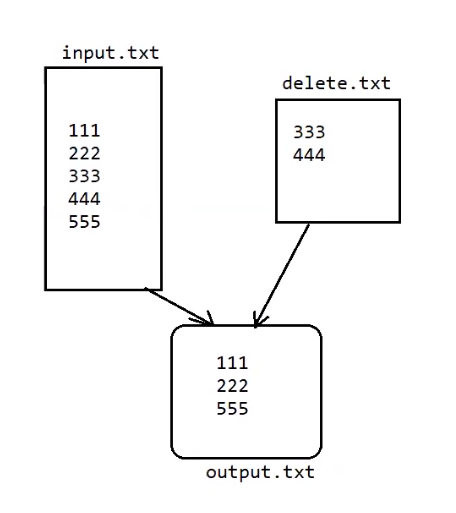
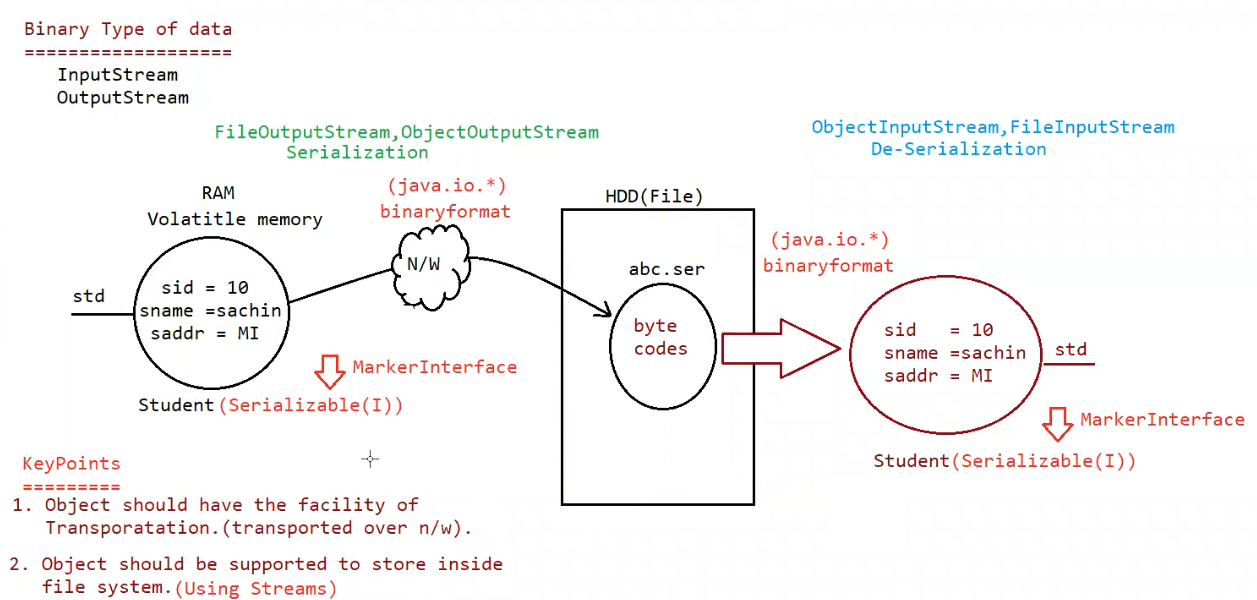
Eg: Remove\_Duplicates\_From\_2Files\_To\_Form\_Unique\_Elements\_InResult



Eg: Highest\_No\_Of\_Characters\_String\_In\_File

// go through the code



Serialization:

The process of saving (or) writing state of an object to a file is called serialization but strictly speaking it is the process of converting object from java supported form to wither network supported form (or) file supported form

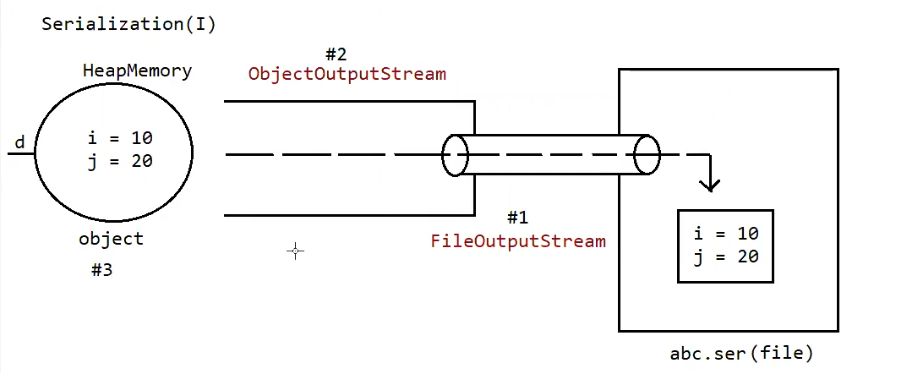
By using FileOutputStream and ObjectOutputStream classes we can achieve serialization process

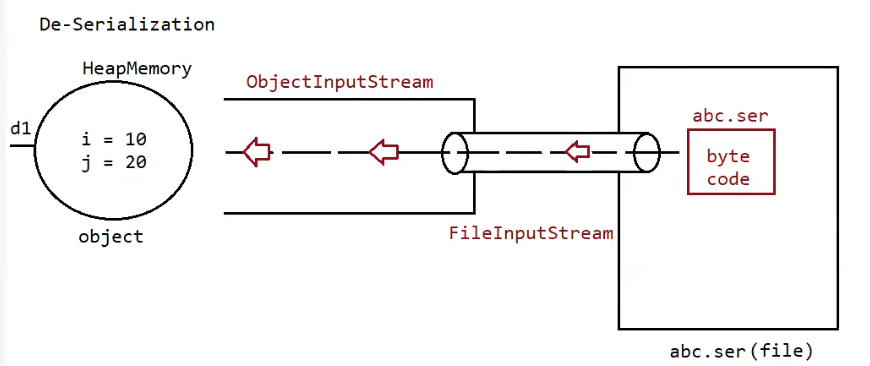
| => writeObject(Object obj)

Deserialization :

The process of reading state of an object from a file is called DeSerialization but strictly speaking it is process of converting object from file file supported form (or) network supported form to java supported form

| => readObject()





Eg: Serilization\_Deserilization

Note: State of the object ( data present inside the object)

Note:

* We can perform serialization only on serializable objects
* An object is said to be serializable if and only if the corresponding class implements serializable interface
* Serializable interface present in io package does not contain any methods. It is marker interface. The required ability will be provided automatically by jvm.
* We can add any no of objects to the file, and we can read all the objects from the file but in which order we wrote objects in the same order only the objects will come back. That order is important.
* If we are trying to serialize a non-serializable object then we will get a runtime exception saying “NotSerializableException”.

Eg: Serialization\_Deserialization\_Multiple\_Objects

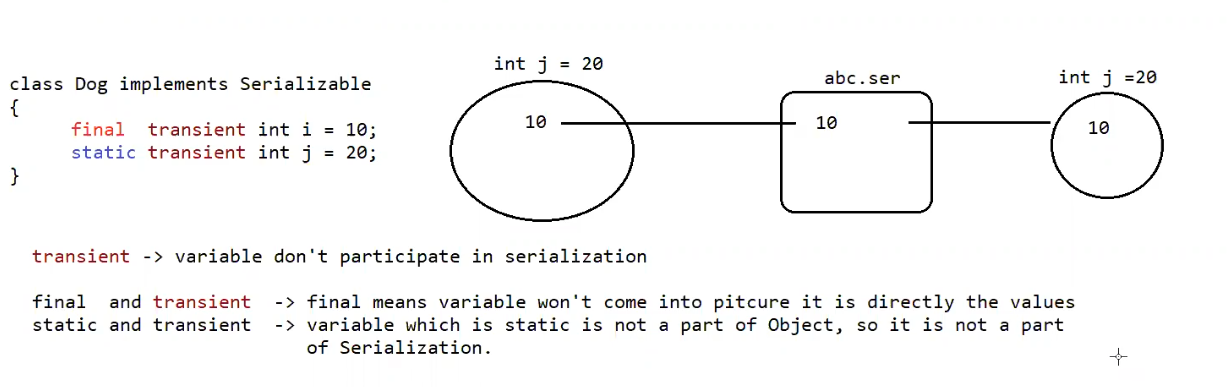
// go through the code

Eg: Serialization\_Deserialization\_Multiple\_Objects\_Different\_Order

// go through the code

Transient keyword:

* transient is the modifier applicable only for variable, but not for classes and methods
* while performing serialization if we don’t want to save the value of the particular variable to meet security constant such type of variable, then we should declare that variable with “transient” keyword
* at the time of serialization jvm ignores the original value of transient variable and save default value to the file.
* Transient means “not to serialize”.



Eg: Serialization\_Deserialization\_Transient

// go through the code

Eg: Serialization\_Deserialization\_Final

// go through the code

Eg: Serialization\_Deserialization\_Static

// go through the code

Declaration output

1. int i=10;

int j=20;

output

10- 20

1. transient int i=10;

int j=20;

output :

0 ------- 20

transient int i=10;

transient static int j=20;

output:

0-------20

4.

transient final int i=10;

transient int j=20;

output

10 ------- 0

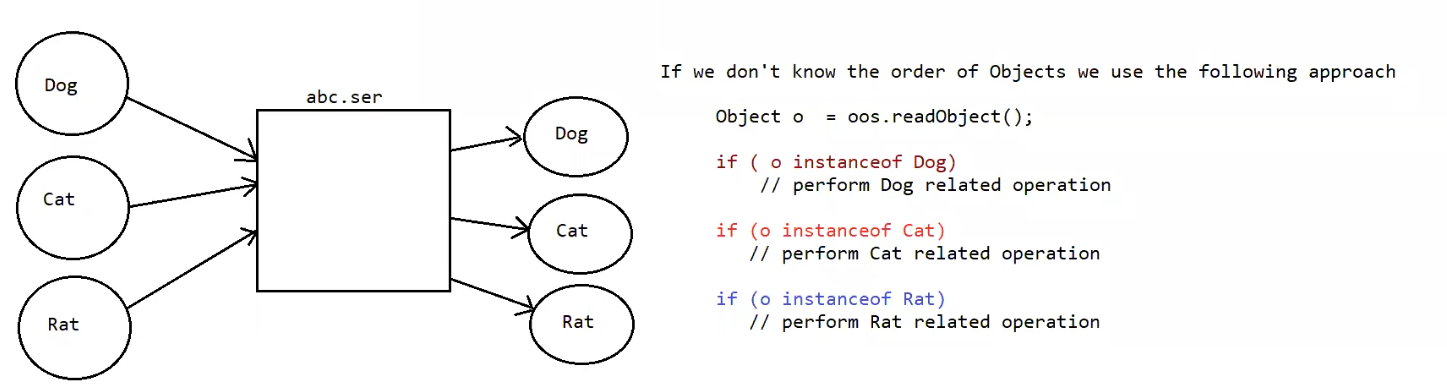
5.

transient final int i=10;

transient static int j=20;

output

10-----20

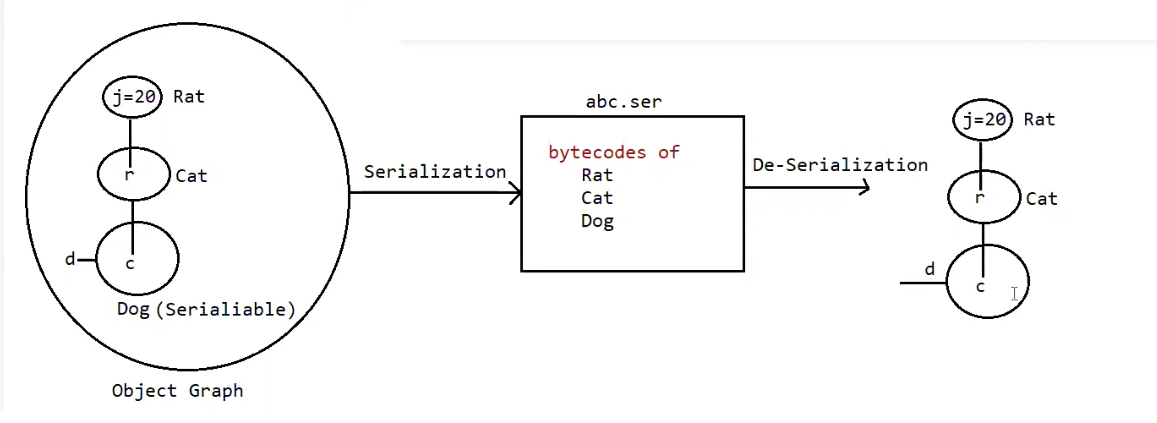


Eg: Serialization\_Deserialization\_Multiple\_Objects\_Different\_Order\_\_Using\_InstanceOf

// go through the code

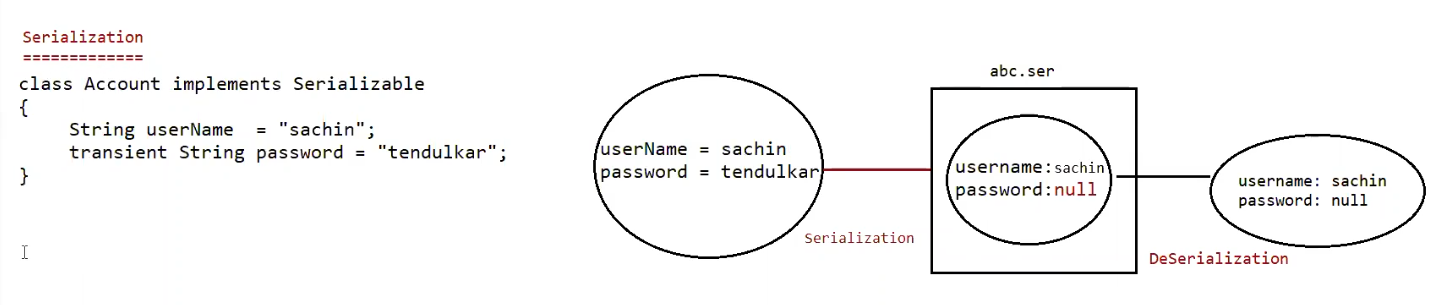
Object graph serialization:

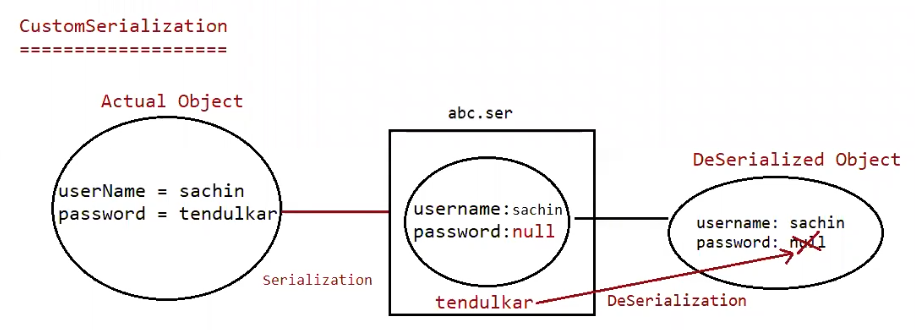
1. Whenever we are serializing an object the set of all objects which are reachable from that object will be serialized automatically. This group of objects is nothing but object graph in serialization.
2. In object graph every graph should be serializable otherwise we will get runtime exception saying “NotSeriablizableException”.

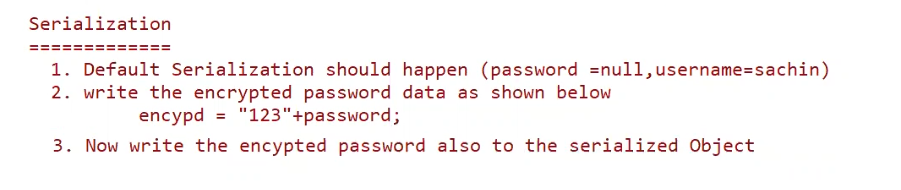


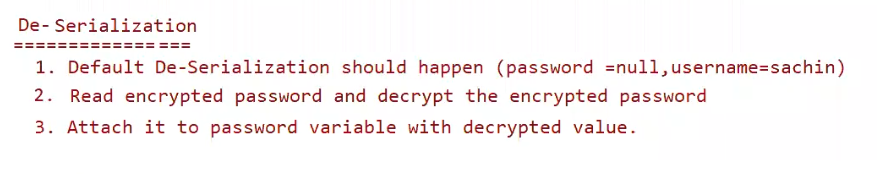
Eg: Object\_Graph\_Serialization

// go through the code









Custom-Serialization:

During default serialization there may be chance of lose of information due to transient keyword

We can implement custom serialization by using following to methods:

1. private void writeObject( ObjectOutputStream os ) throws Exception{

* this method will be executed automatically by jvm at the time of serialization.
* It is a callback method. Hence at the time of serialization if we want to perform any extra work we have to define that in this method only.

(prepare encrypted password and write the encrypted password separate to the file)

}

1. Private void readObject(ObjectInputStream is ) throws Exception{

This method will be executed automatically by jvm at the time of deserialization.

Hence at the time of deserialization if we want to perform any extra activity we have to define that in this method only.

(read encrypted password , perform decryption and assign decrypted password to the current object password variable )

Eg: Customized\_Serialization\_Transit

// go through the code

Eg: Customized\_Serialization\_Transit\_Eg1

// go through the code

Eg: Customized\_Serialization\_Transit\_Eg2

//go through the code

Serialization w.r.t inheritance

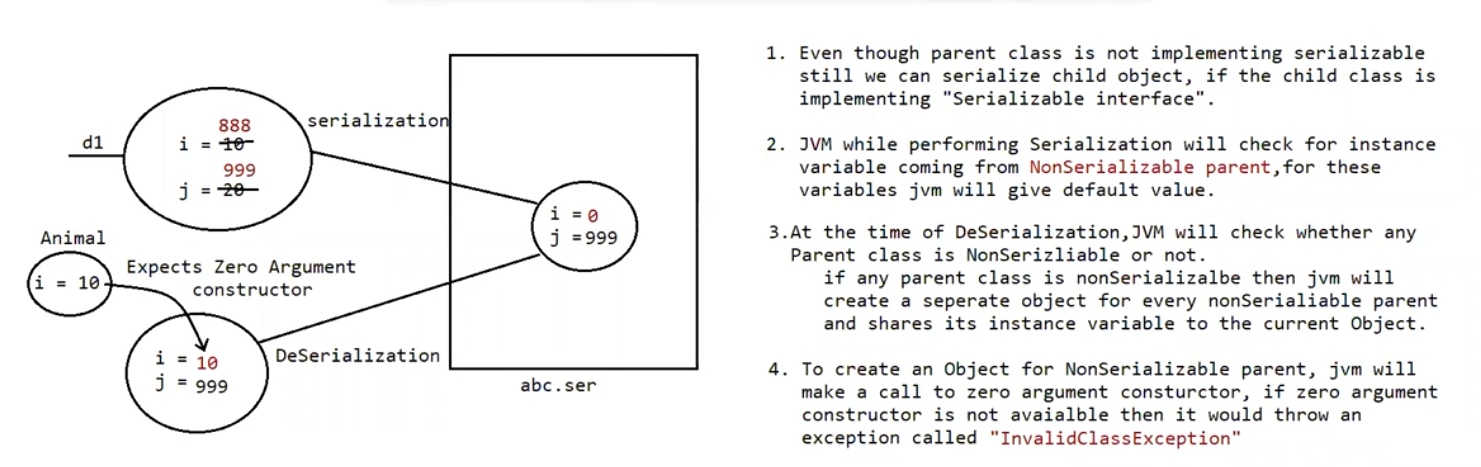
If parent class implements serializable then automatically every child class by default implements serializable.

That is serializable nature inheriting from parent to child.

Hence even though child class doesn’t implement serializable, we can serialize child class object if parent class implement serializable interface.

Eg: Serialization\_With\_Inheritance

// go through the code



Eg: Child\_Implementing\_Serializable

Case 2:

1. Even though parent class does not implements Serializable we can serialize child object if child class implements Serializable interface.

2. At the time of serialization JVM ignores the values of instance variables which are coming

from non Serializable parent then instead of original value JVM saves default values for those variables to the file.

3. At the time of Deserialization JVM checks whether any parent class is non Serializable or not.

If any parent class is nonSerializable JVM creates a separate object for every non Serializable parent and shares its instance variables to the current object.

4. To create an object for non-serializable parent JVM always calls no arg constructor

(default constructor) of that non Serializable parent hence every non Serializable parent should compulsory contain no arg constructor otherwise we will get runtime exception "InvalidClassException".

5. If non-serializable parent is abstract class then just instance control flow will be performed and share it's instance variable to the current object.

Externalization :

Externalizable interface defined 2 methods:

1. writeExternal(ObjectOutput out ) throws IoException
2. readExternal(ObjectInput in) throws IoException,ClassNotFoundException

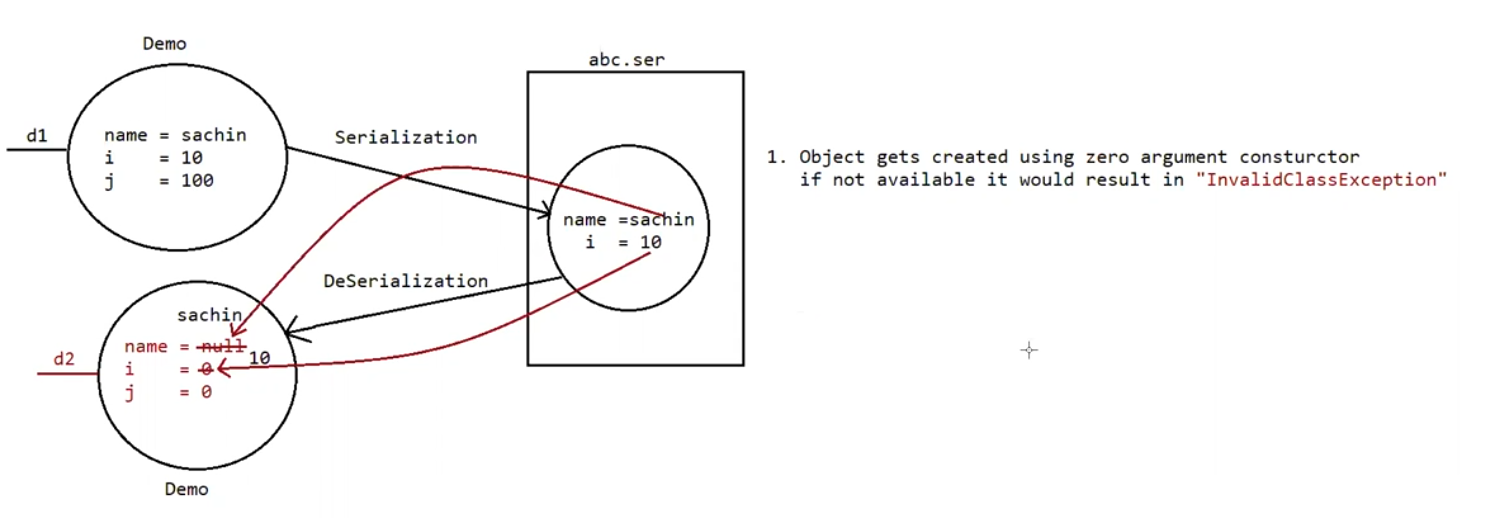
public void writeExternal(ObjectOutput out) throws IoException

This method will be executed automatically at the time of serialization, within this method we have to write the code to save required variables of the file .

public void readExternal(ObjectInput in) throws IoException, ClassNotfoundException

this method will be executed automatically at the time of deserialization, within this method we have to write the code to save the required variable from file and assign to current object.

Eg: Externalization\_1



At the time of deserialization jvm will create a separate new object by executing public no-arg constructor on that object jvm will call readExternal() method

Every Externalizable class should compulsory contain public no-argument constructor, otherwise we will get RuntimeException saying “InvalidClassException”.

Serialization vs Externalization :

Serialization

1. It is meant for default Serialization

2. Here every thing takes care by JVM and programmer doesn't have any control doesn't have any control

3. Here total object will be saved always and it is not possible to save part of the object.

4. Serialization is the best choice if we want to save total object to the file.

5. relatively performence is low.

6. Serializable interface doesn't contain any method

7. It is a marker interface.

8. Serializable class does not required to contains public no-arg constructor.

9. transient keyword play role in serialization

Externalization

1. It is meant for Customized Serialization

2. Here every thing takes care by programmer and JVM does not have any control.

3. Here based on our requirement we can save either total object or part of the object.

4. Externalization is the best choice if we want to save part of the object.

5. relatively performence is high

6. Externalizable interface contains 2 methods :

writeExternal()

readExternal()

7. It is not a marker interface.

8. Externalizable class should compulsory contains public no-arg constructor otherwise we will get RuntimeException saying "InvalidClassException"

9. transient keyword don't play any role in Externalization.