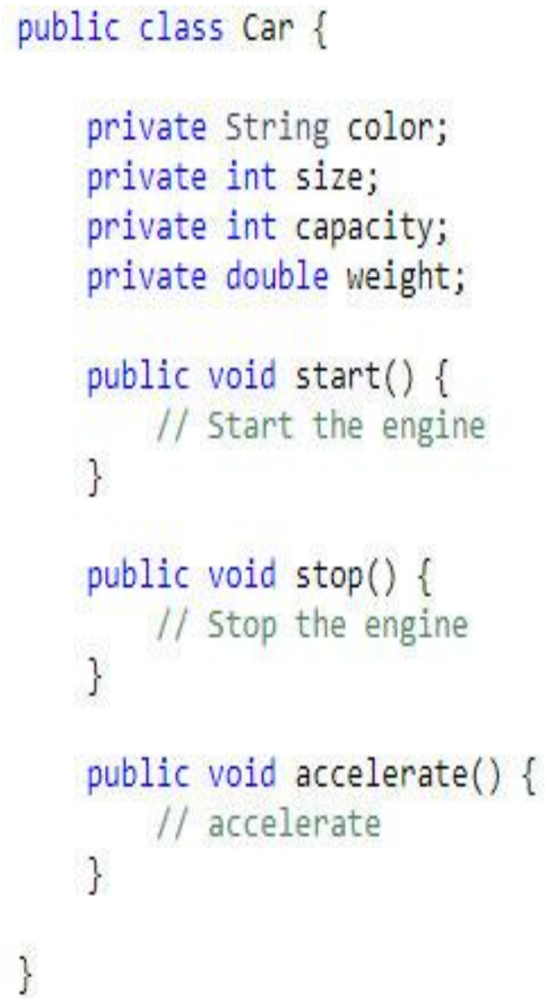
- ✓ **Classes and Objects** are the two main aspects of object-oriented programming.
- ✓ In OOP languages it is **mandatory** to create a class for representing data.
- ✓ **A class** is a **blueprint** of **an object** that contains variables for storing data and functions to perform operations on the data.

Classes and Objects

- ✓ Everything in C# is associated with **classes** and **objects**, along with its attributes and methods.

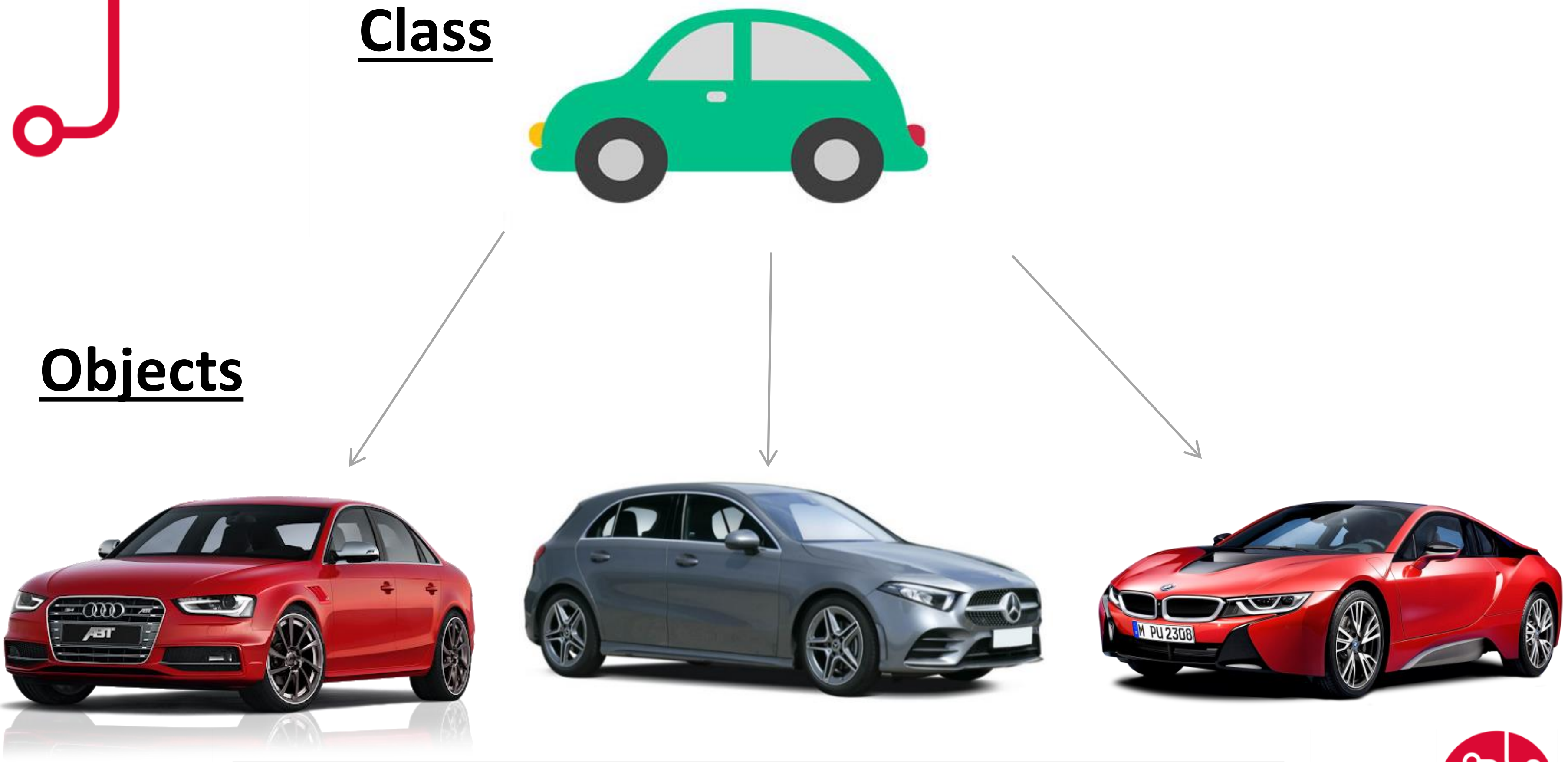
- ✓ For example: in real life, a car is an object. The car has **attributes**, such as weight, size and color, and **methods**, such as drive and brake.

Classes and Objects



```
public class Car {

    private String color;
    private int size;
    private int capacity;
    private double weight;

    public void start() {
        // Start the engine
    }

    public void stop() {
        // Stop the engine
    }

    public void accelerate() {
        // accelerate
    }

}
```

Classes and Objects



Class

Objects

Task

- ✓ Create a class employee detail and create a method to get the employee salary.
- ✓ Create field total salary and assign it to zero as an initial value.
- ✓ The method gets the employee salary takes the parameter id of the employee.

Day 1

1 Clean Code

2 Naming Convention

3 Classes And Objects

4 **Access Modifiers**

5 Properties, Methods and Constructors

6 Exercises

Access Modifiers

✓ The public keyword is an **access modifier**, which is used to set the access level/visibility for classes, fields, methods and properties.

```
public string color;

private string color;

protected string color;

internal string color;
```

Access Modifiers

public	The code is accessible for all classes
private	The code is only accessible within the same class
protected	The code is accessible within the same class, or in a class that is inherited from that class.
Internal	The code is only accessible within its own assembly, but not from another assembly.
Protected Internal	The type or member can be accessed by any code in the assembly in which it's declared, or from within a derived class in another assembly.
Private Protected	The type or member can be accessed only within its declaring assembly, by code in the same class or in a type that is derived from that class.

Why Access Modifiers?

Access Modifiers

- ✓ To control the visibility of class members (the security level of each individual class and class member).

- ✓ To achieve "**Encapsulation**" - which is the process of making sure that "sensitive" data is hidden from users. This is done by declaring fields as private.

- ✓ By default, all members of a class are private if you don't specify an access modifier.

Day 1

1 Clean Code

2 Naming Convention

3 Classes And Objects

4 Access Modifiers

5 **Properties, Methods and Constructors**

6 Exercises

Properties

- ✓ In c#, **Property** is an extension of the class variable and it provides a mechanism to read, write or change the value of the class variable without affecting the external way of accessing it in our applications.

- ✓ In c#, properties can contain one or two code blocks called **accessors** and those are called a get accessor and set accessor. By using get and set accessors, we can change the internal implementation of class variables and expose it without effecting the external way of accessing it based on our requirements.

Properties

```
<access_modifier> <return_type> <property_name>
{
    get
    {
        // return property value
    }
    set
    {
        // set a new value
    }
}
```

Properties

- ✓ Generally, in object-oriented programming languages like **c#** you need to define fields as **private**, and then use properties to access their values in a **public** way with get and set accessors.

```
class User
 {
     //Fields
     private string location;
     private string name;

    //Properties
    public string Location { get => location; set => location = value; }
    public string Name { get => name; set => name = value; }
 }
```

Properties

```
static void Main(string[] args)
      {
         User user1 = new User();
         // set accessor will invoke
         user1.Name = “Ayman Taani";
         // set accessor will invoke
         user1.Location = “Amman";
         // get accessor will invoke
         Console.WriteLine("Name: " + user1.Name);
         // get accessor will invoke
         Console.WriteLine("Location: " + user1.Location);
      }
```

Methods

- ✓ A method is a group of statements that together perform a task. Every C# program has at least one class with a method named Main.

- ✓ To use a method, you need to:
  - ✓ Define the method
  - ✓ Call the method

```
<access_modifier> <return_type> <Method Name>(Parameter List)
{
            Method Body
}
```

Methods

```
public int FindMax(int num1, int num2)
        {
            /* local variable declaration */
            int result;

            if (num1 > num2)
                result = num1;
            else
                result = num2;

            return result;
        }
```

Constructor

Constructors are special types of methods that get executed when an instance of the **class** is created.

```
public class User
{
     //Constructor
     public User()
     {
     //Your Custom Code
     }
}
```

Day 1

1 Clean Code

2 Naming Convention

3 Classes And Objects

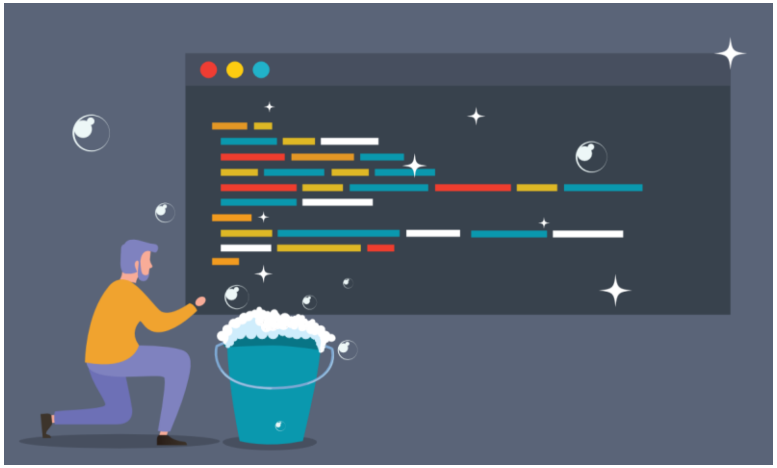
4 Access Modifiers

5 Properties, Methods and Constructors

6 **Exercises**

Exercise 1

We will create a Class (Car Class) in the right way to create the class, and we should follow the clean code concept as much as possible.



Exercise 1

1. Implement a Car class that has fields to store the car's **make**, **model**, **registration**, **year** (of initial registration), and **current value**.
2. Implement a constructor for the Car class which takes parameters corresponding to each of the fields above and constructs an object with these values.
3. Implement a method that **returns the current value of the car**.
4. Implement a method that **returns the year of the car**.
5. Override ToString() to **return a string containing full information about the car**.
6. Write a test class that **constructs a Car and prints its details to the console**.