Desarrollo de Aplicaciones en Red

José Rafael Rojano Cáceres
http://www.uv.mx/rrojano

Java Review Concepts

Exercises

- *Create an application that sends data over Internet to another application. The second application show the message. Use TCP protocol for establishment the communication.*

Streams

- *Streams are included in the package io*
- *Streams represent a ordered bytes sequence*
- *For using streams we must use InputStream and OutputStream libraries*
- *A stream represent the source or origin from which an object can read or write bytes.*
- *Streams are used for files or network connections*
- *InputStream: used for fetching data*
  - `read, read(byte[] buffer), read(byte[] buffer, int startOffset, int numBytes)`
- *OutputStream: used for sending data*
  - `write, write(int numBytes)`

InputStream

- *InputStream let's read a set of bytes from an object.*
- *In general InputStream let's write a byte, but if it's necessary ro read a chunk of byte you should use the interface DataInputStream which offer the methods:*
  - **readByte, readBoolean, readChar,…, readUTF**

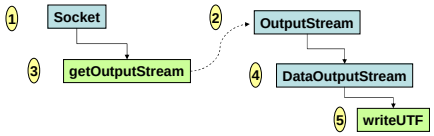
OutputStream

- *As well as InputStream, OutputStream let's write bytes.*
- *In order to get some filter bytes should be used the class DataOutputStream which has the methods:*
  - **writeChars, writeByte, writeInt, … the rest of primitive and, writeUTF**
- *writeUTF writes stringswith UTF format.*

Filter streams

- ***The filter streams let to programmer to feed different streams with the constructor from another stream.***
- ***This action let's create more interesting and useful data types.***
- ***This action is made through by DataXxxxStream y PrintWriter.***

Sending data

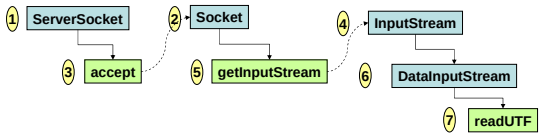


1. Create a socket
2. Create a new OutputStream
3. Associate the output stream socket with the object OutputStream
4. Create a new DataOutputStream in order to write data to a stream (this is like a presentation layer)
5. With the object DataOutputStream use the method writeUTF to send the information

Client

```
/*
        Application that communicates with other sending a message
*/
import java.io.*;  // import stream as medium of write data
import java.net.*; // import package to communicate over Internet
public class client01
{
        static String nameServer = "localhost";
        static int port = 2000;
        public static void main(String[] arg)
        {
                if (arg.length == 0)
                        System.out.println("Using default: Server='localhost' and
port='2000'");
                else
                {
                        nameServer = arg[0];
                        port = Integer.parseInt(arg[1]);
                }
                Socket socketClt = null;
                try
                {
                        // client use socket to send establish communication
                        socketClt = new Socket(nameServer, port);
                        // defines ouput stream ASSOCIATING WITH socket
                        OutputStream stream = socketClt.getOutputStream();
                        // makes a presentation of data at the stream
                        DataOutputStream mesg = new DataOutputStream(stream);
                        // send information throught stream
                        mesg.writeUTF("Hola amigo");
                        socketClt.close();
                }
                catch (IOException e)
                {
                        e.printStackTrace();
                }
        }
}
```


Receiving Data



1. Creates a Serversocket to listen in an specific port
2. Creates a new Socket client and,
3. Associates with the canal through accept
4. Creates a new InputStream and,
5. Associate with the Socket through getInputStream
6. Create a new DataOutputStream in order to read data to a stream (this is like a presentation layer for decodification)
7. With the object DataOutputStream use the method ReadUTF to send the information

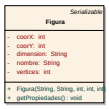
Server

```
/*
        Application that read information from another source
*/
import java.io.*;  // import stream as medium of write data
import java.net.*; // import package to communicate over Internet
public class server01 {
        // Define default por for listening
        static int port = 2000;
        public static void main(String[] arg){
                if (arg.length > 1)
                        port = Integer.parseInt(arg[0]);
                try {
                        // Creates services for attending
                        ServerSocket socketSrv = new ServerSocket (port);
                        // wait for request undefinely
                        while (true) {
                          // create a new socket and ass ASSOCIATE with a new server
                          // that waits for connection
                          Socket socketClt = socketSrv.accept();
                          // Defines a new stream for reading ASSOCIATING WITH the
                          // socket
                          InputStream stream = socketClt.getInputStream();
                          // Makes the presentation of data in stream
                          DataInputStream mesg = new DataInputStream(stream);
                          System.out.println("Message from remote client: " +
mesg.readUTF());
                          socketClt.close();
                        }
                }
                catch (IOException e){
                        e.printStackTrace();
                }
        }
}
```


Serialization

Exercise

- ***Define the class called Figura according to the figure showed that implements the class Serializable***



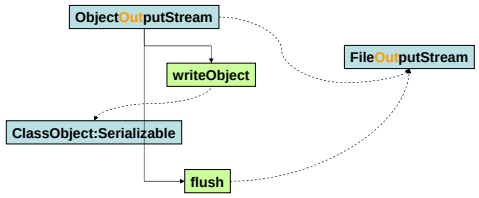
Class Figura

```
import java.io.*;

public class Figura implements Serializable
{
        private String dimension;
        private int vertices;
        private String nombre;
        private int coorX, coorY;

        public Figura(String nom, String dim, int ver, int x, int y)
        {
                nombre = nom; dimension = dim; vertices = ver;
                coorX = x; coorY = y;
        }

        public void getPropiedades()
        {
                System.out.println("Nom:" + nombre);
                System.out.println("Dim:" + dimension);
                System.out.println("Ver:" + vertices);
                System.out.println("[" + coorX + "," + coorY + "]");
        }
}
```


Exercise

- ***Create the class which handle the serialization and deserialization according to:***

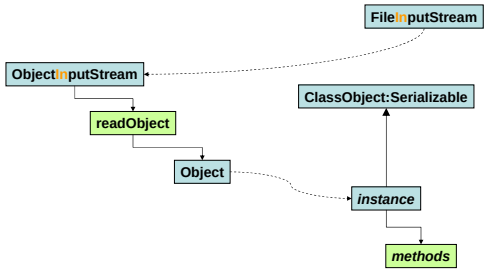
Writing the object



Serializing an object

```
import java.io.*;
public class SerializaFigura
{
        public static void main(String arg[])
        {
            FileOutputStream archivo = null;
            ObjectOutputStream salida = null;
          Figura f1 = new Figura("cubo", "3d", 8, 4, 5);
          Figura f2 = new Figura("piramide", "3d", 5, 12, 6);
          Figura f3 = new Figura("esfera", "3d", 0, 9, 20);
                try
                {
                  archivo = new FileOutputStream("figuras.xyz");
                  salida = new ObjectOutputStream(archivo);
                  salida.writeObject(f1);
                  salida.writeObject(f2);
                  salida.writeObject(f3);
                  salida.flush();
                }
                catch (IOException e)
                {
                  System.out.println("No se pudo crear el archivo");
                  e.printStackTrace();
                }
                finally
                {
                 try
                 {   salida.close(); }
                 catch (Exception e)
                 { System.out.println("No se pudo cerrar el flujo"); }
                }
        }
}
```


Reading the object



Deserializing an object

```
1.  import java.io.*;
2.  public class DeSerializaFigura
3.  {
4.       public static void main(String arg[])
5.       {
6.            FileInputStream archivo = null;
7.            ObjectInputStream entrada = null;
8.            Figura[] arreglo = new Figura[3];
9.            try
10.           {
11.                archivo = new FileInputStream("figuras.xyz");
12.                entrada = new ObjectInputStream(archivo);
13.                Object figurita = entrada.readObject();
14.                Figura figu = null;
15.                int i = 0;
16.                while (figurita != null)
17.                {
18.                  figu = (Figura)figurita;
19.                  System.out.println (figu.toString());
20.                  arreglo[i] = figu;
21.                  figurita = entrada.readObject();
22.                  i++;
23.                }
24.           }
26.  catch (EOFException e)
27.       {
28.            System.out.println("Fin archivo");
29.       }
30.       catch (Exception e)
31.       {
32.            System.out.println("error I/O lectura de
archivo");
33.       }
34.       finally
35.       {
36.       try
37.       {
38.            entrada.close();
39.       }
40.       catch (Exception e)
41.       { System.out.println("No se pudo cerrar el
flujo"); }
42.       }
43.       System.out.println("in detalle");
44.       System.out.println("- - - - - ");
45.       for (int i = 0; i < arreglo.length; i++)
46.       {
47.            arreglo[i].getPropiedades();
48.            System.out.println("- - - - - ");
49.       }
50.  }
51.  }
```


References

- **Cay Horstmann, Java 2 Fundamentos, Prentice Hall**