**Implementing Serialization Techniques in .NET**

**Serialization Techniques**

**Objective**: By the end of this lab, you will set up a .NET project for serialization and implement binary, XML, and JSON serialization techniques. You will also compare the output of each method to understand their use cases and performance considerations.

**Step 1: Setting Up the .NET Project for Serialization**

You will set up the basic .NET project structure and install any necessary libraries to implement serialization.

**Steps:**

1. In Visual Studio Code, open the terminal and navigate to the folder where you want to create your project.
2. Run dotnet new console -o SerializationDemo to create a new console project named SerializationDemo.
3. Navigate into the project folder by running cd SerializationDemo.
4. Open the Program.cs file, and add a Person class with UserName and UserAge properties.

**Step 2: Implementing Binary Serialization**

In this step, you will implement binary serialization for the Person object, saving it as a binary file.

**Steps:**

1. In Program.cs, add using System.IO; at the top of the file.
2. Instantiate a Person object and populate it with data, such as a sample name and age.
3. Open a FileStream to create a file named person.dat for storing the binary data.
4. Instantiate a BinaryWriter using the FileStream object.
5. Use BinaryWriter.Write to serialize each property of the Person object manually.
6. Close the writer and the file stream after writing the data.
7. Print a message confirming that binary serialization is complete.

**Step З: lmplementing XML Serialization**

Now, you will implement XML serialization for the Person object.

**Steps:**

1. Add using System.Xml.Serialization; at the top of Program.cs.
2. Instantiate an XmlSerializer object for the Person class.
3. Serialize the Person object and save it to an .xml file.
4. Print a confirmation message after saving the XML file.

**Step 4: Implementing JSON Serialization**

In this step, you will implement JSON serialization for the Person object.

**Steps:**

1. Add using System.Text.Json; at the top of Program.cs.
2. Use JsonSerializer.Serialize to convert the Person object into a JSON string.
3. Save the JSON string to a .json file.
4. Print a confirmation message once the JSON serialization is complete.

**Step 5: Comparing Serialization outputs**

Examine the output files to compare each serialization format’s size, readability, and structure.

**Steps:**

1. Open the .bin, .xml, and .json files created in the previous steps.
2. Take note of the differences in format, readability, and file size.

**Person.cs:**

**namespace** **SerializationDemo.Models**;

**public** **class** **Person**

{

**public** **string** UserName { **get**; **set**; } = **string**.Empty;

**public** **int** UserAge { **get**; **set**; }

}

**BinarySerializerManual.cs:**

**using** **SerializationDemo.Models**;

**namespace** **SerializationDemo.Serializers**;

**public** **static** **class** **BinarySerializerManual**

{

**public** **static** **void** **Serialize**(List<Person> people, **string** filePath)

{

**using** **FileStream** fs = **new** FileStream(filePath, FileMode.Create);

**using** **BinaryWriter** writer = **new** BinaryWriter(fs);

writer.Write(people.Count);

**foreach** (**var** person **in** people)

{

writer.Write(person.UserName);

writer.Write(person.UserAge);

}

}

**public** **static** List<Person> Deserialize(**string** filePath)

{

**using** **FileStream** fs = **new** FileStream(filePath, FileMode.Open);

**using** **BinaryReader** reader = **new** BinaryReader(fs);

**int** count = reader.ReadInt32();

**var** people = **new** List<Person>(count);

**for** (**int** i = **0**; i < count; i++)

{

**string?** name = reader.ReadString();

**int** age = reader.ReadInt32();

**if** (name == **null**)

**throw** **new** **InvalidOperationException**("Binary deserialization failed: name is null.");

people.Add(**new** Person { UserName = name, UserAge = age });

}

**return** people;

}

}

**JsonSerializerHelper.cs:**

**using** **System.Text.Json**;

**using** **SerializationDemo.Models**;

**namespace** **SerializationDemo.Serializers**;

**public** **static** **class** **JsonSerializerHelper**

{

**public** **static** **void** **Serialize**(List<Person> people, **string** filePath)

{

**var** json = JsonSerializer.Serialize(people, **new** JsonSerializerOptions { WriteIndented = **true** });

File.WriteAllText(filePath, json);

}

**public** **static** List<Person> Deserialize(**string** filePath)

{

**string** json = File.ReadAllText(filePath);

**var** people = JsonSerializer.Deserialize<List<Person>>(json);

**if** (people **is** **not** **null**)

**return** people;

**throw** **new** **InvalidOperationException**("JSON deserialization failed: object is null.");

}

}

**XmlSerializerHelper.cs:**

**using** **System.Xml.Serialization**;

**using** **SerializationDemo.Models**;

**namespace** **SerializationDemo.Serializers**;

**public** **static** **class** **XmlSerializerHelper**

{

**public** **static** **void** **Serialize**(List<Person> people, **string** filePath)

{

XmlSerializer serializer = **new** XmlSerializer(**typeof**(List<Person>));

**using** **FileStream** fs = **new** FileStream(filePath, FileMode.Create);

serializer.Serialize(fs, people);

}

**public** **static** List<Person> Deserialize(**string** filePath)

{

XmlSerializer serializer = **new** XmlSerializer(**typeof**(List<Person>));

**using** **FileStream** fs = **new** FileStream(filePath, FileMode.Open);

**var** obj = serializer.Deserialize(fs);

**if** (obj **is** List<Person> people)

**return** people;

**throw** **new** **InvalidOperationException**("XML deserialization failed: object is null or not List<Person>.");

}

}

**Program.cs:**

**using** **SerializationDemo.Models**;

**using** **SerializationDemo.Serializers**;

**namespace** **SerializationDemo**;

**class** **Program**

{

**static** **void** **Main**()

{

**string** dataDir = Path.Combine(Directory.GetCurrentDirectory(), "Data");

Directory.CreateDirectory(dataDir);

**var** people = GeneratePeople(**20**);

**string** binPath = Path.Combine(dataDir, "people.dat");

BinarySerializerManual.Serialize(people, binPath);

Console.WriteLine($"Binary serialization complete: {binPath}");

**string** xmlPath = Path.Combine(dataDir, "people.xml");

XmlSerializerHelper.Serialize(people, xmlPath);

Console.WriteLine($"XML serialization complete: {xmlPath}");

**string** jsonPath = Path.Combine(dataDir, "people.json");

JsonSerializerHelper.Serialize(people, jsonPath);

Console.WriteLine($"JSON serialization complete: {jsonPath}");

Console.WriteLine("\n--- Compare Outputs ---");

Console.WriteLine($"Binary file size: {new FileInfo(binPath).Length} bytes");

Console.WriteLine($"XML file size: {new FileInfo(xmlPath).Length} bytes");

Console.WriteLine($"JSON file size: {new FileInfo(jsonPath).Length} bytes");

Console.WriteLine("\n--- Deserialization Results ---");

**var** binPeople = BinarySerializerManual.Deserialize(binPath);

Console.WriteLine($"Binary restored count: {binPeople.Count}, first: {binPeople[0].UserName} ({binPeople[0].UserAge})");

**var** xmlPeople = XmlSerializerHelper.Deserialize(xmlPath);

Console.WriteLine($"XML restored count: {xmlPeople.Count}, first: {xmlPeople[0].UserName} ({xmlPeople[0].UserAge})");

**var** jsonPeople = JsonSerializerHelper.Deserialize(jsonPath);

Console.WriteLine($"JSON restored count: {jsonPeople.Count}, first: {jsonPeople[0].UserName} ({jsonPeople[0].UserAge})");

}

**static** List<Person> GeneratePeople(**int** count)

{

**var** rnd = **new** Random();

**var** list = **new** List<Person>();

**for** (**int** i = **1**; i <= count; i++)

{

list.Add(**new** Person

{

UserName = $"User{i}",

UserAge = rnd.Next(**18**, **70**)

});

}

**return** list;

}

}