Andrea Marcelli
CPSC 24500 NET02
April 10, 2023
Serialization

Serialization

Let's start by defining both the serialization and deserialization processes. "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." (GeekForGeeks). Serialization is commonly used to simplify data exchange giving the ability to convert objects into files that can be read by other systems. Also, as stated previously serialization gives persistence to data by storing it in the memory stream. Some important aspects to note about serialization are that it can be challenging and complex when there is a need to handle a big amount of data and can be hard to analyze since it can be not human-readable.

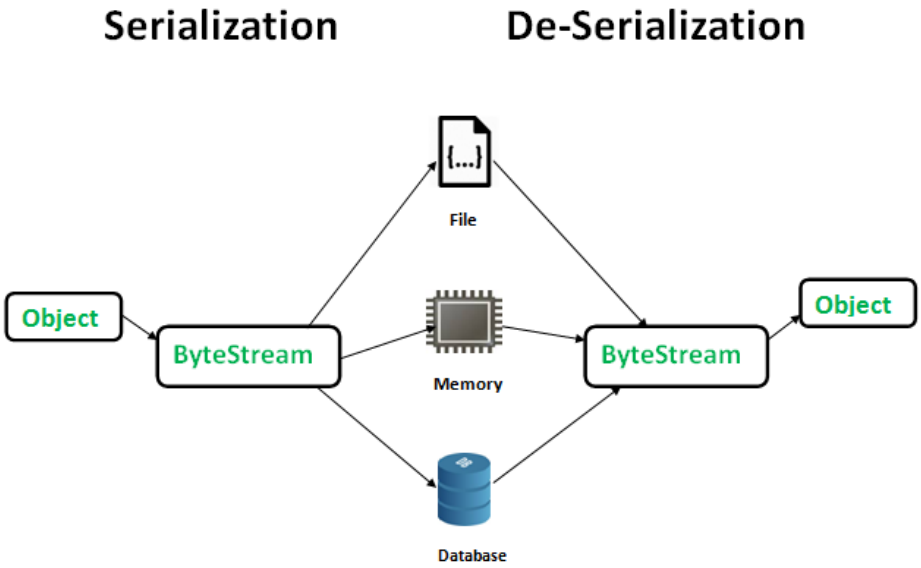

Figure 1.0 (source: https://media.geeksforgeeks.org/wp-content/cdn-uploads/gq/2016/01/serialize-deserialize-java.png)

Let's look at some examples of the two cases of serialization and deserialization:

```
import java.io.*;
public class SerializeDemo {

   public static void main(String [] args) {
      Employee e = new Employee();
      e.name = "Reyan Ali";
      e.address = "Phokka Kuan, Ambehta Peer";
      e.SSN = 11122333;
```

```
      e.number = 101;

      try {
         FileOutputStream fileOut =
         new FileOutputStream("/tmp/employee.ser");
         ObjectOutputStream out = new ObjectOutputStream(fileOut);
         out.writeObject(e);
         out.close();
         fileOut.close();
         System.out.printf("Serialized data is saved in
/tmp/employee.ser");
      } catch (IOException i) {
         i.printStackTrace();
      }
   }
}
```

(Source of the example in references)

As we can see the code line after the try statement is the output stream created in order to write inside a file that eventually will be used for data exchange between systems. While the components above define the creation of an object created in a separate class. The program above is the main driver program, there is also the Employee class which is the serialized object. The class is shown below:

```
public class Employee implements java.io.Serializable {
   public String name;
   public String address;
   public transient int SSN;
   public int number;

   public void mailCheck() {
      System.out.println("Mailing a check to " + name + " " +
address);
   }
}
```

(Source of the example in references)

Now let's see how we would deserialize the file that we created:

Andrea Marcelli
CPSC 24500 NET02
April 10, 2023
Serialization

```
import java.io.*;
public class DeserializeDemo {

   public static void main(String [] args) {
      Employee e = null;
      try {
         FileInputStream fileIn = new
FileInputStream("/tmp/employee.ser");
         ObjectInputStream in = new ObjectInputStream(fileIn);
         e = (Employee) in.readObject();
         in.close();
         fileIn.close();
      } catch (IOException i) {
         i.printStackTrace();
         return;
      } catch (ClassNotFoundException c) {
         System.out.println("Employee class not found");
         c.printStackTrace();
         return;
      }

      System.out.println("Deserialized Employee...");
      System.out.println("Name: " + e.name);
      System.out.println("Address: " + e.address);
      System.out.println("SSN: " + e.SSN);
      System.out.println("Number: " + e.number);
   }
}
```

(Source of the example in references)

As we can see we are using the InputStream object in order to read the content of the file specified. Once read the file, its contents will be saved as their corresponding variables and printed on the console.

Usually, serialization can be useful in systems that require efficient data exchange, messaging systems, databases, etc…

Andrea Marcelli
CPSC 24500 NET02
April 10, 2023
Serialization

References

Saraf, N. (n.d.). Serializing and Deserializing Objects. Oracle. Retrieved from https://docs.oracle.com/javase/tutorial/jndi/objects/serial.html#:~:text=To%20serialize%20an%20object%20means,io.

GeeksforGeeks. (2022, February 12). Serialization in Java - GeeksforGeeks. Retrieved from https://www.geeksforgeeks.org/serialization-in-java/

TutorialsPoint. (2022, March 28). Java Serialization. Retrieved from https://www.tutorialspoint.com/java/java_serialization.htm