Serialization-

The process of storing the state of objects into file called as serialization and process of reading the state of objects from file called as Deserialization.

The process of converting the object from java supported form to network or file supported form is serialization and the process of converting the object from file/network supported form to java supported form is called deserialisation

How to implement serialization in java

By using FileOutputStream and ObjectOutputStream we can achieve serialisaiton. By using FileInputStream and ObjectInputStream, we can achieve deserialisation.

Example-1 Suppose I have one student class in which first name, last name and mobile number. I just want to store that into file name. Then go for serialization.

```
import java.io.Serializable;

public class Student implements Serializable {
        String firstname;
        int lastname;
        String city;
}
```

```
import java.io.FileOutputStream;
import java.io.ObjectOutputStream;

public class SerializeStudent {
        public static void main(String[] args) {
    Student s = new Student();
    s.firstname = "jeevan";
    s.lastname = “kulkarni”;
    s.city = "pune";
    try{
FileOutputStream fos = new FileOutputStream ("D:\\state.txt");
  ObjectOutputStream oos = new ObjectOutputStream(fos);
  oos.writeObject(s);
  oos.flush();
  System.out.println("Serialization is done...");
    }
    catch (Exception e) {
    e.printStackTrace();
   }
```

```
 }
}

import java.io.FileInputStream;
import java.io.ObjectInputStream;

public class DeserializeStudent {
        public static void main(String[] args) {
                try {
                        FileInputStream fis = new
FileInputStream("D:\\state.txt");
                        ObjectInputStream ois = new ObjectInputStream(fis);
                        Object o = ois.readObject(); // Read the object
                        Student s = (Student) o;// convert to student
                        System.out.println(s.firstname);
                        System.out.println(s.lastname);
                        System.out.println(s.city);
                } catch (Exception e) {
                        e.printStackTrace();
                }
        }
}
```

Output:

```
    jeevan
    kulkarni
    Pune
```

- The ObjectOutputStream and ObjectInputStream are used to serialize and de-serialize objects respectively.
- If the superclass implements serializable interface, then all its subclasses will be serializable by default.
- All static members of class are not serialized because static members are related to class only, not to object.
- If we don't want to serialize some fields of class then we use the transient keyword. If any member is declared as transient then it won't be serialized.

- You can use custom serialization you prevent data loss due to use of transient keyword.
- Also to serialize the object partially, you can use Externalization
- In case of array or collection, all the objects of array or collection must be serializable; if any object is not serializable then the serialization will fail.
- The serialization associated with each serializable class has a version number called Serial Version UID.
- It is used during de-serialization to verify that the sender and receiver of a serialized object have loaded classes for that and are compatible with respect to serialization.
- If the receiver is loaded with different version of a class that has different serial version UIDs than the corresponding sender's class, then de-serialization will result in an Invalid Class Exception.
- A Serializable class can declare its own serial version UID explicitly by declaring a field named serial version UID that must be static, final and of type long.
- If a superclass variable is made transient, then after de-serialization, it gives default value like zero or null.
- We can serialize any no of objects to the file but in which order we serialized in the same order only we have to deserialize.

Consider the above same program in which we don't want to serialize the age of a student

```
public class Student implements Serializable {
  String name;
  transient int age;
  String location;
}
```

```
import java.io.FileOutputStream;
import java.io.ObjectOutputStream;
```

```
public class SerializeStudent {
 public static void main(String[] args) {
 Student s = new Student();
  s.name = "abc";
  s.age = 25;//won’t be serialized.
  s.location = "pune";
  try{
 FileOutputStream fos = new FileOutputStream("D:\\state.txt");
  ObjectOutputStream oos = new ObjectOutputStream(fos);
  oos.writeObject(s);
  oos.flush();
  System.out.println("Serialization is done...");
  }
  catch (Exception e) {
  e.printStackTrace();
  }
 }
}

import java.io.FileInputStream;
import java.io.ObjectInputStream;

public class DeserializeStudent {
        public static void main(String[] args) {
                try {
                        FileInputStream fis = new
FileInputStream("D:\\state.txt");
                        ObjectInputStream ois = new ObjectInputStream(fis);
                        Object o = ois.readObject(); // Read the object
                        Student s = (Student) o;// convert to student
                        System.out.println(s.name);
                        System.out.println(s.age); // wont be deserialize,will
printdefault value
                        System.out.println(s.location);
                } catch (Exception e) {
                        e.printStackTrace();
                }
        }
}
```

Output:

```
    abc
    0   // Not Serialized as transient
    Pune
```