**REAL TIME Eg Abstract class && Interface**

Here are some points regarding abstract class.

1. Abstract class can contain abstract members as well as non-abstract members in it.
2. A class can only inherit from one abstract Class.
3. We cannot create object of an abstract class.

it is also user defined type like a class which only contains abstract members in it. These abstract members should be given the implementation under a child class of an interface. A class can be inherited from a class or from an interface.

**Points to remember**

1. Interface contains only abstract methods.
2. We cannot create object of an interface.
3. The default scope for a member in Interface is Public. So, no need to use the Public access specifier in the program.

**NOTE - I**n case of multiple inheritance, use Interface

* **using System;**
* **using System.Collections;**
* **using System.Collections.Generic;**
* **namespace oops1**
* **{**
* **public abstract class Cars**
* **{**
* **//put all the common functions but diffrent implementation in abstract method.**
* **public abstract double price();**
* **public abstract int getTotalSeat();**
* **public abstract string colors();**
* **//put all the common property in normal class**
* **public string Wheel()**
* **{**
* **return "4 wheeler";**
* **}**
* **public string CheckAC()**
* **{**
* **return "AC is available";**
* **}**
* **public string CallFacility()**
* **{**
* **return "Call Facility supported";**
* **}**
* **}**
* **}**

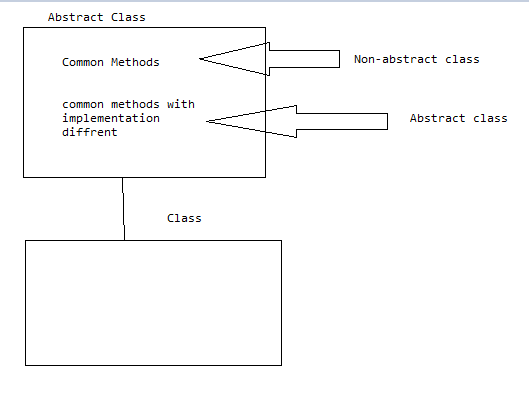
**Now, here is my Toyota class which is derived from Cars abstract class.**

* **using System;**
* **using System.Collections.Generic;**
* **using System.Linq;**
* **using System.Text;**
* **using System.Threading.Tasks;**
* **namespace oops1**
* **{**
* **public class Toyota : Cars**
* **{**
* **public string DiscountPrice()**
* **{**
* **return "20% discount on buying Toyoya Cars";**
* **}**
* **public override double price()**
* **{**
* **return 1000000.00;**
* **}**
* **public override int getTotalSeat()**
* **{**
* **return 5;**
* **}**
* **public override string colors()**
* **{**
* **return "Red";**
* **}**
* **static void Main(string[] args)**
* **{**
* **Toyota Toy = new Toyota();**
* **Console.WriteLine("-------Common property defined commonly in Cars Class----------");**
* **Console.WriteLine(Toy.CallFacility());**
* **Console.WriteLine(Toy.Wheel());**
* **Console.WriteLine(Toy.CheckAC());**
* **Console.WriteLine("-------Own property defined in Toyota class------------");**
* **Console.WriteLine(Toy.DiscountPrice());**
* **Console.WriteLine("-------Common method but implementation is diffrent defined in IExtra Interface------------");**
* **Console.WriteLine("Total ONRoad Price:"+ Toy.price());**
* **Console.WriteLine(Toy.getTotalSeat());**
* **Console.WriteLine(Toy.colors());**
* **Console.ReadLine();**

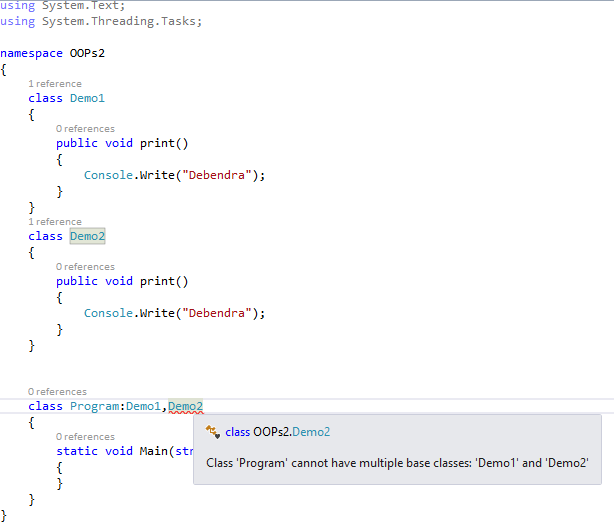
* **}**
* **}**

* **}**

**And thus, the result will be the same.**

****

**in C#, multiple inheritance is not supported. When there is a situation like multiple inheritance, use Interface**

****

**A new feature for the Hyundai car is Introduced called GPS which is not supported in Toyota cars. Then, how can we implement and which way we can implement these?**

**Here, we have certain options like**

* **Go for a new class defining the GPS method and inherit it to the Hyundai Class.**
* **Go for an abstract class and define GPS method and inherit it on Hyundai class and implement the GPS method there.**
* **Directly create a method in Hyundai class and consume it.**
* **Go for Interface**
* **class NewFeatures**
* **{**
* **public void GPS()**
* **{**
* **Console.WriteLine("GPS supported");**
* **}**
* **}**
* **using oops1;**
* **using System;**
* **using System.Collections.Generic;**
* **using System.Linq;**
* **using System.Text;**
* **using System.Threading.Tasks;**
* **namespace oops1**
* **{**
* **public class Hyundai:Cars,NewFeatures**
* **{**

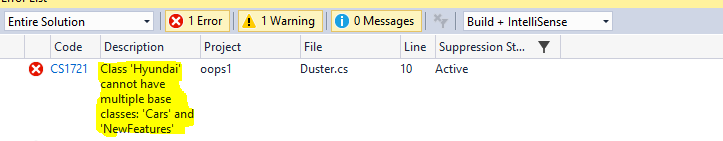
* **static void Main(string[] args)**
* **{**
* **Hyundai hun = new Hyundai();**
* **Console.WriteLine(hun.CallFacility());**
* **Console.WriteLine(hun.Wheel());**
* **Console.WriteLine(hun.CheckAC());**

* **Console.ReadLine();**

* **}**
* **}**

* **}**

**Now, run the program and find out the error.**

****

**This is simple because C# does not support multiple inheritance.**

**CASE 2 - By using Abstract class**

**Now, go for abstract class and see what happens.**

* **public abstract class NewFeatures**
* **{**
* **abstract public void GPS();**
* **}**

**Now, let's try to inherit from abstract class.**

* **using oops1;**
* **using System;**
* **using System.Collections.Generic;**
* **using System.Linq;**
* **using System.Text;**
* **using System.Threading.Tasks;**
* **namespace oops1**
* **{**
* **public class Hyundai:Cars,NewFeatures**
* **{**
* **public override void GPS()**
* **{**
* **Console.WriteLine("GPS supported.");**
* **}**

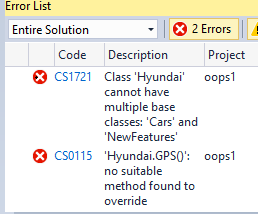
* **static void Main(string[] args)**
* **{**
* **Hyundai hun = new Hyundai();**
* **Console.WriteLine(hun.CallFacility());**
* **Console.WriteLine(hun.Wheel());**
* **Console.WriteLine(hun.CheckAC());**

* **Console.ReadLine();**

* **}**
* **}**

* **}**

**So, here is the error I got.**

****

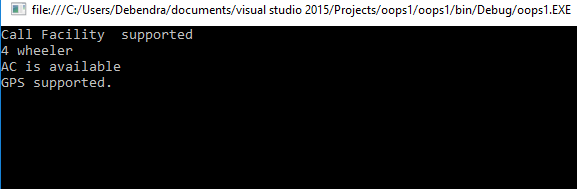
* **interface INewFeatures**
* **{**
* **void GPS();**
* **}**
* **using oops1;**
* **using System;**
* **using System.Collections.Generic;**
* **using System.Linq;**
* **using System.Text;**
* **using System.Threading.Tasks;**
* **namespace oops1**
* **{**
* **public class Hyundai:Cars,INewFeatures**
* **{**
* **public void GPS()**
* **{**
* **Console.WriteLine("GPS supported.");**
* **}**

* **static void Main(string[] args)**
* **{**
* **Hyundai hun = new Hyundai();**
* **Console.WriteLine(hun.CallFacility());**
* **Console.WriteLine(hun.Wheel());**
* **Console.WriteLine(hun.CheckAC());**
* **hun.GPS();**

* **Console.ReadLine();**

* **}**
* **}**

* **}**

****