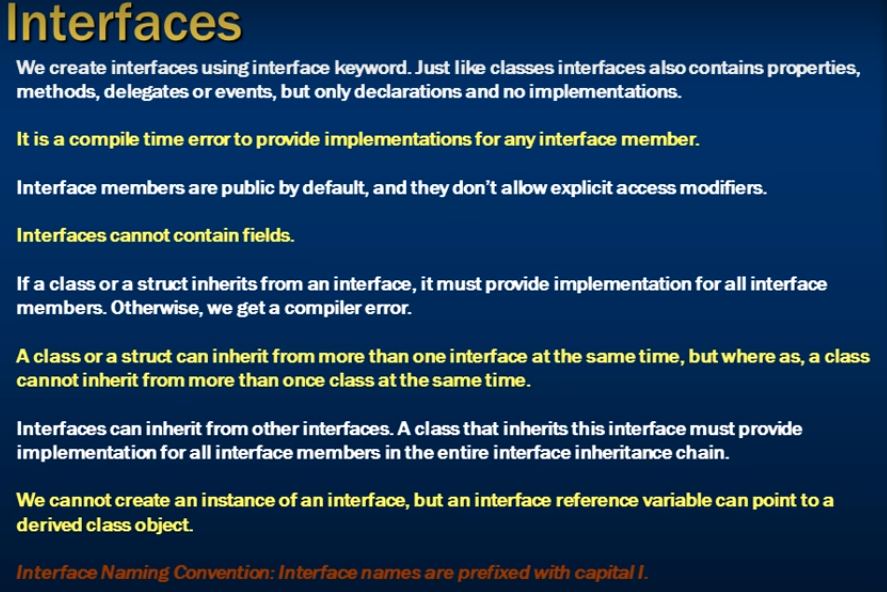
**Rules of INTERFACE:**



\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*GENERAL RULES\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Net.Http;

using System.Reflection.Metadata.Ecma335;

namespace ConsoleApp3

{

interface Iinter1

{

//int a; //interface cant contain FIELD -> CLASS DOES

void print(); //public by default

}

class parent : Iinter1

{

//class gives the IMPLEMENTATION of METHOD in INTERFACE

public void print()

{

Console.WriteLine("Print from parent");

}

}

class Program

{

static void Main(string[] args)

{

parent ob=new parent();

ob.print();

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*ONE INTERFACE is inheriting from OTHER\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Net.Http;

using System.Reflection.Metadata.Ecma335;

namespace ConsoleApp3

{

interface Iinter1

{

void print();

}

interface Iinter2 : Iinter1 //Iinter2 inharites from IInter1.

{

void print2();

}

class parent : Iinter2

{

//when Iinter2 inharites from Iinter1 and a class inharites Iinter2;; THEN BOTH METHOD should be implemented in CLASS.

public void print()

{

Console.WriteLine("Print from parent - Interface1");

Console.ReadKey();

}

public void print2()

{

Console.WriteLine("Print from parent - Interface2");

Console.ReadKey();

}

}

class child

{

}

class Program

{

static void Main(string[] args)

{

parent ob=new parent();

ob.print();

ob.print2();

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*A CLASS CAN INHERIT FROM MULTIPLE NO OF “INTERFACES”\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Net.Http;

using System.Reflection.Metadata.Ecma335;

namespace ConsoleApp3

{

interface Iinter1

{

void print();

}

interface Iinter2 : Iinter1 //Iinter2 inharites from IInter1.

{

void print2();

}

class parent : Iinter2, Iinter1

{

//A CLASS CAN INHERIT FROM MULTIPLE INTERFACES

public void print()

{

Console.WriteLine("Print from parent - Interface1");

Console.ReadKey();

}

public void print2()

{

Console.WriteLine("Print from parent - Interface2");

Console.ReadKey();

}

}

class child

{

}

class Program

{

static void Main(string[] args)

{

parent ob=new parent();

ob.print();

ob.print2();

}

}

}