Serialization

- Definition
- Binary serialization
- Recursive serialization, transient
- Deserialization
- Text files
- Example

Serialization

The process of translating data structures or object state into a format that can be stored.

For example: in a file or memory buffer, or transmitted across a network connection link.

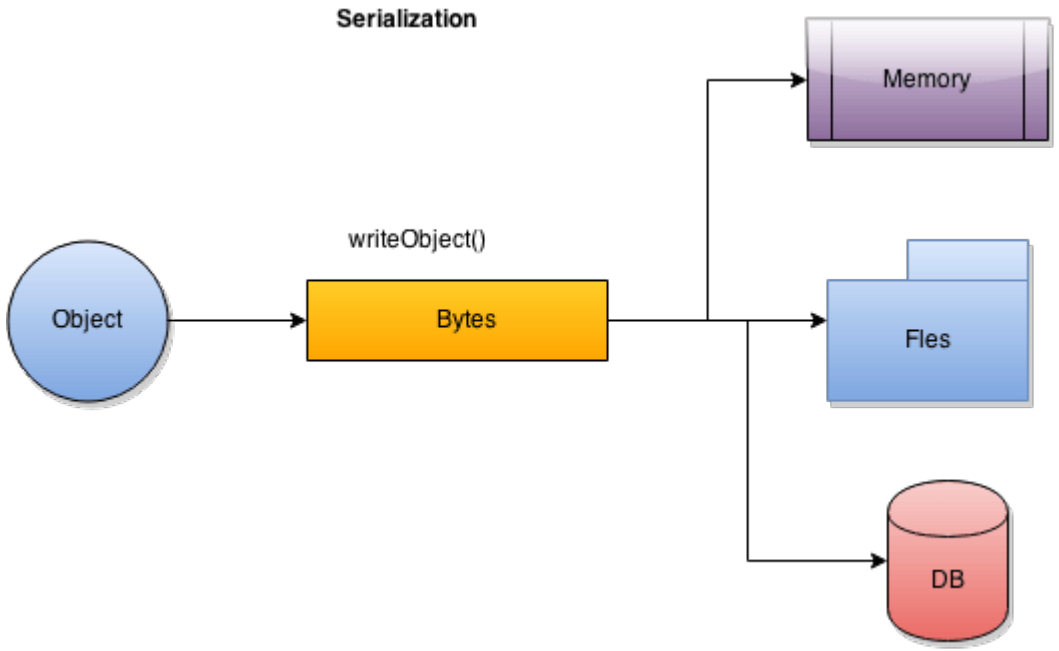
Serialization is reversible process.

Object can be reconstructed later in the same or another computer environment.

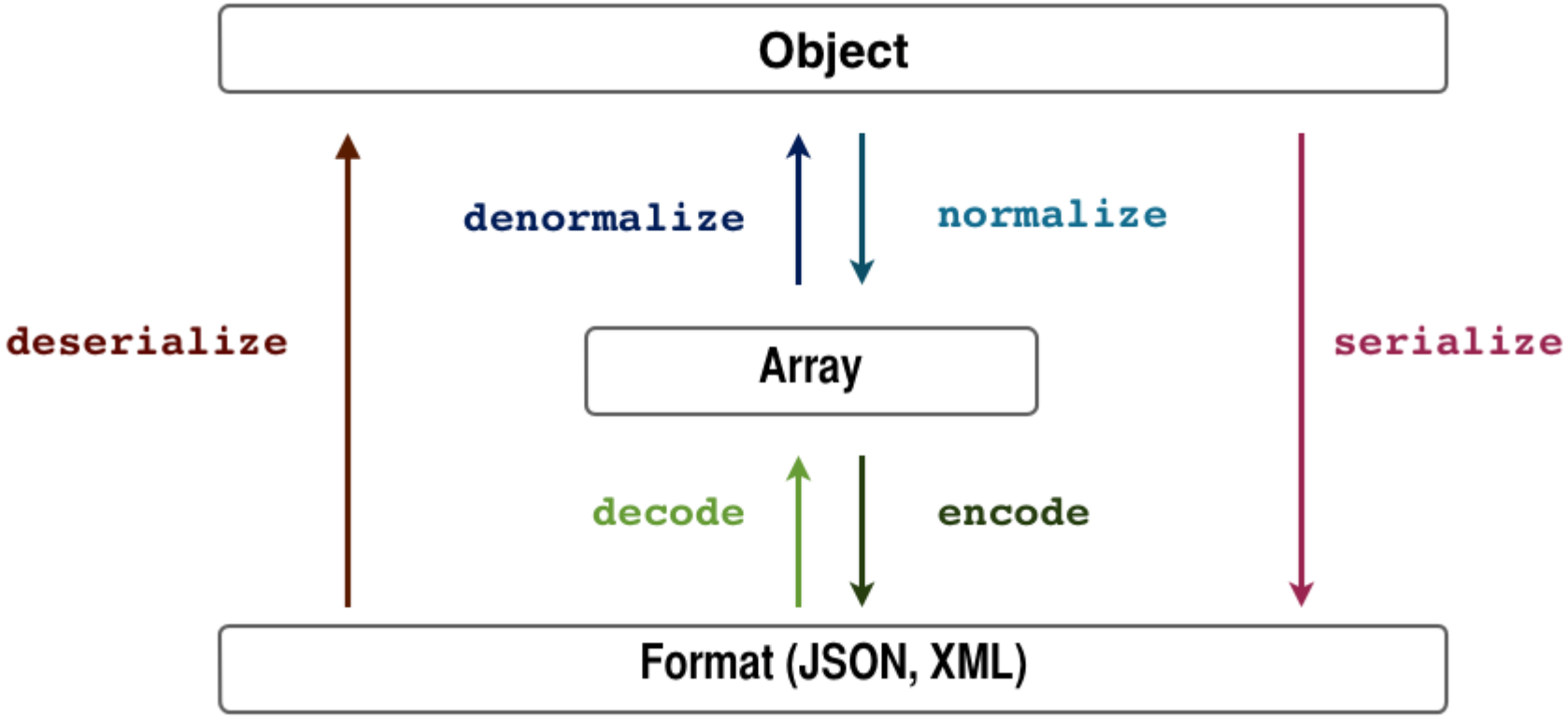
Examples of Serialization

- byte array in memory
- binary file
- xml file
- json string
- blob in DB

Binary serialization



Xml or json serialization



Binary serialization

Writing of the object to the byte array

```
public static byte[] serialize(Object obj) throws IOException {
    ByteArrayOutputStream out = new ByteArrayOutputStream();
    ObjectOutputStream os = new ObjectOutputStream(out);
    os.writeObject(obj);
    return out.toByteArray();
}
```

Reading ot the object form the byte array

```
public static Object deserialize(byte[] data) throws IOException, ClassNotFoundException {
    ByteArrayInputStream in = new ByteArrayInputStream(data);
    ObjectInputStream is = new ObjectInputStream(in);
    return is.readObject();
}
```

ObjectOutputStream knows how to convert your object in to byte array.

To be serializable

- class must implement interface Serializable
- all super classes must be serializable
- all valiables of the class must be Serializable

If you do not need to serialize a variable you can use key word *transient*

ObjectOutputStream uses Reflection to serialize and de-serialize objects.

Wrining of the byte array to the file

Without any libraries

```
try(FileOutputStream stream = new FileOutputStream(path)) {
    stream.write(bytes);
}
```

With Google Guava

```
Files.write(bytes, new File(path));
```

With Apache Commons

```
FileUtils.writeByteArrayToFile(new File(path), bytes);
```

Writing of the object to the file

```
//let's open output stream to the file
FileOutputStream out = new FileOutputStream(fileName);
//let's use bufferization
BufferedOutputStream bout = new BufferedOutputStream(out);
//we need to write an object, so let's use
ObjectOutputStream dout = new ObjectOutputStream(bout);
//writing
dout.writeObject(object);
dout.flush();
```

ObjectOutputStream knows how to convert your object in to byte array.

Text files

Write

```
byte[] textBytes = text.getBytes(Charset.forName("UTF-8"));
try (FileOutputStream stream = new FileOutputStream(textFile)) {
        stream.write(textBytes);
}
```

Read

```
List<String> lines = Files.readAllLines(Paths.get(textFile));
```