**Customized Serialization and Deserialization In Java**

Serialization is a mechanism of converting the state of an object into a byte stream. Deserialization is the reverse process where the byte stream is used to recreate the actual Java object in memory. This mechanism is used to persist the object.

**Why is custom serialization needed?**

During serialization, there may be data loss if we use the ‘transient’ keyword. ‘Transient’ keyword is used on the variables which we don’t want to serialize. But sometimes, it is needed to serialize them in a different manner than the default serialization (such as encrypting before serializing etc.), in that case, we have to use custom serialization and deserialization.

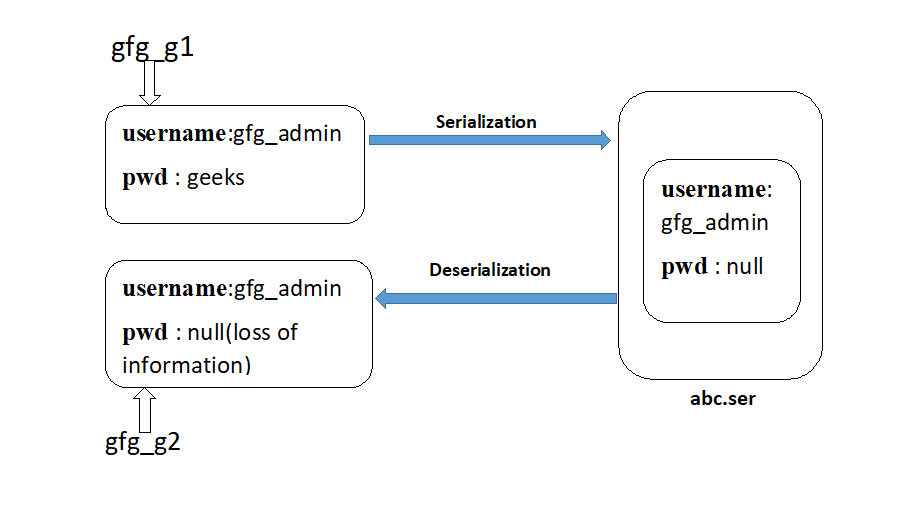
Below program illustrates the above situation of data loss:

|  |
| --- |
| // Java program to illustrate loss of information  // because of transient keyword.  import java.io.\*;    class GfgAccount implements Serializable {        String username = "gfg\_admin";        transient String pwd = "geeks";    }    class CustomizedSerializationDemo {      public static void main(String[] args) throws Exception      {          GfgAccount gfg\_g1 = new GfgAccount();            System.out.println("Username : " + gfg\_g1.username +                                   "    Password : " + gfg\_g1.pwd);            FileOutputStream fos = new FileOutputStream("abc.ser");            ObjectOutputStream oos = new ObjectOutputStream(fos);            // writeObject() method present in GfgAccount class          // will be automatically called by jvm          oos.writeObject(gfg\_g1);            FileInputStream fis = new FileInputStream("abc.ser");            ObjectInputStream ois = new ObjectInputStream(fis);            // readObject() method present GfgAccount class          // will be automatically called by jvm          GfgAccount gfg\_g2 = (GfgAccount)ois.readObject();            System.out.println("Username : " + gfg\_g2.username +                                 "      Password : " + gfg\_g2.pwd);      }  } |

**Output**:

Username : gfg\_admin Password : geeks

Username : gfg\_admin Password : null

  
In the above image example, before serialization, Account object can provide proper username and password but deserialization of Account object provides only username and not the password. This is due to declaring password variable as transient.

Hence during default serialization, there may be a chance of loss of information because of the transient keyword. To recover this loss, we will have to use **Customized Serialization.**

Customized serialization can be implemented using the following two methods:

1. **private void writeObject(ObjectOutputStream oos) throws Exception**: This method will be executed automatically by the jvm(also known as Callback Methods) at the time of serialization. Hence to perform any activity during serialization, it must be defined only in this method.
2. **private void readObject(ObjectInputStream ois) throws Exception**: This method will be executed automatically by the jvm(also known as Callback Methods) at the time of deserialization. Hence to perform any activity during deserialization, it must be defined only in this method.

**Note:** While performing object serialization, we have to define the above two methods in that class.

|  |
| --- |
| // Java program to illustrate customized serialization  import java.io.\*;    class GfgAccount implements Serializable {        String username = "gfg\_admin";        transient String pwd = "geeks";        // Performing customized serialization using the below two methods:      // this method is executed by jvm when writeObject() on      // Account object reference in main method is      // executed by jvm.      private void writeObject(ObjectOutputStream oos) throws Exception      {          // to perform default serialization of Account object.          oos.defaultWriteObject();            // epwd (encrypted password)          String epwd = "123" + pwd;            // writing encrypted password to the file          oos.writeObject(epwd);      }        // this method is executed by jvm when readObject() on      // Account object reference in main method is executed by jvm.      private void readObject(ObjectInputStream ois) throws Exception      {          // performing default deserialization of Account object          ois.defaultReadObject();            // deserializing the encrypted password from the file          String epwd = (String)ois.readObject();            // decrypting it and saving it to the original password          // string starting from 3rd  index till the last index          pwd = epwd.substring(3);      }  }    class CustomizedSerializationDemo {      public static void main(String[] args) throws Exception      {          GfgAccount gfg\_g1 = new GfgAccount();            System.out.println("Username :" + gfg\_g1.username +                             "       Password :" + gfg\_g1.pwd);            FileOutputStream fos = new FileOutputStream("abc.ser");            ObjectOutputStream oos = new ObjectOutputStream(fos);            // writeObject() method on Account class will          // be automatically called by jvm          oos.writeObject(gfg\_g1);            FileInputStream fis = new FileInputStream("abc.ser");            ObjectInputStream ois = new ObjectInputStream(fis);            GfgAccount gfg\_g2 = (GfgAccount)ois.readObject();            System.out.println("Username :" + gfg\_g2.username +                               "       Password :" + gfg\_g2.pwd);      }  } |

**Output**:

Username :gfg\_admin Password :geeks

Username :gfg\_admin Password :geeks

