Assignment 7

Question2

using System;

using System.Collections;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Assignment7question2

{

internal class Program

{

static void Main(string[] args)

{

int emp\_id = 2399;

string name = "Uma Bharti";

string emp\_role = "Analyst3";

string path = @"F:\Employee\_det.txt";

FileStream file = new FileStream(path, FileMode.Open, FileAccess.Write);

Console.WriteLine("File created");

StreamWriter writer = new StreamWriter(file, Encoding.UTF8);

writer.WriteLine("Employee\_Id : {0}",emp\_id);

writer.WriteLine("Employee\_name: {0}", name);

writer.WriteLine("Employee\_role : {0}", emp\_role);

writer.Close();

file.Close();

Console.ReadLine();

}

}

}

Question 3

//Employeeclass

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Serialization\_demo2

{

[Serializable]//The class which we want to serailize must not be

//inherited and we must use serializable concept.

internal class Employee

{

public int Id;

public string Name;

public Employee(int id, string name)

{

Id = id;

Name = name;

}

}

}

Manager Class

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Serialization\_demo2

{

[Serializable]

internal class Manager

{

int pa;

int fa;

int oa;

public Manager(int pa, int fa, int oa)

{

this.pa = pa;

this.fa = fa;

this.oa = oa;

}

}

}

Marketing Executive

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Serialization\_demo2

{

[Serializable]

internal class MarketingExecutive

{

int km;

float tour\_allowance;

float telephone\_all;

public MarketingExecutive(int km, float tour\_allowance, float telephone\_all)

{

this.km = km;

this.tour\_allowance = tour\_allowance;

this.telephone\_all = telephone\_all;

}

}

}

Here we are using serializing

Program.css

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Runtime.InteropServices;

using System.Runtime.Serialization.Formatters.Binary;

using System.Text;

using System.Threading.Tasks;

namespace Serialization\_demo2

{

internal class Program

{

static void Main(string[] args)

{

//string path = @"F:\uma\sample.txt"; //we can create txt file

//string path = @"F:\uma\sample.pdf";//we can create pdf file

// string path = @"F:\uma\sample.png";

//string path = @"F:\uma\sample.docx";

string path2= @"F:\uma\sample3.txt";

string path1 = @"F:\uma\sample2.txt";

string path = @"F:\uma\sample.exe";

Employee emp = new Employee(241, "Uma Bharti");

Manager M = new Manager(8, 13, 3);

FileStream fileMa = new FileStream(path2, FileMode.OpenOrCreate);

MarketingExecutive me = new MarketingExecutive(100, 5.0F, 1000.00F);

FileStream fileme=new FileStream(path1, FileMode.OpenOrCreate);

FileStream filestream =new FileStream(path,FileMode.OpenOrCreate);

BinaryFormatter bf = new BinaryFormatter();//this is very important for serialization

bf.Serialize(fileMa, M); //and deserialization

bf.Serialize(fileme, me);

bf.Serialize(filestream, emp);//here object emp is serializing here and storing itself in

//its filestream; this is hsppenning with the help of Binarryforamtter class and serialize

//method.

//with the help of this file sample.txt can retrieve our object

//again

filestream.Close();

Console.WriteLine("File Created Successfully -> " + path);

Console.WriteLine("fIle Creeated Succesfully->" + path1);

Console.WriteLine("file created successfuly->" + path2);

Console.ReadLine();

//

//Steps for Serailizing

//1.create an instance of File that will store serialize

//object

//2. create a stream from the file object

//3. create an instance of Binary Formatter

//4. Call serialize method of the instance passing

//it stream and object to serialize.

//HOW SERIALIZATION WORK

/\*The object is serialized to a stream

that carries the data.

.The stream may also have information about

the objects type such as its version,culture,and

assembly name

.From that stream,the object can be stored in a database, a file

,or memory.\*/

///USES FOR SERIALIZATION

/\*

\* Serialization allows the developer to save the state of an object

\* and recreate it as needed, providing storage of object as well

\* as data exchange.

\* Through serailzation ,a developer can perform action such as

\* :

\* sending the object to a remote application by using a web service

\* passing an object through a firewall as a json or XmL string.

\* Maintainig security or user-specific inforrmation across applications

\* IMPORTANT POINTS:

\* if you want to make serializable then we must include

\* [seriliazable] attribute on top of your class.

\*

\* If you have applied [serializable] attribute to the class then that

\* class will not be inherited;

\* .Namespace used for this purpose

\* System.Runtime.Serialization.Formatters.Binary;

\* We can serialize data in XML and JSON as

\*

\*/

}

}

}

//DESERIALIZATION

string path = @"F:\uma\sample.txt";

FileStream filestream = new FileStream(path, FileMode.OpenOrCreate);

BinaryFormatter bf = new BinaryFormatter();

Employee emp = (Employee)bf.Deserialize(filestream);

Console.WriteLine("Employee Id : "+emp.Id);

Console.WriteLine("Employee Name :" + emp.Name);

filestream.Close();

Console.ReadLine();