# Serialization and Deserialization in Java

10 October 2025 08:46

**What is Serialization in Java?**

Serialization in Java is the process of converting an object into a byte stream, so that it can be:

- Saved to a file
- Sent over a network
- Or stored in a database.

Later the byte string can be deserialized. That is converted back into the original Java object.

In short:

Serialization = Object -> Byte Stream
Deserialization = Byte Stream -> Object

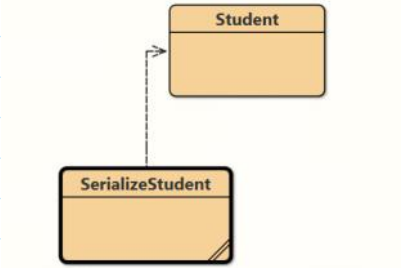
**Why Serialization?**

Serialization is used when:

- You want to store the state of an object permanently (example, Save it in a file.)
- You want to send objects over a network. (Example in socket programming, RMI, etc.)
- You want to deep copy complex object.

**How Serialization Works in Java?**

- To make class serializable it must:
  1. Implement the **java.io.Serializable** interface, which is a marker interface(it has no methods).
  2. All non-transient, non-static fields of the class will be serialized.
  3. Use **ObjectOutputStream** to serialize and **ObjectInputStream** to deserialize.



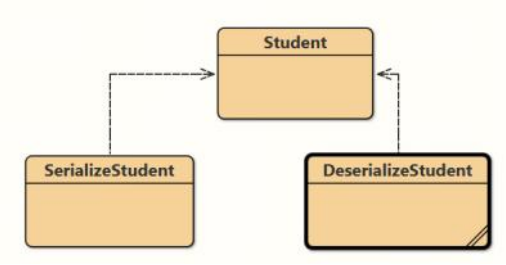

```
package FileHandling;
import java.io.*;


/**
 * Write a description of class Student here.
 *
 * @author (your name)
 * @version (a version number or a date)
 */
public class Student implements Serializable
{
   //Step-1 Create a serializable class
```

```
    private static final long serialVersionUID = 1L; //version control for serialization

    private int id;
    private String name;
    transient private int age; //tranisient -> will not be serialized.

    public Student(int id, String name, int age){
        this.id = id;
        this.name = name;
        this.age = age;
    }
}

package FileHandling;
import java.util.*;
import java.io.*;


/**
 * Write a description of class SerializeStudent here.
 *
 * @author (your name)
 * @version (a version number or a date)
 */
public class SerializeStudent
{
    public static void main(String[] args){
        System.out.print("\fEnter number of students: ");
        Scanner sc = new Scanner(System.in);
        Student[] st = new Student[sc.nextInt()];
        //initialize memory to each reference variable to Student[] st
        for(int i = 0; i < st.length; i++){
            System.out.print("Enter student id: ");
            int id = sc.nextInt();
            System.out.print("Enter name: ");
            String name = sc.nextLine();
            sc.nextLine();
            System.out.print("\nAge: ");
            int age = sc.nextInt();
            st[i] = new Student(id, name, age);
        }
        sc.nextLine();
        System.out.print("\nEnter file name: ");
        String fileName = sc.nextLine();
        try(FileOutputStream fos = new FileOutputStream("./DataFiles/"+fileName);
        ObjectOutputStream oos = new ObjectOutputStream(fos)){
            for(int i = 0; i < st.length; i++){
                oos.writeObject(st[i]); //Serialization happens here
                System.out.print("\nObject has been serialized and saved as "+fileName);
            }
        }
```

```
        catch(IOException ioe){
            System.err.print("\nNetwork error.");
        }
    }
}
```

```
BlueJ: Terminal Window - JavaSem2
Options
Enter number of students: 3
Enter student id: 1
Enter name: Jack

Age: 21
Enter student id: 2
Enter name: Emily

Age: 21
Enter student id: 3
Enter name: Jiya

Age: 20

Enter file name: Student.ser

Object has been serialized and saved as Student.ser
Object has been serialized and saved as Student.ser
Object has been serialized and saved as Student.ser
```

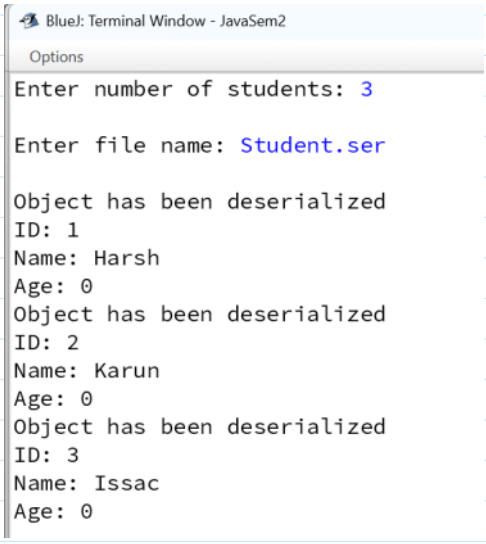
```
package FileHandling;
import java.util.*;
import java.io.*;


/**
 * Write a description of class DeserializeStudent here.
 *
 * @author (your name)
 * @version (a version number or a date)
 */
public class DeserializeStudent
{
    public static void main(String[] args){
        System.out.print("\fEnter number of students: ");
        Scanner sc = new Scanner(System.in);
        Student[] st = new Student[sc.nextInt()];
        sc.nextLine();
        System.out.print("\nEnter file name: ");
        String fileName = sc.nextLine();
        try(FileInputStream fis = new FileInputStream("./DataFiles/"+fileName);
        ObjectInputStream ois = new ObjectInputStream(fis)){
            for(int i = 0; i < st.length; i++){
                st[i] = (Student)ois.readObject(); //De-Serialization happens here
                System.out.print("\nObject has been deserialized");
                System.out.print("\n" + st[i]); //printing information from the Object.
            }
```




Serialized and Deserialized classes

```
        }
        catch(ClassNotFoundException cnfe){
            System.err.print("\nClass is not defined.");
        }
        catch(IOException ioe){
            System.err.print("\nNetwork error.");
        }
    }
}
```

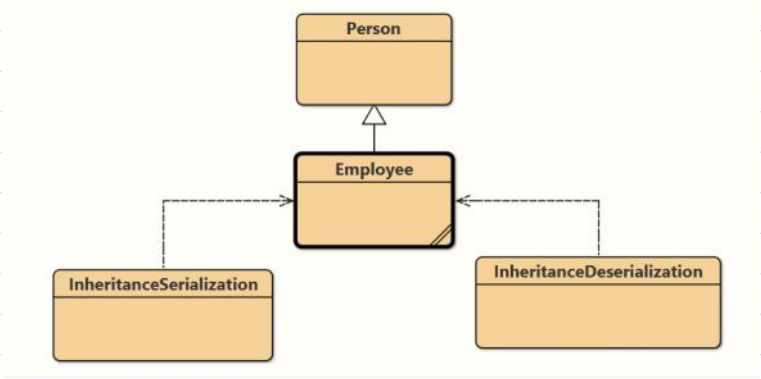
Output:




**Important points**

| Concept | Description |
|---|---|
| Serialized interface | Marker interface that tells JVM the class can be serialized. |
| serialVersionUID | Unique ID is used to verify compatibility between serialized and deserialized classes. |
| transient keyword | Used to skip fields during serialization. |
| static Fields | Are not serialized (because they belong to class, not to instance) |
| Deserialization | Restores the object's state from byte stream to memory. |

**Inheritance Serialization and Deserialization**




```
package FileHandling;
import java.io.*;


/**
 * Write a description of class Person here.
 *
 * @author (your name)
 * @version (a version number or a date)
 */
public class Person implements Serializable
{
   private String name;

   public Person(String name){
      this.name = name;
   }

   @Override
   public String toString(){
      return "Name: " + name;
   }
}

package FileHandling;


/**
 * Write a description of class Employee here.
 * //As super class Person is Serialized, sub-class is also Serialized
 * @author (your name)
 * @version (a version number or a date)
 */

public class Employee extends Person
{
   private int empId;

   public Employee(String name, int empId){
      super(name);
```

```
        this.empId = empId;
    }

    @Override
    public String toString(){
        return super.toString() + "\nID: " + empId;
    }
}

package FileHandling;
import java.util.*;
import java.io.*;


/**
 * Write a description of class InheritanceSerialization here.
 *
 * @author (your name)
 * @version (a version number or a date)
 */
public class InheritanceSerialization
{
    public static void main(String[] args){
        Scanner sc = new Scanner(System.in);
        System.out.print("\fEnter number of employees: ");
        Employee[] empArr = new Employee[sc.nextInt()];
        //Feeding data into employee objects
        for(int i = 0; i < empArr.length; i++){
            System.out.print("Enter employee id: ");
            int id = sc.nextInt();
            sc.nextLine();
            System.out.print("Enter name: ");
            String name = sc.nextLine();
            empArr[i] = new Employee(name, id);
        }
        String path = "./DataFiles/";
        System.out.print("\nEnter file name: ");
        String fileName = sc.nextLine();

        try( ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream(path+fileName, true))){
            for(int i = 0; i < empArr.length; i++){
                oos.writeObject(empArr[i]);
            }
        }
        catch(IOException ioe){
            System.out.print("\nNetwork not available!!!");
        }
    }
}

package FileHandling;
```

```
import java.util.*;
import java.io.*;


/**
 * Write a description of class InheritanceDeserialization here.
 *
 * @author (your name)
 * @version (a version number or a date)
 */
public class InheritanceDeserialization
{
   public static void main(String args[]){
      Scanner sc = new Scanner(System.in);
      System.out.print("\f");
      String path = "./DataFiles/";
      System.out.print("Enter file name: ");
      String fileName = sc.nextLine();
      System.out.print("Enter number of employees: ");
      Employee[] earr = new Employee[sc.nextInt()];
      try(ObjectInputStream ois = new ObjectInputStream(new FileInputStream(path+fileName))){
         System.out.print("\nEmployee details: ");
         for(int i = 0; i < earr.length; i++){
            earr[i] = (Employee)ois.readObject();
            System.out.print("\n"+earr[i]);
         }
      }
      catch(ClassNotFoundException cnfe){
         System.err.print("\nClass Employee is not defined!!!");
      }
      catch(IOException ioe){
         System.err.print("\nNetwork error!!!!");
      }
   }
}
```

**In the following code we handled the EOFException to stop the execution of infinite for loop while reading the file**

```
package FileHandling;
import java.util.*;
import java.io.*;

/**
 * Write a description of class InheritanceDeserialization here.
 *
 * @author (your name)
 * @version (a version number or a date)
 */
public class InheritanceDeserialization
{
```

```
    public static void main(String args[]){
        Scanner sc = new Scanner(System.in);
        System.out.print("\f");
        String path = "./DataFiles/";
        System.out.print("Enter file name: ");
        String fileName = sc.nextLine();
        //System.out.print("Enter number of employees: ");
        //Employee[] earr = new Employee[sc.nextInt()];
        try(ObjectInputStream ois = new ObjectInputStream(new FileInputStream(path+fileName))){
            System.out.print("\nEmployee details: ");
            /*for(int i = 0; i < earr.length; i++){
            earr[i] = (Employee)ois.readObject();
            System.out.print("\n"+earr[i]);
            }*/
            for(;;){
                Employee earr = (Employee)ois.readObject();
                System.out.print("\n"+earr);
            }
        }
        catch(EOFException eofe){
            System.out.print("\nEnd of file.");
        }
        catch(ClassNotFoundException cnfe){
            System.err.print("\nClass Employee is not defined!!!");
        }
        catch(IOException ioe){
            System.err.print("\nNetwork error!!!!");
        }
    }
}
```

**Output:**