Serialization

Serialization in java is a mechanism of writing the state of an object into a byte stream. It is mainly used in Hibernate, RMI, JPA, EJB and JMS technologies. The reverse operation of serialization is called deserialization.

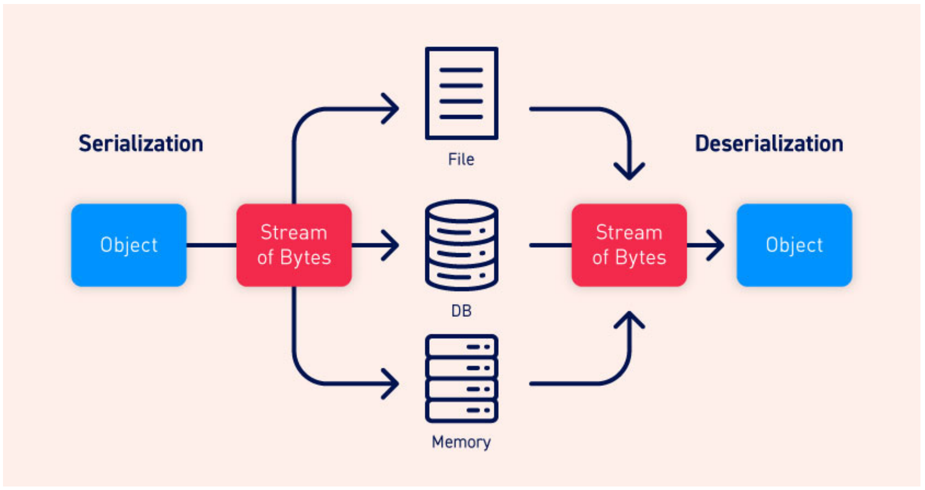

Figure 1: serialization-deserialization

Advantage of Java Serialization : It is mainly used to travel object's state on the network (known as marshaling).

java.io.Serializable interface : Serializable is a marker interface (has no data member and method). It is used to "mark" java classes so that objects of these classes may get certain capability. The Cloneable and Remote are also marker interfaces.

It must be implemented by the class whose object you want to persist.

The String class and all the wrapper classes implements java.io.Serializable interface by default.

ObjectOutputStream class

The ObjectOutputStream class is used to write primitive data types and Java objects to an OutputStream. Only objects that support the java.io.Serializable interface can be written to streams.

Constructor :

```
public ObjectOutputStream(OutputStream out) throws IOException {}
```

Above constructor creates an ObjectOutputStream that writes to the specified OutputStream.

Important Methods :

Method

Description

1) public final void writeObject(Object obj) throws IOException {}
   writes the specified object to the ObjectOutputStream.
   2) public void flush() throws IOException {}
      flushes the current output stream.
      3) public void close() throws IOException {}
         closes the current output stream.

Example of Java Serialization

Example to serialize the object of Student class.

```
import java.io.*;
class Persist{
 public static void main(String args[])throws Exception{
  Student s1 =new Student(211,"John");

  FileOutputStream fout=new FileOutputStream("f.txt");
  ObjectOutputStream out=new ObjectOutputStream(fout);

  out.writeObject(s1);
  out.flush();
  System.out.println("success");
 }
}
```

Deserialization in java

Deserialization is the process of reconstructing the object from the serialized state.It is the reverse operation of serialization.

ObjectInputStream class

An ObjectInputStream deserializes objects and primitive data written using an ObjectOutputStream.

Constructor :

```
public ObjectInputStream(InputStream in) throws IOException {}
```

Above constructor creates an ObjectInputStream that reads from the specified InputStream.

Important Methods :

Method

Description

1) public final Object readObject() throws IOException, ClassNotFoundException{}
   reads an object from the input stream.
   2) public void close() throws IOException {}
      closes ObjectInputStream.

Example of Java Deserialization

Example to deserialize the object of Student class.

```
import java.io.*;
class Depersist{
 public static void main(String args[])throws Exception{

  ObjectInputStream in=new ObjectInputStream(new FileInputStream("f.txt"));
  Student s=(Student)in.readObject();
  System.out.println(s.id+" "+s.name);

  in.close();
 }
}
```

Serialization Rules

Serialization with Inheritance (IS-A Relationship) : If a class implements serializable then all its sub classes will also be serializable. Parent class properties are inherited to subclasses so if parent class is Serializable, subclass would also be.

Serialization with Aggregation (HAS-A Relationship) : If a class has a reference of another class, all the references must be Serializable otherwise serialization process will not be performed. In such case, NotSerializableException is thrown at runtime.

All the objects within an object must be Serializable.

Serialization with static data member : If there is any static data member in a class, it will not be serialized because static is the part of class not object.

Serialization with array or collection : In case of array or collection, all the objects of array or collection must be serializable. If any object is not serialiizable, serialization will be failed.

Externalizable in Java

The Externalizable interface provides the facility of writing the state of an object into a byte stream in compress format. It is not a marker interface.

The Externalizable interface provides two methods: - public void writeExternal(ObjectOutput out) throws IOException - public void readExternal(ObjectInput in) throws IOException

Java Transient Keyword

If you don't want to serialize any data member of a class, you can mark it as transient. **Example :**

```
transient int age; //It will not be serialized
```