BenchResources.Net

Java, Collection, JDBC, Spring, Web Services, Maven, Android, Oracle SOA-OSB & Open Source

Serialization and De-Serialization in Java

November 2, 2016 SJ Serialization 0

In this article, we will discuss Java ***serialization*** and ***de-serialization*** in detail

Serialization:

- The process of ***writing a state of an Object*** to a file is called ***Serialization***
- In other words, the ***process of saving an Object's state to a file*** is called ***Serialization***
- But practically, it is the ***process of converting & storing Java Object's state from heap memory*** (in byte stream) ***to file supported form*** (in binary format)

SEARCH TUTORIALS

SEARCH …

SUBSCRIBE VIA EMAIL

Join 194 other subscribers

Email Address

SUBSCRIBE

POPULAR ARTICLES

JDBC: An example to connect MS Access database in Java 8

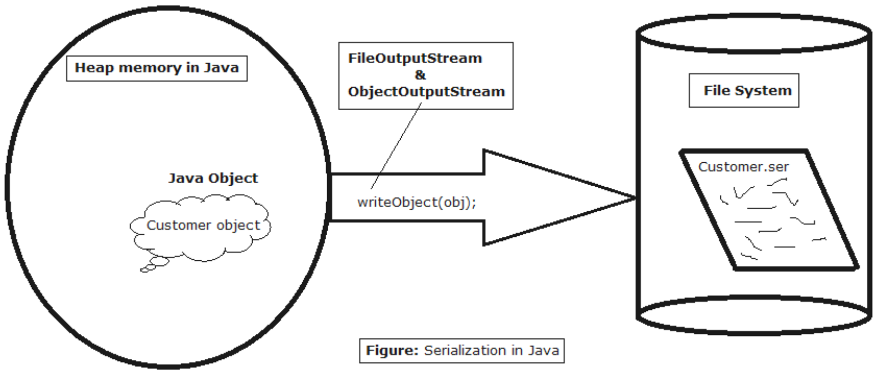

Figure: Serialization in Java

Demo example on Java Serialization

Using *ObjectOutputStream* and *FileOutputStream* classes available from *java.io* package, we can *serialize* an Object to file

Customer class

- Customer class is the one to be ***serialized***
- Therefore, it is must to implement ***java.io.Serializable*** interface
- Consists of ***3 member variables*** namely
- Two integer member (customer id and customer age) and String member (customer name)

Customer.java

```
package in.bench.resources.serialization;

import java.io.Serializable;

public class Customer implements Serializable {

    // member variables
    int customerId;
    String customerName;
    int customerAge;

    // 3-arg parametrized constructor
    public Customer(int customerId,
            String customerName, int customerAg
        super();
        this.customerId = customerId;
        this.customerName = customerName;
        this.customerAge = customerAge;
    }

```

Spring JDBC: An example on JdbcTemplate using Annotation
Java JDBC: An example to connect MS Access database
Oracle OSB 12c: Service Callout and Routing Table example
Oracle OSB 12c: Hello World service with both Business and Proxy Service

```
    // overriding toString() method
    @Override
    public String toString() {
        return "Customer [customerId=" + custom
                + " customerName=" + customerNa
                + " customerAge=" + customerAge
                + "]";
    }
}
```

Note: An Object said to be Serializable only if corresponding class implements *java.io.Serializable* interface

SerializeCustomer.java

```
package in.bench.resources.serialization;

import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.ObjectOutputStream;

public class SerializeCustomer {

    public static void main(String[] args) {

        // create a customer object using 3-arg
        Customer customer = new Customer(101, "

        // creating output stream variables
        FileOutputStream fos = null;
        ObjectOutputStream oos = null;

        try {
            // for writing or saving binary dat
            fos = new FileOutputStream("Custome

            // converting java-object to binary
            oos = new ObjectOutputStream(fos);

            // writing or saving customer objec
            oos.writeObject(customer);
            oos.flush();
            oos.close();
        }
        catch (FileNotFoundException fnfex) {
            fnfex.printStackTrace();
        }
        catch (IOException ioex) {
            ioex.printStackTrace();
        }

        System.out.println("Customer object sav
    }
}
```

Output:

```
Customer object saved to Customer.ser file
```

Note: Object must implement *java.io.Serializable*, otherwise Run time exception will be thrown saying *NotSerializableException*

De-Serialization:

- The ***process of reading a state of an Object from a file*** is called *De-Serialization*
- But practically, it is the ***process of converting & re-storing Java Object's state into heap memory from file supported form*** (which is in binary format)

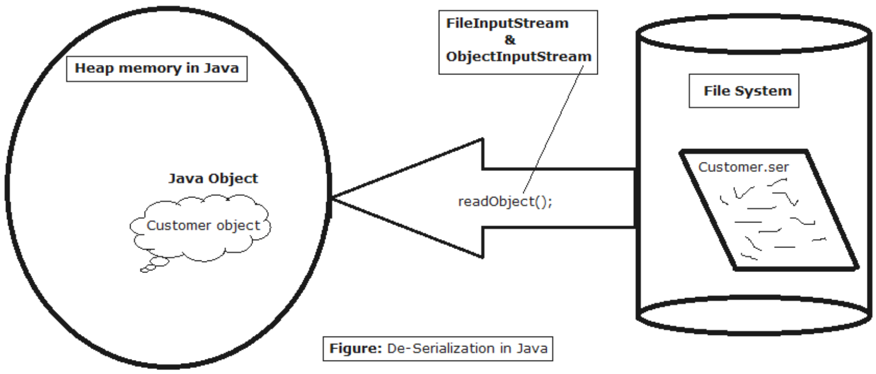

Figure: De-Serialization in Java

Demo example on Java De-Serialization

Using *ObjectInputStream* and *FileInputStream* classes available from *java.io* package, we can ***de-serialize*** an Object from file

Note: we will use ***Customer object*** from above example to de-serialize (and make sure class implements *java.io.Serializable* interface)

DeSerializeCustomer.java

```
package in.bench.resources.serialization;
```

```

import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.io.ObjectInputStream;

public class DeSerializeCustomer {

    public static void main(String[] args) {

        // creating input stream variables
        FileInputStream fis = null;
        ObjectInputStream ois = null;

        // creating customer object reference
        // to hold values after de-serializatio
        Customer customer = null;
        try {
            // reading binary data
            fis = new FileInputStream("Customer

            // converting binary-data to java-o
            ois = new ObjectInputStream(fis);

            // reading object's value and casti
            customer = (Customer) ois.readObjec
        }
        catch (FileNotFoundException fnfex) {
            fnfex.printStackTrace();
        }
        catch (IOException ioex) {
            ioex.printStackTrace();
        }
        catch (ClassNotFoundException ccex) {
            ccex.printStackTrace();
        }

        System.out.println("Customer object de-
                + "Customer.ser file\nLet's pri

        // printing customer object to console
        System.out.println(customer);
    }
}
```

Output:

```
Customer object de-serialized from Customer.ser
Let's print to console...

Customer [customerId=101, customerName=SJ, custo
```

References:

https://docs.oracle.com/javase/7/docs/api/java/io/Serializable.html
https://docs.oracle.com/javase/7/docs/platform/serialization/spec/serial-arch.html

https://docs.oracle.com/javase/7/docs/api/java/io/ObjectOutputStream.html
https://docs.oracle.com/javase/7/docs/api/java/io/ObjectInputStream.html
https://docs.oracle.com/javase/7/docs/api/java/io/FileOutputStream.html
https://docs.oracle.com/javase/7/docs/api/java/io/FileInputStream.html

Read Also:

- Java Serialization and De-Serialization Tutorial Index
- Serializable interface
- Transient keyword with Serialization in Java
- Transient keyword with static variable in Serialization
- Transient keyword with final variable in Serialization
- Serializing a variable with transient modifier or keyword
- Order of Serialization and De-Serialization
- Serialization with Aggregation
- Serialization with Inheritance
- Externalizable interface with example
- Serializable v/s Externalizable
- Importance of SerialVersionUID in Serialization
- Singleton Design pattern with Serialization
- How to stop Serialization in Java
- How to construct a singleton class in a multi-threaded environment in Java
- How to serialize and de-serialize ArrayList in Java

Happy Coding !!
Happy Learning !!

« **Serializable interface**

Related Posts:

1. **Serializable interface**
2. **Transient keyword with Serialization in Java**
3. **Transient keyword with static variable in Serialization**

4. Serializing a variable with transient modifier or keyword

DESERIALIZATION JAVA SERIALIZABLE

SERIALIZATION

0 Comments **BenchResources.Net** Ava

♡ **Recommend** **Share** **Sort by Best**

Start the discussion…

Be the first to comment.

ALSO ON **BENCHRESOURCES.NET**

Apache CXF JAX-WS: Web Service using Top-Down

1 comment • 2 years ago

basanta hota — Any one can help me out same example i tried , even i am not pass any

5 important keywords in Java Exception handling

1 comment • 2 years ago

Saurabh Naik — Various possible combinations of try catch finally 1) try{} catch(x x)

Java Collection Interview question and answers

2 comments • 2 years ago

Adil Khan — A small request,would be happy to see if you write article on how

How to remove an entry from HashMap in Java 8

8 comments • a year ago

BenchResources.Net — Glad that it helped you !!Following article covers how to remove