# Serialization

1.How to make a Java class serializable ?

* Class should implement java.io.serializable interface.
* java.io.serializable interface has no methods, so there will be no impact on the class which implements this interface.
* We can also add special static field "serialVersionUID".

Example -

public class Person implements Serializable {

private static final long serialVersionUID = 20987689172797908L;  
private String name;  
private int age;

//Some more methods..  
}

Now this Person class can be passed from one application to the another via network or through a disk.

2.How to compute serialVersionUID?  
 All Java IDE's support mechanism to generate serialVersionUID. Below are its details,

* Its basically a hash code
* It is computed from the class name
* Interface implemented
* Methods and fields
* Using a SHA algorithim
* Please note that you can have this field (serialVersionUID) present at the compile time, i.e let say in this case you have computed it and then put it inside the class. In such case JVM will use it as it is , with no validation.

3. Tell the difference between private static final and private final ?  
Ans - "private final" should be considered as constant per object / instance. While "private static final" should be considered as constant on the class level. so "private static final" is a constaint which is available to all instances of a class.

4.Tell something about serialization and de-serialization.  
Ans - Object Serialization is a process used to convert the state of an object into a byte stream, which can be persisted into disk/file or sent over the network to any other running Java virtual machine.

The reverse process of creating an object from the byte stream is called deserialization. The byte stream created is platform independent. So, the object serialized on one platform can be deserialized on a different platform.  
  
Serializability can be enabled in your Java class by implementing the java.io.Serializable interface. It is a marker interface that means it contains no methods or fields and only serves to identify the semantics of being serializable

5. What is java.io.NotSerializableException ?  
Ans - if We Are Trying to Serialize a Non-Serializable Object, i.e if class has not implemented java.io.Serializable interface, and we are trying to serialize it .

6. What Is the serialVersionUID?  
Ans - SerialVersionUID is used for version control of an object. The consequence of not specifying serialVersionUID is that when you add or modify any field in the class, then the already-serialized class will not be able to recover because the serialVersionUID was generated for the new class and the old serialized object will be different. The Java serialization process relies on correct serialVersionUID for recovering the state of the serialized object and throws java.io.InvalidClassException in case of serialVersionUID mismatch.

7. What is transient keyword?  
Ans - The transient modifier/keyword is applicable only for variables but not for methods and classes.

At the time of serialization, if we don't want to serialize the value of a particular variable to meet security constraints, then we should declare that variable as transient.

8. Transient Vs. Static ?  
Ans - A static variable is not part of an object state, and hence, it won't participate in serialization. Due to this declaring static variable as transient, there is no use.

9. Final Vs. Transient ?  
Ans - Final variables will be participated in serialization directly by the value. Hence, declaring a final variable as transient causes no impact.