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

Object-oriented Programming with C#

Tahaluf Training Center 2021

Day 6

1. Interface
2. C# List
3. LINQ

Interface

Interface: in C#, is a keyword, which holds a group of abstract methods and properties, which are to be implemented or used by an abstract or non-abstract class.

These interfaces can hold various methods, indexers, properties and also events as members.

Interface

In simplest terms, An Interface is like a Contract, where every member or component included in the body has to follow the contract, it defines what must be done.

Always defined by the use of the keyword “interface”.

Syntax starts with the interface keyword followed by the name for the interface and then the body:

```
interface <name_for_interface>
{
      //abstract methods
     //abstract properties.
}
```

When to use Interface?

1- Security: When we have to simply hide some features and have to use those later. It is essential to hide a few details while only showing the details important to the user.

2- Multiple Inheritance: In c#, one class can inherit from a simple parent class, inheriting all its features. Multiple Inheritance is not supported in C# for the simple reason to not make C# complex. But with the use of an interface, multiple interfaces can be implemented into a single class.

Let's have a demo and get the task.

Interface


```
interface IAnimal// Interface

{
    void animalSound(); // interface method (does not have a body)
}

// Pig "implements" the IAnimal interface
class Bee : IAnimal
{
    public void animalSound()
    {
        // The body of animalSound() is provided here
        Console.WriteLine("The pig says: Buzzz");
    }
}
```

In The Main :

```
Bee bee = new Bee();  // Create a Pig object

bee.animalSound();
```

Multiple Interface

```
interface IFirstInterface
{
   void myMethod(); // interface method
}

interface ISecondInterface
{
   void myOtherMethod(); // interface method
}

// Implement multiple interfaces
class DemoClass : IFirstInterface, ISecondInterface
{
   public void myMethod()
   {
      Console.WriteLine("Some text..");
   }
   public void myOtherMethod()
   {
      Console.WriteLine("Some other text...");
   }
}
```

Exercise

Create a C# program that implements an IVehicle interface with two methods, one for Drive of type void and another for Refuel of type bool that has a parameter of type integer with the amount of gasoline to refuel. Then create a Car class with a builder that receives a parameter with the car's starting gasoline amount and implements the Drive and Refuel methods of the car.

The Drive method will print on the screen that the car is Driving, if the gasoline is greater than 0. The Refuel method will increase the gasoline of the car and return true.

To carry out the tests, create an object of type Car with 0 of gasoline in the Main of the program and ask the user for an amount of gasoline to refuel, finally execute the Drive method of the car.

Exercise (Solution)

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

```
public interface IVehicle {
        void Drive();
        bool Refuel(int amount);
}
    public class Car : IVehicle
    {
        public int Fuel { get; set; }
        public Car(int fuel)
        { Fuel = fuel; }
        public void Drive()
        {
            if (Fuel > 0)
            { Console.WriteLine("Driving"); }
            else
            { Console.WriteLine("Not fuel"); }
        }
        public bool Refuel(int amount)
        { Fuel += amount;
            return true; }
    }
```

Exercise (Solution)

In The Main :

```
Car car = new Car(0);
int fuel = int.Parse(Console.ReadLine());
if (car.Refuel(fuel))
{
    car.Drive();
}
```

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

Day 6

1 Interface

2 C# List

3 LINQ

C# List

List in C# plays a very important role in data storage and retrieval.

List< T > is a strongly typed list of objects where T represents the type of objects in the list.

It is present under Collections. Generic namespace.

The elements of the list can be accessed through its index number and indexing in the list starts with zero.

The list can be resized dynamically.

If the elements of the list are of reference type then the list can also accept null values.

C# List

It allows the duplication of elements.

```
List<T> list_name = new List<T>();
```

Here T can be of any type like int, string, etc.

And list_name is the user-given name of the list.

C# List

In order to work with List< T >, first, we need to import the System.Collections.Generic namespace in our program.

```
List<string> MyList = new List<string>(3)
{
"tahaluf ",
"trainig",
"Irbid"
};
```

C# List

The elements of the list can be accessed through its index number using ‘for’ or ‘foreach’ loop.

We can perform many operations on a list such as add, insert, search, sort, etc.

It’s dynamic sized.

Let’s have a demo

C# List

```
List<string> MyList = new List<string>(3)
            {
            "tahaluf ",
            "trainig",
            "Irbid"
            };

            MyList.Add("2021");
            MyList.Remove("Irbid");
            MyList.Insert(1, "Mutaz");
            MyList.RemoveAt(3);
            MyList.Contains("2021");

            foreach (var item in MyList)
            {
                Console.WriteLine(item);
            }
```

C# List


```
class Student
{
    int id;
    string name;

    public int Id { get => id; set => id = value; }
    public string Name { get => name; set => name = value; }
}
```

In The Main :

```
var students = new List<Student>() {
                new Student(){ Id = 1, Name="Mutaz"},
                new Student(){ Id = 2, Name="Ahmad"},
                new Student(){ Id = 3, Name="Ali"},
                new Student(){ Id = 4, Name="Sami"}
            };

            foreach (var item in students)
            {
                Console.WriteLine(item.Id +"   "+item.Name);
            }
```

Day 6

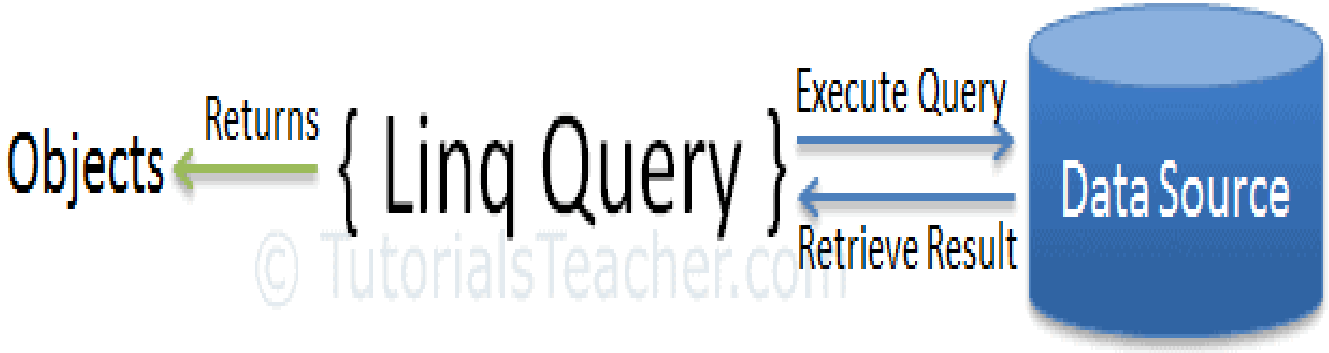
1. Interface
2. C# List
3. **LINQ**

LINQ

LINQ (Language Integrated Query) is uniform query syntax in C# to retrieve data from different sources and formats.

It is integrated in C# , thereby eliminating the mismatch between programming languages and databases, as well as providing a single querying interface for different types of data sources.

SQL is a Structured Query Language used to save and retrieve data from a database.



LINQ Query to Array

```
using System.Linq;

// Data source
string[] names = { "Bill", "Steve", "James", "Mohan" };

// LINQ Query
var myLinqQuery = from name in names
                  where name.Contains('a')
                  select name;
// Query execution
foreach (var name in myLinqQuery)
Console.Write(name + " ");
```

Exercise

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

Write a program in C# Sharp to find the positive numbers from a list of numbers using two where conditions in LINQ Query.

Expected Output:
The numbers within the range of 1 to 11 .

Day Six Task

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

On the E-Learning Portal