Serialization/deserialization in Java

Serialization in java is a mechanism of writing the state of an object into a byte stream. It is mainly used in Hibernate, RMI, JPA, EJB, JMS technologies. The reverse operation of serialization is called deserialization. The String class and all the wrapper classes implements java.io.Serializable interface by default.

* Transient Keyword in Serialization
* Transient Static variable in Serialization
* Transient Final variable in Serialization

Problem is:

Example

If I want to send a biggest balloon to another city by courier in same condition. They firstly reject to do this, then they think over it, they find a opener (hole) over it, and said yes we can send it. By removing air from balloon we can send it, and in second city we can fill balloon with air again.

So we have to convert out data from normal to transportable form, ------🡪[Serialization]

Then again from transport supported form to original form ------🡪 [DeSerialization]

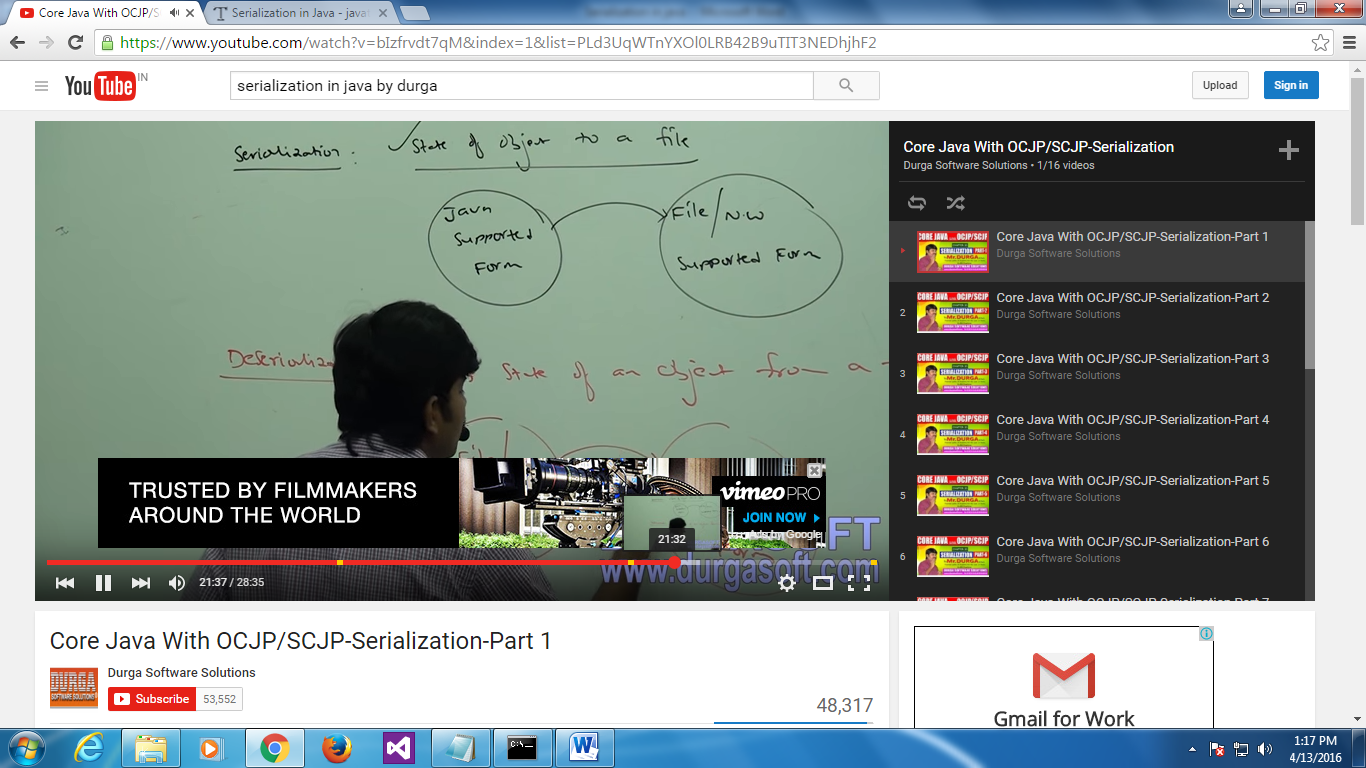
Now mapped this example to java supported form

Process of converting Java supported form to File supported form/Network supported form is consider as Serialization,

Process of reading object [state of an object] form File supported form/Network supported form is considered as DeSerialization

Process of converting

File supported form---------------------🡪Java supported form



How to do this:

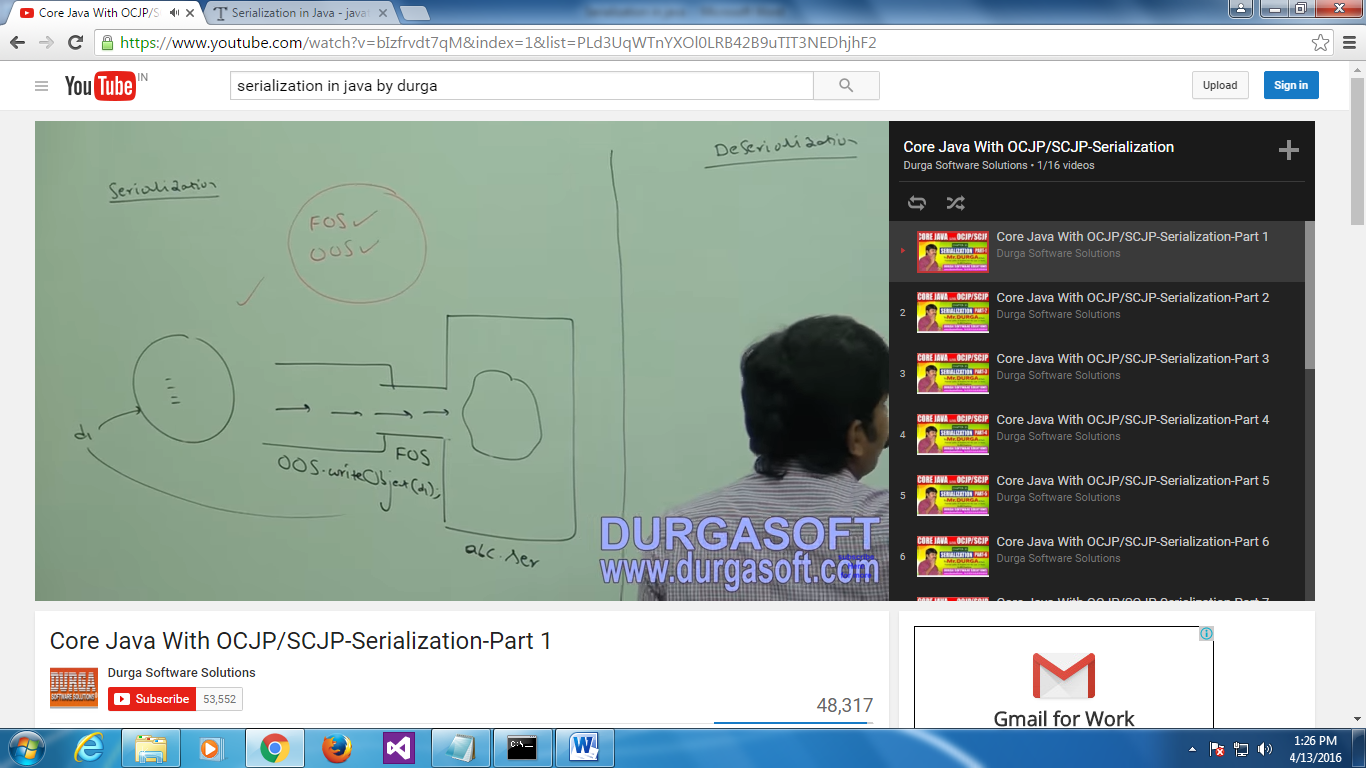
Step 1 :When we are writtig object [that is binary data] we require FileOutputStream, so will create a file output stream.

We can write binary data but not object directly, for this purpose we need ObjectOutputStream.

So to send directly object we have ObjectOutputStream.

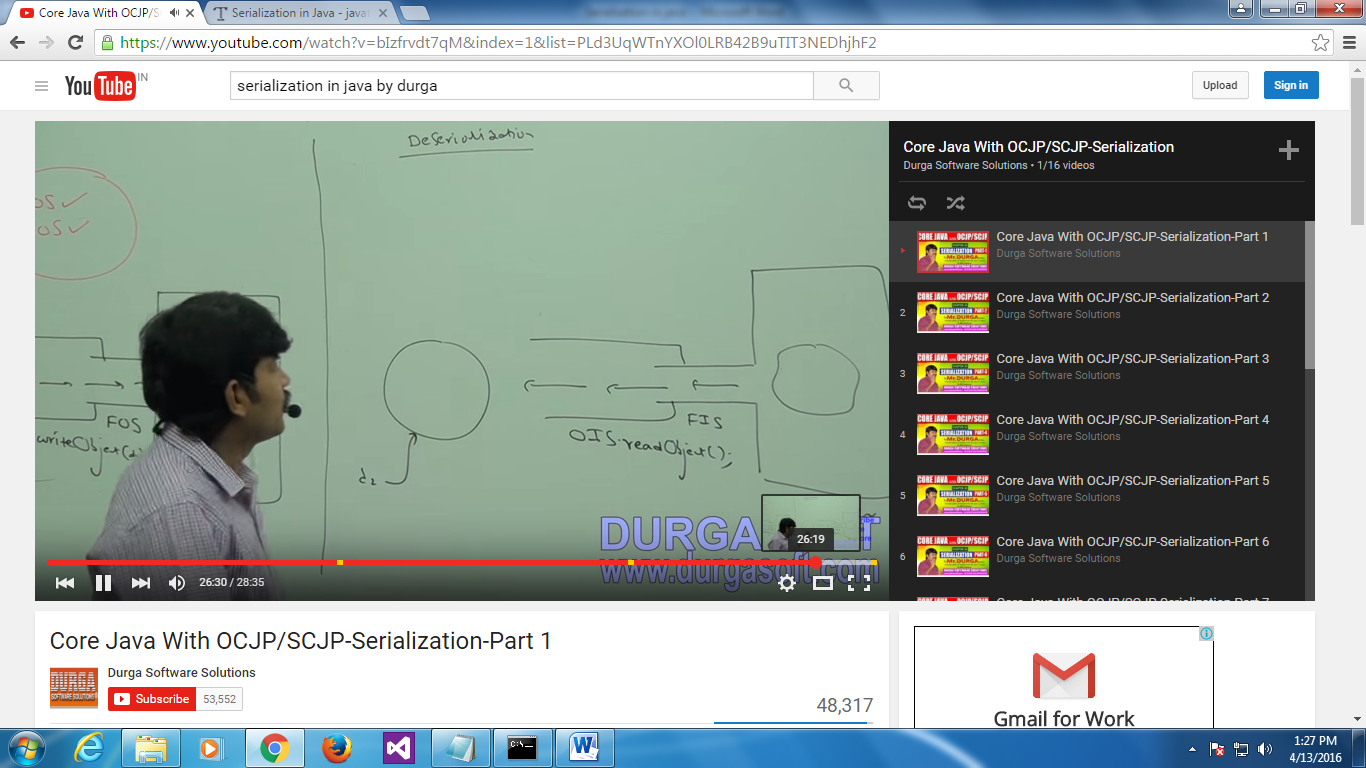
So first open FileOutputStream, then open ObjectOutputStream.

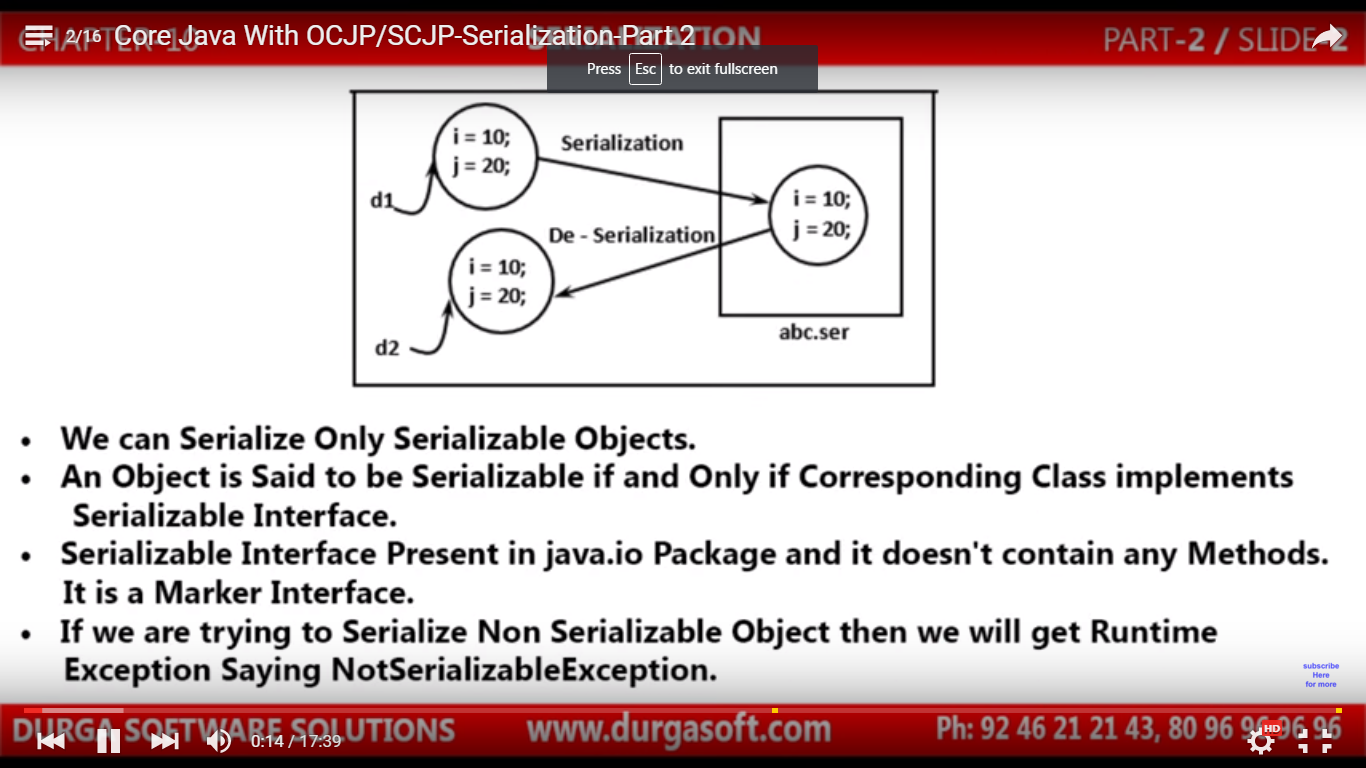
ObjectOutputStream have a method with name writeObject pass class object in this method.



Same for DeSerialization.

FileInputStream, ObjectInputStream





import java.io.\*;

class Dog implements Serializable

// if we do not make dog class Serializable we get run time error

{

int a=10,b=20;

}

class Test{

public static void main(String args[]) throws Exception

{

// by using throws I don’t need to handle exception, jvm automatically handle it.

Dog d1=new Dog();

FileOutputStreamfos=new FileOutputStream("abc.ser");

ObjectOutputStreamOos=new ObjectOutputStream(fos);

Oos.writeObject(d1);

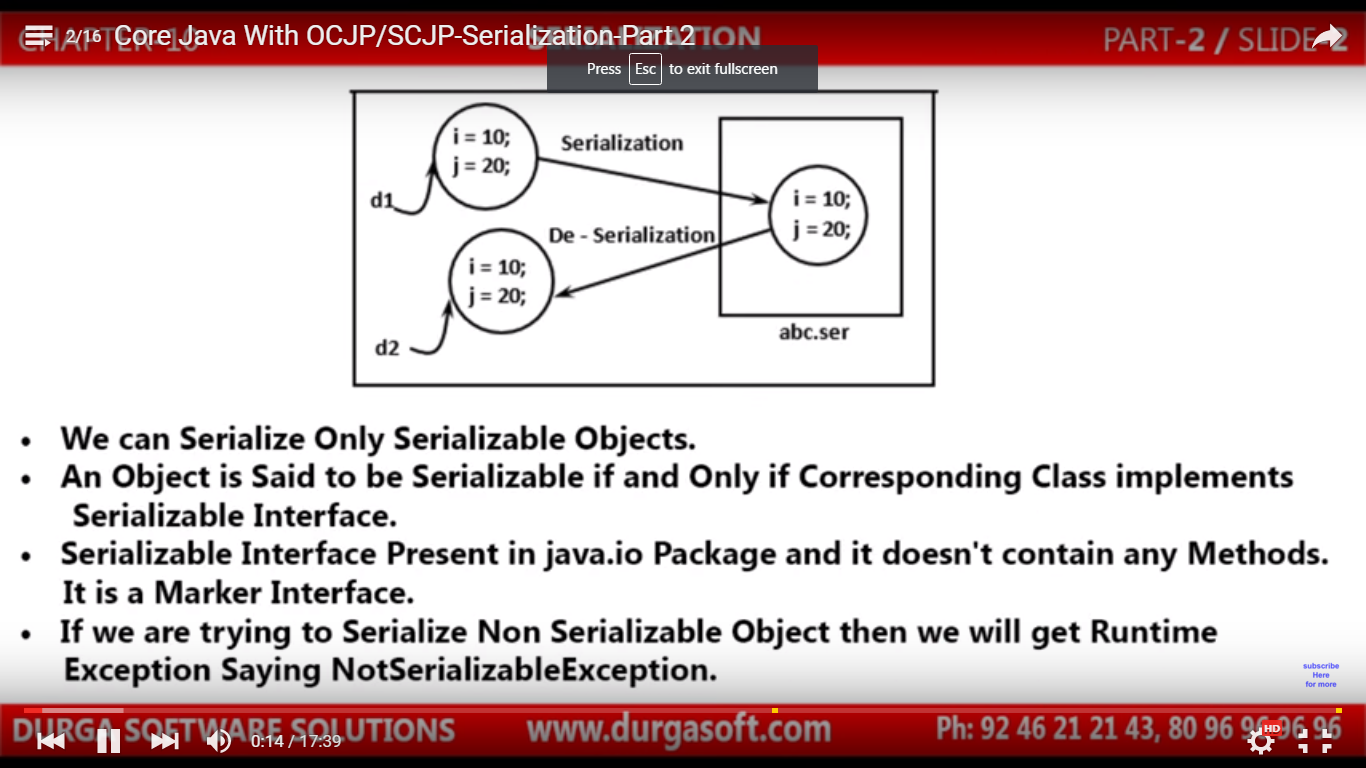
FileInputStream fin=new FileInputStream("abc.ser");

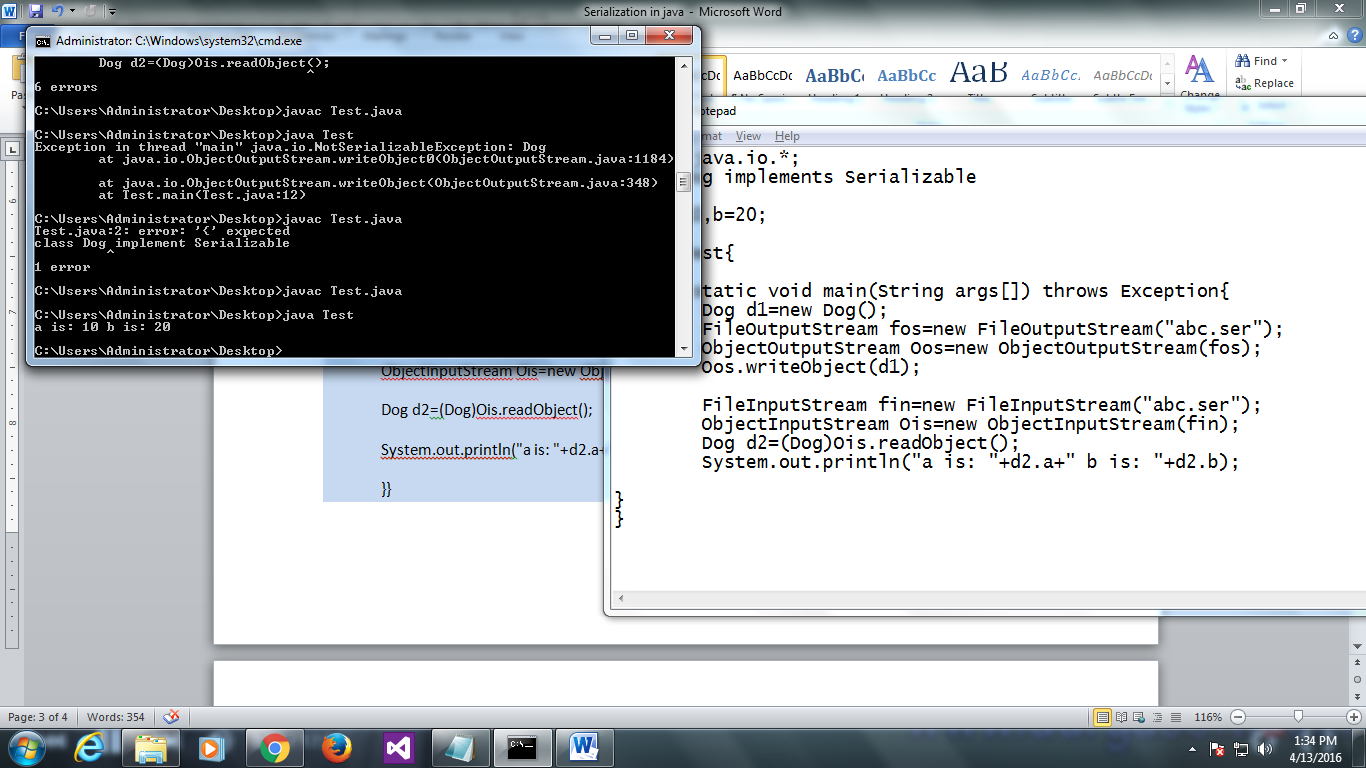
ObjectInputStreamOis=new ObjectInputStream(fin);

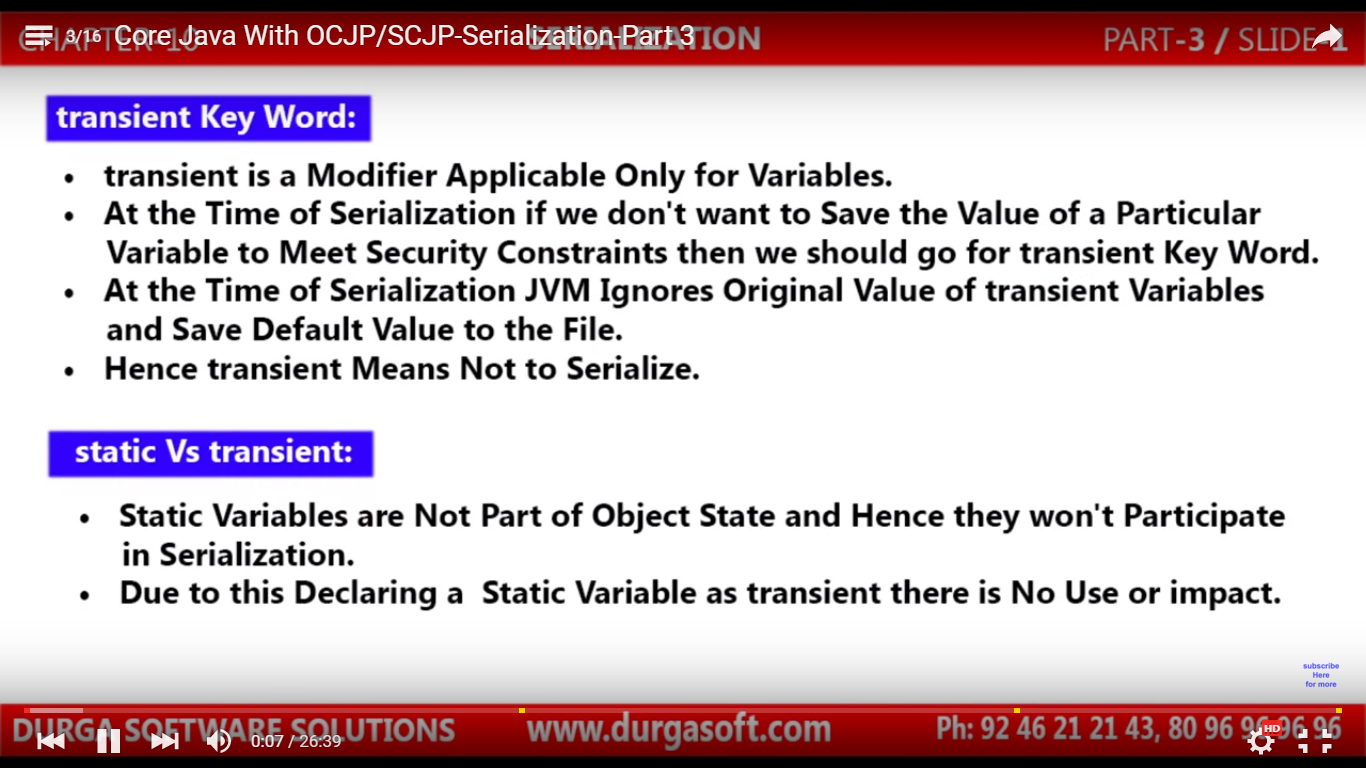
Dog d2=(Dog)Ois.readObject();

System.out.println("a is: "+d2.a+" b is: "+d2.b);

}}







Example for Transient:

Is I want to save username we can use normal variable, but for password we can not use normal variables, for this purpose we will use transient, during serialization.

If I declare any variable as transient, JVM ignores original value of that variable, and saves the default value

Values Stored in file

Int i=10, -------------🡪 i=10;

Transient int j=20; serialization j=0//because j is int type

Example:

import java.io.\*;

class Dog implements Serializable// if we do not make dog class Serializable we get run time error

{

int a=10;

transient int b=20;

}

class Test{

public static void main(String args[]) throws Exception{//throws by using I dnt need to handle exception //jvm automatically handle it.

Dog d1=new Dog();

FileOutputStreamfos=new FileOutputStream("abc.ser");

ObjectOutputStreamOos=new ObjectOutputStream(fos);

Oos.writeObject(d1);

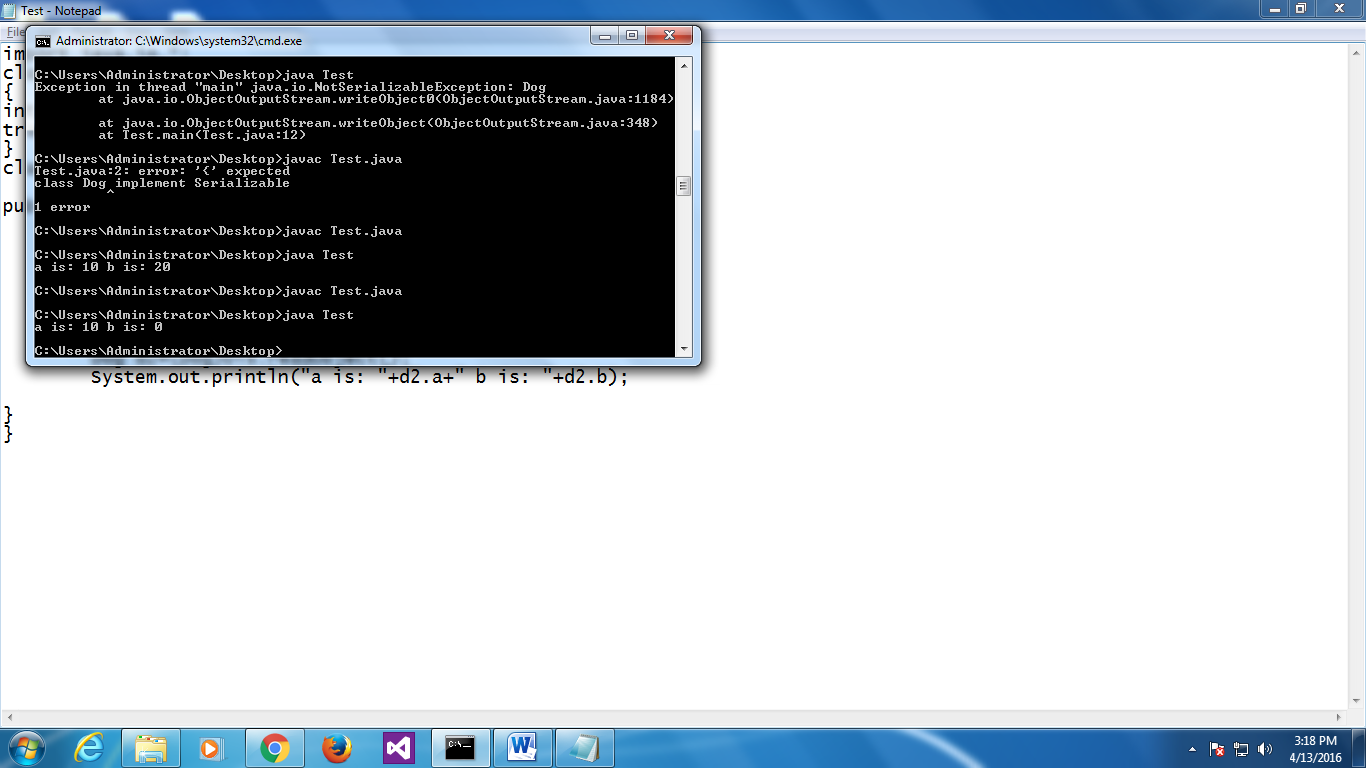
FileInputStream fin=new FileInputStream("abc.ser");

ObjectInputStreamOis=new ObjectInputStream(fin);

Dog d2=(Dog)Ois.readObject();

System.out.println("a is: "+d2.a+" b is: "+d2.b);

}}



import java.io.\*;

classPersistExample{

public static void main(String args[])throws Exception{

Student s1 =new Student(211,"ravi",22);//creating object

//writing object into file

FileOutputStream f=new FileOutputStream("f.txt");

ObjectOutputStream out=new ObjectOutputStream(f);

out.writeObject(s1);

out.flush();

out.close();

f.close();

System.out.println("success");

}

}

Output:

success

