1. Difference between FileOutputStream and ObjectOutputStream

|  |  |
| --- | --- |
| FileOutputStream | ObjectOutputStream |
| It can write binary data but not object directly | I can write primitive data – type and graphs of java objects |
|  | Object -> binary |
| out.write(**"abc"**.getBytes()); | out.writeObject(**new** Person(1)); |

2. What is Serialization?

The process of writing state of an object to file or converting an object from Java form to either file supported or Network Supported form

By using FileOutputStream and ObjectOutputStream we can achive serialization.

3. What is transient keyword?

At the time of serialization if we dont want to save the value of particular variable to meet security constraints then we should go for transient keyword. And during serialization JVM ignore original value of transient variable and save default value to the file.

4. What will happen if I declare static variable as transient?

Static variable is not part of object state and hence they **won’t participate** in serialization. Due to this declaring static variable as transient is no use.

5. What will happen if I declare final variable as transient?

Final variable will participate in serialization directly by their values. Due to this declaring a final variable as transient is no use.

6. What is object graph in serialization?

Whenever we serialize an object the set of all object which are reachable from that object will be serialized automatically. This group of objects are call object graph.

In case if atleast one object is non-serializable then we will get RuntimeException saying NotSerializableException.

7. What is customized Serialization and how we can implement?

We can implement customized serialization by using

private void writeObject(ObjectOutputStream os) throws Exception

private void readObject(ObjectInputStream is) throws Exception

writeObject – will be executed automatically at the time of serialization

readObject – will be executed automatically at the time of serialization

Strictly speaking at the time of deserialization JVM will create a separate new object by executing public no-argument constructor. Hence it is compulsary to have no-argument constructor.

What is SerialVersionUID?

At the time of serialization JVM will save a unique id with every object. This unique id will be generated by JVM based .class file. At the time of deserialization receiver side JVM will compare object unique id with local .class unique id. If both are matched only then deserialization will be performed other wise receiver unable to deserialize and will get Runtime Exception saying **InvalidClassException**

In serialization both sender and receiver need not to be same and need not be from the same location and need no to use same machine.

So we can solve the above problem by cofiguring our own serialVersionUID

private static final long serialVersionUID = 1L;