
Serialization and Deserialization
©2015 Manipal Global Education

manipalglobal
education services

LEARNING OBJECTIVES

At the end of this lesson, you will be able to:

- Understand the concept of Serialization and Deserialization
- Use Classes and Interface for Serialization/Deserialization
- Understand the use of transient keyword
- Understand object graph in Serialization
- Understand Inheritance in Serialization

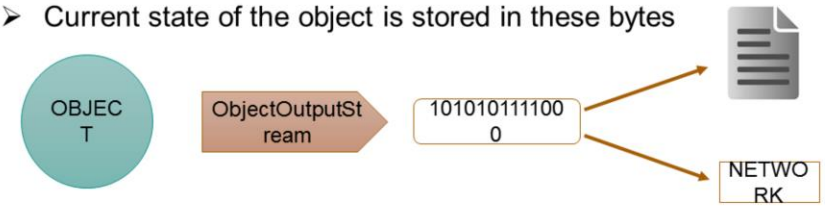
Refer package **com.mgait.io.serialization** in the provided code base for demo programs on the topics covered in this presentation


manipalglobal
education services
INTRODUCTION
Serialization and
Deserialization
4
©2015 Manipal Global Education

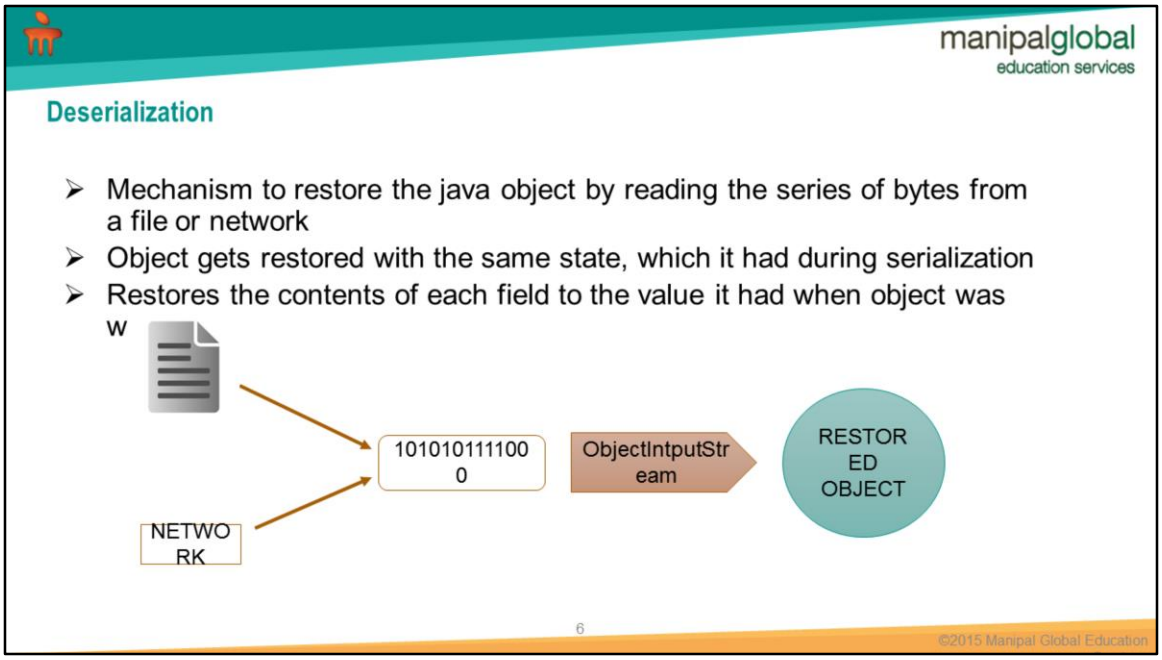
manipalglobal
education services

Serialization

- Mechanism to transform a Java object in to a sequence of bytes
- This sequence of bytes can then be saved in a File or transmitted across a network
- Current state of the object is stored in these bytes



- Need for Serialization
  - Save the object and use it at a future time Ex. Session
  - Transmit the object over a network to other JVM


manipalglobal
education services
Deserialization
Mechanism to restore the java object by reading the series of bytes from a file or network
Object gets restored with the same state, which it had during serialization
Restores the contents of each field to the value it had when object was w
101010111100
0
ObjectIntputStr
eam
RESTOR
ED
OBJECT
NETWO
RK
6
©2015 Manipal Global Education

Serializable Interface

- Not all objects can be serialized
- Only objects of classes that implement **java.io.Serializable** can be serialized
- Serializable is a marker interface and hence no methods have to be implemented

```
public class Employee implements Serializable{
      //...
}
```

- **NotSerializableException** is thrown, if object of a class which does not implement Serializable is Serialized

manipalglobal
education services

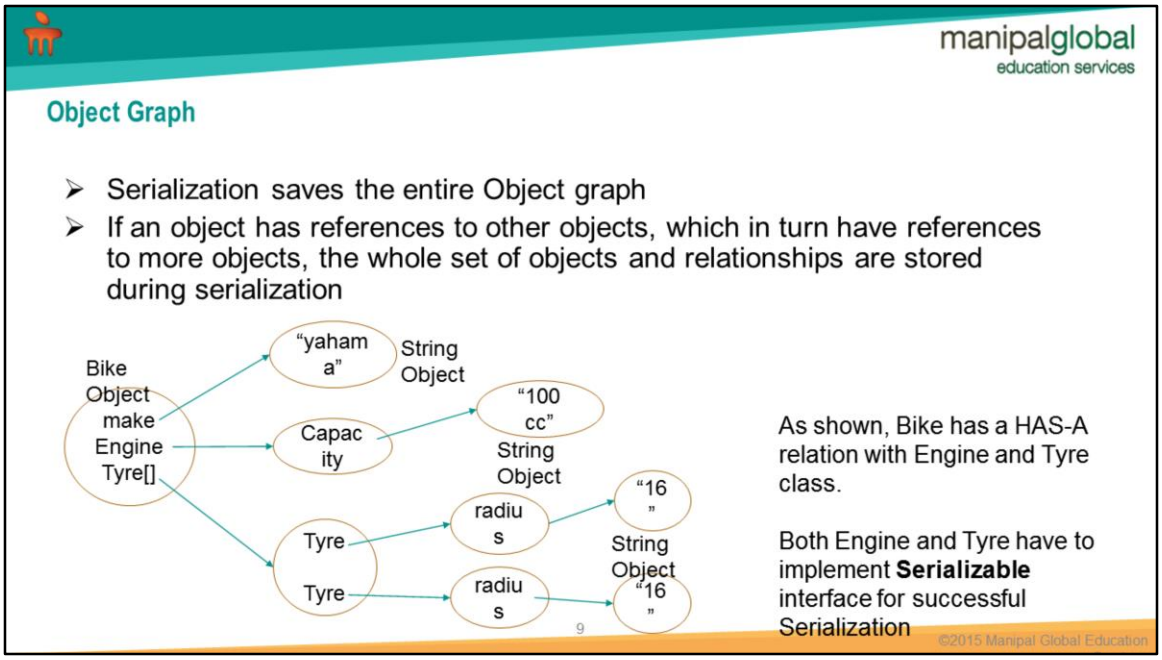
ObjectOutputStream for Serialization

- ObjectOutputStream
  - Is the class used to write objects to an OutputStream i.e. transform the object in to a sequence of bytes
  - Needs another class to write the sequence of bytes to a destination
    - For ex. To write the object to a File, a FileOutputStream is needed
  - writeObject(..) method of ObjectOutputStream does the serialization

```
FileOutputStream file = new FileOutputStream("employee.data");
ObjectOutputStream out = new ObjectOutputStream(file);

Employee emp  = new Employee();
out.writeObject(emp);
```

- ObjectOutputStream has methods to write primitive data to an OutputStream
  - ex. writeInt(..), writeFloat(..), writeBoolean(..), etc


manipalglobal
education services
Object Graph
Serialization saves the entire Object graph
If an object has references to other objects, which in turn have references to more objects, the whole set of objects and relationships are stored during serialization
Bike Object
make
Engine
Tyre[]
"yaham a"
String Object
Capac ity
"100 cc"
String Object
Tyre
Tyre
radiu s
radiu s
"16 "
String Object
"16 "
As shown, Bike has a HAS-A relation with Engine and Tyre class.
Both Engine and Tyre have to implement Serializable interface for successful Serialization
9
©2015 Manipal Global Education

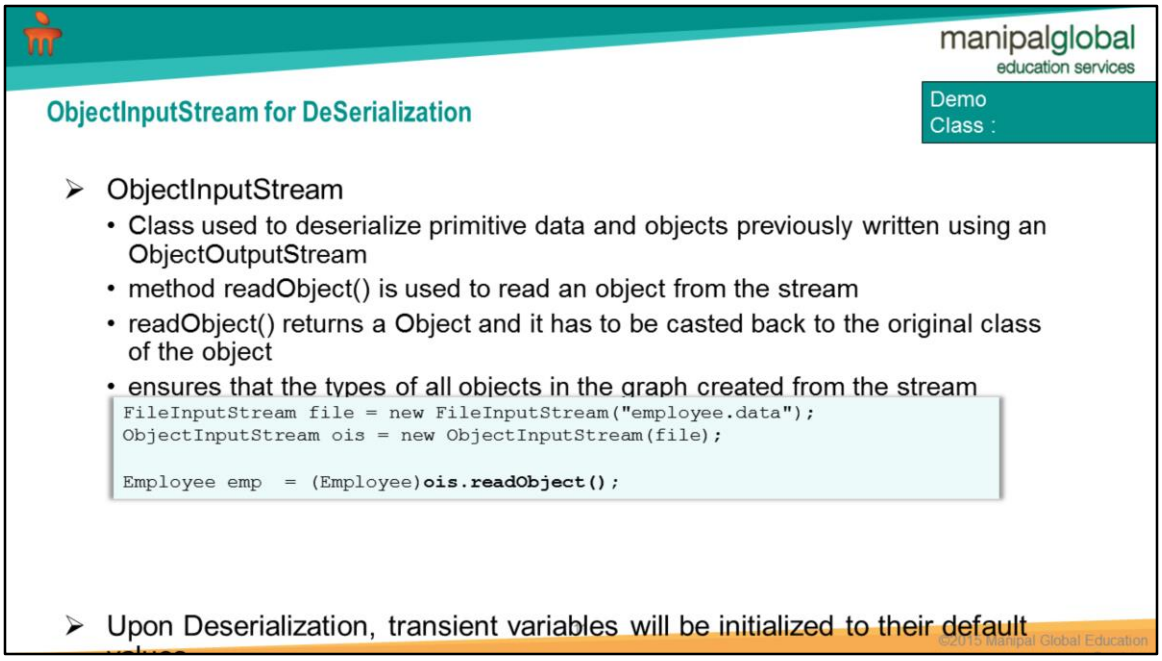
manipalglobal
education services

Demo
Class :

Transient Variable

- During Serialization, values for all the instance variable are stored by default
- You cannot serialize a Bike object, if its Engine instance variable refuses to be serialized (by not implementing Serializable)
- If an instance variable has to be skipped by the Serialization process, then it has to be marked **transient**

```
public class Bike implements Serializable{
       private transient String make;
       private transient Engine eng = new Engine();
       private Tyre[] tyres = new tyre[2];
}
```


manipalglobal
education services
ObjectInputStream for DeSerialization
Demo
Class :
ObjectInputStream
Class used to deserialize primitive data and objects previously written using an ObjectOutputStream
method readObject() is used to read an object from the stream
readObject() returns a Object and it has to be casted back to the original class of the object
ensures that the types of all objects in the graph created from the stream
FileInputStream file = new FileInputStream("employee.data");
ObjectInputStream ois = new ObjectInputStream(file);
Employee emp = (Employee)ois.readObject();
Upon Deserialization, transient variables will be initialized to their default
©2015 Manipal Global Education

manipalglobal
education services

Inheritance in Serialization

Demo
Class :
SaveEmployee

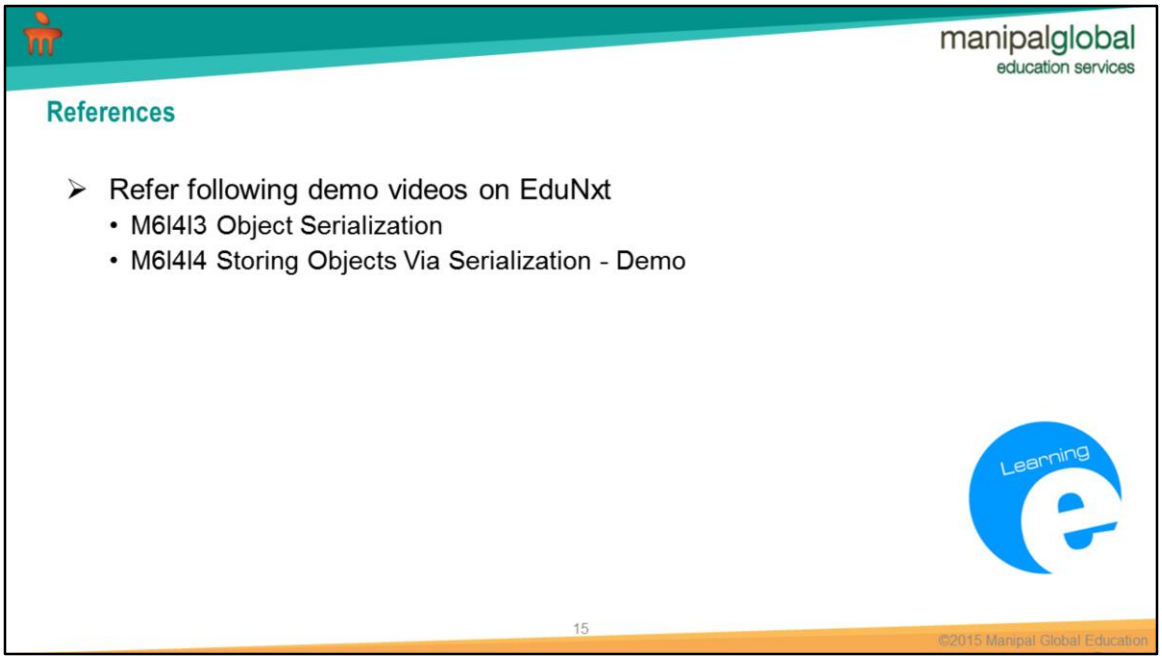
- If a class is Serializable, all of its subclasses become Serializable
- Sub classes can be serializable, even if super class is not Serializable (i.e. sub class implements Serializable and Super class does not)
  - In this case, the superclass must have a no-arg constructor to allow its fields to be initialized
  - InvalidClassException is thrown during deserialization, if no-arg constructor is not provided in super class

Points to remember

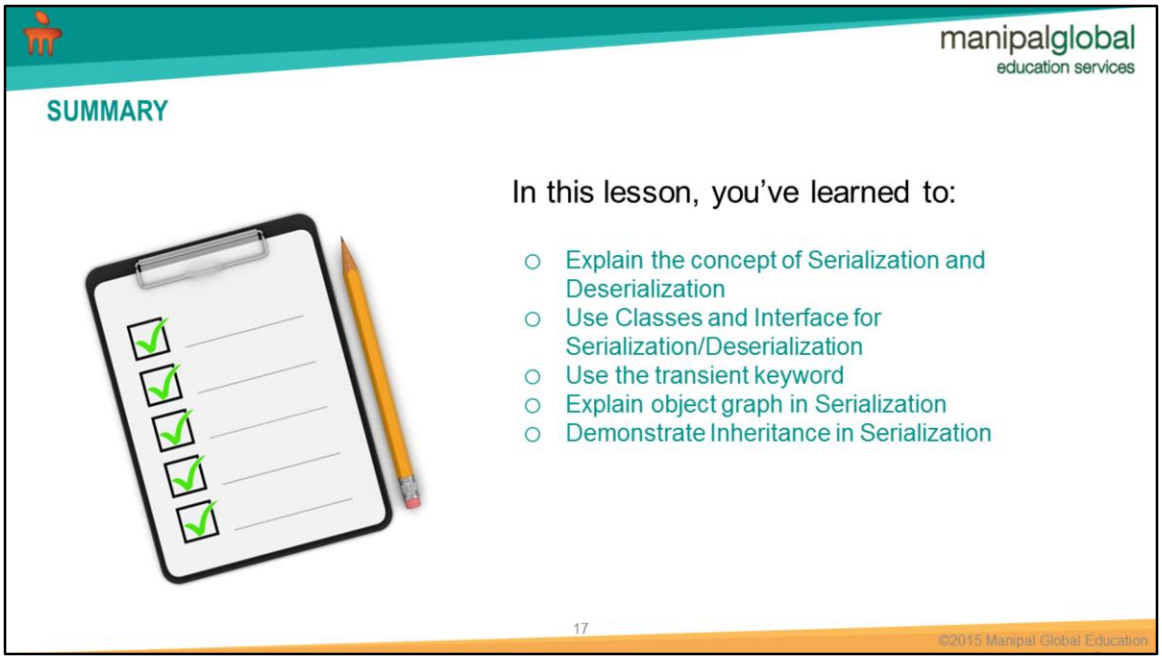
- An object can be serialized only if its class implements Serializable
- NotSerializableException is thrown, if an object is serialized, whose class does not implement serializable
- Whole graph of the object is saved during serialization
- Transient variables and static variables are not saved during serialization
- During deserialization, transient and static variables are assigned their default/initialized values

Points to remember

- During deserialization, the constructor of serializable objects are not run
- If a class has instance variables which is an object(HAS-A), then class of this object should implement serializable
- If super class implements serializable, all objects of sub class also become serializable
- If super class is not serializable and sub class is, then super class should have a default constructor, which is executed to initialize the object


manipalglobal
education services
References
Refer following demo videos on EduNxt
M6I4I3 Object Serialization
M6I4I4 Storing Objects Via Serialization - Demo
Learning
15
©2015 Manipal Global Education


manipalglobal
education services
SUMMARY
Serialization and
Deserialization
16
©2015 Manipal Global Education


manipalglobal
education services
SUMMARY
In this lesson, you've learned to:
Explain the concept of Serialization and Deserialization
Use Classes and Interface for Serialization/Deserialization
Use the transient keyword
Explain object graph in Serialization
Demonstrate Inheritance in Serialization
17
©2015 Manipal Global Education