**SERIALISATION**

Object 🡪 stream of bytes 🡪 **Memory**( network memory/hard disk memory/flash) or **File** or **Database**

The object stored in memory or file or database is encoded and difficult to understand.

We can also retrieve our object from the file/ network/database 🡪 Deserialisation

The class made serializable cant be inherited (cant create child class for it)

Main purpose :

1. to store the state of object, so that we can recreate it when needed.
2. Developer can send the object to remote application by web services.
3. Object can be passed from one domain to another.
4. Object can be passed through a firewall by json format
5. Maintain security or user specific information across application

What is stream of bytes?

Byte streams are a sequence of bytes used by program to input and output information.

Steps for serialisation:

1. Create an object of the class to which you want to make it serializable.
2. Create a stream of file object
3. Create an instance of binary formatter
4. Call the serialise method of the instance by passing the stream and object as parameters.

Working:

1. Object is serialised to a stream that carries the data.
2. Stream also have info about the object, version name , assembly and culture.
3. From that stream, the object is stored in database, file or memory.

Namespace used : System.Runtime.Serialization.Formatters.Binary

Example:

Employee emp = new Employee(1, "Ramesh");

string path = @"C:\Users\nikhita\_palla\Documents\Github\Dotnet\_Practice\06-02-2024\Serialisation\Serialisation\_EG\Serialisation\_Output\Sample.txt";

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

BinaryFormatter bF = new BinaryFormatter();

bF.Serialize(stream, emp);

stream.Close(); // if not closed memory gets occupied

Console.WriteLine("File created:" + path);

File stream : FileStream is a class provided in the System.IO namespace that enables reading from and writing to files. It provides low-level access to files, allowing you to manipulate data at the byte level. Here's a basic overview of how you can use FileStream

**Deserialisation**

Inverse process of serialization.

Memory/File/database 🡪 Stream of bytes 🡪 Object

Takes data from file/network/database and converts into object.

Using the state of object we recreate the object to restore the methods and properties of the class.

string path = @"C:\Users\nikhita\_palla\Documents\Github\Dotnet\_Practice\06-02-2024\Serialisation\Serialisation\_EG\Serialisation\_Output\Sample.txt";

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

BinaryFormatter bF = new BinaryFormatter();

Employee obj=(Employee)bF.Deserialize(stream);

Console.WriteLine(obj.Empid);

Console.WriteLine(obj.Name);

Console.ReadLine();

XML serialisation and deserialization

Namespace used : System.Xml.Serialization

using (FileStream fs = new FileStream(path, FileMode.OpenOrCreate))

{

XmlSerializer xmlSerializer = new XmlSerializer(typeof(Employee));

Employee employee = new Employee(1, "Ramesh");

xmlSerializer.Serialize(fs, employee);

}

Deserialization

using (FileStream fs = new FileStream(path, FileMode.Open))

{

XmlSerializer xmlSerializer = new XmlSerializer(typeof(Employee));

Employee emp=(Employee)xmlSerializer.Deserialize(fs);

Console.WriteLine(emp.Name);

Console.WriteLine(emp.Empid);

}

**Formatters**

Formatters are the classes that are used to convert objects into a specific format and vice versa.

1. BinaryFormatter

Formatter is part of System.Runtime.Serialization.Formatter.Binary

Serialises objects into binary format

Efficient but not human readable.

Used for local storage and communication between dotnet applications.

1. XmlSerializer

Part of System.Xml.Serialization.

Serialises objects into XML format.

widely used for structured data exchange, particularly in enterprise applications and web services.

Has higher payload than Json.

1. JsonSerializer

In System.Text.Json.JsonSerializer

formatters serialize objects into JSON format, which is lightweight and human-readable, making it ideal for web APIs and data exchange between different platforms.