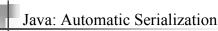


address.txt

Serialization

- Java's answer:
 - Serialization
 - Object know how to read/write themselves to streams
- Problem- Objects have state in memory
- Serialization is also called
 - Flattening, Streaming, Dehydrate (rehydrate = read), Archiving



- Serializable Interface
 - By implementing this interface a class declares that it is willing to be read/written by automatic serialization machinery
 - Found in java.io package
 - Tagging interface has no methods and serves only to identify the semantics of being serializable
- Automatic Writing
 - System knows how to recursively write out the state of an object to stream
 - · Recursively follows references and writes out those objects too!
- Automatic Reading
 - System knows how to read the data from Stream and re-create object in memory
 - Downcasting is required



How it works?

- To write out an object of PersonInfo
 - PersonInfo p = new PersonInfo();
 - ObjectOutputStream out;
 - out.writeObject(p)
- To read that object back in
 - ObjectInputStream in;
 - PersonInfo obj = (PersonInfo) in.readObject();
- Must be of the same type
 - class and version issue



Example Code: Serialization

Reading/Writing PersonInfo objects



25-Oct-0

```
Example Code: Serialization

import javax.swing.*;
import java.io.*;

public class PersonInfo implements Serializable {

String name;
String address;
String phoneNum;

public void printPersonInfo() {

JOptionPane.showMessageDialog("name:"+ name + "address:" + address + "phoneNum:" + phoneNum);
}
```

```
Example Code: Serialization (cont.)

import java.io*;
public class WriteEx{
public static void main(String args[]){

PersonInfo pWrite = new PersonInfo("ali", "defence", "9201211");

try {

FileOutputStream fos = new FileOutputStream("ali.dat");

ObjectOutputStream out = new ObjectOutputStream(fos);

//serialization
out.writeObject(pWrite);
out.close();
fos.close();
} catch (Exception ex){

System.out.println(ex)
}
```

```
Example Code: Serialization (cont.)

import java.io*;
public class ReadEx{
public static void main(String args[]){

try {

FileInputStream fis = new FileInputStream("ali.dat");
ObjectInputStream in = new ObjectInputStream(fis);

//de - serialization
PersonInfo pRead = (PersonInfo) in.readObject();
pRead.printPersonInfo();
in.close();
fis.close();
} catch (Exception ex){
System.out.println(ex)
}
```

Object Serialization and Network

- You can read/write objects to network using sockets
- The class version should be same on both sides (client & Server) of the network

10

Sending Objects over Network

- PersonInfo p = new PersonInfo ("ali", "defence", "9201211");
- Socket s = new Socket("localhost", 4444);
- OutputStream os = s.getOutputStream();
- ObjectOutputStream oos = new ObjectOutputStream(os);
- oos.write(p);

4,7,

Reading Objects from Network

- Socket s = ss.accept();
- InputStream in = s.getInputStream();
- ObjectInputStream ois = new ObjectInputStream(is);
- PersonInfo p = (PersonInfo) ois.read();

.....

12

Preventing Serailization

- transient keyword is used to mark a field that should not be serialized
- Often there is no need to serialize sockets, streams & DB connections etc (they do not represent the state of object, rather connections to external resources)
- So, we can mark them as
 - public transient Socket s;
 - public transient OutputStream os;
 - public transient Connection con;
- Transient fields are returned as null on reading

import javax.swing.*;
import javax.swing.*;
import java.io.*;

public class PersonInfo implements Serializable {

String name;
String address;
transient String phoneNum;

public void printPersonInfo() {

JOptionPane.showMessageDialog("name: " + name + "address:" + address + "phoneNum: " + phoneNum);

}

C

Circularity: not an issue

 Serialization machinery will take circular references into account and do the right thing!

15