

Entrada/Salida

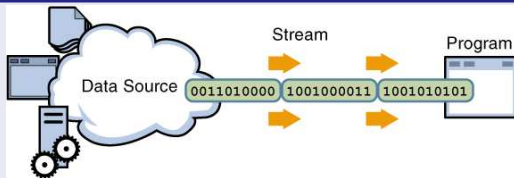
Luis Fernando Llana Díaz

Dept. Sistemas Informáticos y Computación Universidad Complutense de Madrid

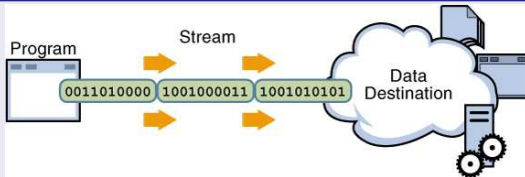
19 de abril de 2007

Flujos de entrada/salida

Stream de entrada



Stream de entrada



Byte Streams

```
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;

public class CopiaBytes {
    public static void main(String[] args) throws IOException {
        FileInputStream in = null;
        FileOutputStream out = null;
        try {
            in = new FileInputStream(args[0]);
            out = new FileOutputStream(args[1]);
            int c;

            while ((c = in.read()) != -1) {
                out.write(c);
            }

        } finally {
            if (in != null) {
                in.close();
            }
            if (out != null) {
                out.close();
            }
        }
    }
}
```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27

Llamada desde la línea de comandos

```
luis@casa:~/docencia/Java06-07$ java CopiaBytes quijote.txt quijote2.txt
```

1

```
public static void main(String[] args) throws IOException {  
    .....  
}  
  
args.length() ~ 2  
args[0] ~ "quijote.txt"  
args[1] ~ "quijote2.txt"
```

1

2

3

4

5

6

7

Streams caracteres

```
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;

public class CopiaCaracteres {
    public static void main(String[] args) throws IOException {
        FileReader inputStream = null;
        FileWriter outputStream = null;

        try {
            inputStream = new FileReader(args[0]);
            outputStream = new FileWriter(args[1]);

            int c = inputStream.read();
            while (c != -1) {
                outputStream.write(c);
                c = inputStream.read();
            }
        } finally {
            if (inputStream != null) {
                inputStream.close();
            }
            if (outputStream != null) {
                outputStream.close();
            }
        }
    }
}
```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28

Streams Líneas

```
import java.io.FileReader;
import java.io.FileWriter;
import java.io.BufferedReader;
import java.io.PrintWriter;
import java.io.IOException;

public class CopiaLineas {
    public static void main(String[] args) throws IOException {
        BufferedReader inputStream = null;
        PrintWriter outputStream = null;
        try {
            inputStream = new BufferedReader(new FileReader(args[0]));
            outputStream = new PrintWriter(new FileWriter(args[1]));

            String l = inputStream.readLine();
            while ( l!= null ) {
                outputStream.println(l);
                l = inputStream.readLine();
            }
        } finally {
            if (inputStream != null)
                inputStream.close();
            if (outputStream != null)
                outputStream.close();
        }
    }
}
```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27

Buffering

```
inputStream =  
    new BufferedReader(new FileReader(args[0]));  
outputStream =  
    new BufferedWriter(new FileWriter(args[1]));  
.....  
.....  
outputStream.flush(); //vaciamos el buffer de escritura
```

1
2
3
4
5
6
7

Leyendo cosas que no son caracteres

- Métodos `Integer.parseInt(java.lang.String)`,
`Double.parseDouble(java.lang.String)`
- Clase `java.util.Scanner`

class Scanner

```
import java.io.*;
import java.util.Scanner;

public class Escaner {
    public static void main(String[] args) throws IOException {
        Scanner s = null;
        try {
            s = new Scanner(new BufferedReader(new FileReader(args[0])));

            while (s.hasNext()) {
                System.out.println(s.next());
            }
        } finally {
            if (s != null) {
                s.close();
            }
        }
    }
}
```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19

class Scanner

```
import java.io.*;
import java.util.Scanner;

public class SumaScan {
    public static void main(String[] args) throws IOException {
        Scanner s = null;
        try {
            s = new Scanner(new BufferedReader(new FileReader(args[0])));
            double suma = 0;
            while (s.hasNext()) {
                if (s.hasNextDouble()) {
                    suma += s.nextDouble();
                } else {
                    String d = s.next();
                    suma += Double.parseDouble(d);
                    System.out.println("Error: "+d);
                }
            }
            System.out.format("Suma: %s\n", suma);
        } finally {
            if (s != null) {
                s.close();
            }
        }
    }
}
```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26



clase Scanner

Fichero numeros.txt

```
1.994.986,34
2.000.000,00
3.000.000,00
3000000
10.11
```

1
2
3
4
5

```
~/Java$ export CLASSPATH=./classes/
~/Java$ /opt/java/jdk1.5/bin/java SumaScan numeros.txt
Error:10.11
Suma: 9994996.45
```

1
2
3
4

Construyendo puntos

```
public class Punto implements FiguraPlana{
    private double posX,posY;
    public Punto(String s){
        Scanner scan = new Scanner(s);
        scan.useLocale(Locale.US);
        posX=scan.nextDouble();
        posY=scan.nextDouble();
    }
    .....
    .....
}
```

1
2
3
4
5
6
7
8
9
10
11

Construyendo puntos

```
public class Punto implements FiguraPlana{
    private double posX,posY;
    public Punto (String s){
        Scanner scan = new Scanner(s);
        posX=Double.parseDouble(scan.next());
        posY=Double.parseDouble(scan.next());
    }
    .....
    .....
}
```

1
2
3
4
5
6
7
8
9
10

Fichero de figuras

Fichero figuras.txt

```
circulo 1
5 5 2 2
segmento 3
3 4 8 4 4
recta 5
3 5 3 6 6
recta 7
0 1 1 1 8
circulo 9
0 2 0.5 10
recta 11
0 2 4 5 12
paralelogramo 13
0 4 2 2 -2 2 14
paralelogramo 15
0 0 2 2 -2 2 16
paralelogramo 17
2 2 0 4 0 0 18
paralelogramo 19
0 0 2 2 4 0 20
punto 21
2 3 22
```

Leyendo puntos

Fichero puntos.txt

```
4 5
7 3
1 1
0.5 1
1 0.5
7 4
0 2
```

1
2
3
4
5
6
7

Leyendo

```
public class PruebaGeometriaI{
    private static void leeFiguras(String fichero) {...}

    private static void leePuntos(String fichero) {...}

    public static void main(String args[]) throws IOException{
        String nombrePuntos=args[0];
        String nombreFiguras=args[1];
        System.out.println("Puntos:");
        leePuntos(nombrePuntos);
        System.out.println("Figuras:");
        leeFiguras(nombreFiguras);
    }
}
```

1
2
3
4
5
6
7
8
9
10
11
12
13
14

Leyendo puntos

```
private static void leePuntos(String fichero)
    throws FileNotFoundException, IOException{

    BufferedReader lector = new BufferedReader(new FileReader(fichero));
    String datos=lector.readLine();
    while (datos!=null){
        Punto p=new Punto(datos);
        System.out.println(p);
        datos=lector.readLine();
    }
}
```

1
2
3
4
5
6
7
8
9
10
11

Leyendo figuras

```
private static FiguraPlana construyeFigura(String tipoFigura, String datos){..}1
2
private static void leeFiguras(String fichero)3
    throws FileNotFoundException, IOException{4
5
    BufferedReader lector = new BufferedReader(new FileReader(fichero));6
    String tipoFigura=lector.readLine();7
    int i=1;8
    while (tipoFigura!=null){9
        String datos=lector.readLine();10
        FiguraPlana figura=construyeFigura(tipoFigura,datos);11
        System.out.println(figura);12
        tipoFigura=lector.readLine();13
    }14
}15
```

Construyendo figuras

```
private static FiguraPlana construyeFigura(String tipoFigura, String datos){ 1
    FiguraPlana figura; 2
    if (tipoFigura.equals("punto")) figura=new Punto(datos); 3
    else if (tipoFigura.equals("recta")) figura=new Recta(datos); 4
    else if (tipoFigura.equals("segmento")) figura=new Segmento(datos); 5
    else if (tipoFigura.equals("paralelogramo")) figura=new Paralelogramo(datos); 6
    else if (tipoFigura.equals("circulo")) figura=new Circulo(datos); 7
    else throw new RuntimeException("Figura no implementada:"+tipoFigura+":"); 8
    return figura; 9
} 10
```

Invocación desde la línea de comandos

Variable de entorno **CLASSPATH**

Debe apuntar al directorio raíz donde estén los ficheros **.class**

```
~/Java$ ls
build.xml  figuras.txt  prj.el      quijote3.txt  src
classes   lib          puntos.txt  quijote4.txt
CVS       numeros.txt quijote2.txt quijote.txt

~/Java$ echo $CLASSPATH

~/Java$ export CLASSPATH=./classes
~/Java$ echo $CLASSPATH
./classes/
```

1
2
3
4
5
6
7
8
9

Invocación desde la línea de comandos

Variable de entorno `JAVA_HOME`

Debe apuntar al directorio donde está la instalación de Java.

```
~/Java$ ls -ld /opt/java/jdk1.5
lrwxrwxrwx 1 root root 11 2006-07-18 07:53 /opt/java/jdk1.5 -> jdk1.5.0_07
~/Java$ echo $JAVA_HOME
/opt/java/jdk
~/Java$ export JAVA_HOME=/opt/jdk/jdk1.5
~/Java$ echo $JAVA_HOME
/opt/jdk/jdk1.5
~/Java$ $JAVA_HOME/bin/java PruebaGeometriaI puntos.txt figuras.txt
Puntos:
Punto: (4.0,5.0)
.....
Punto: (0.0,2.0)
Figuras:
Círculo: Punto: (5.0,5.0):2.0*
Segmento: Punto: (3.0,4.0)-Punto: (8.0,4.0)
Recta: 1.0*X + 0.0*Y = 3.0
.....
Paralelepípedo*Punto: (2.0,2.0):Punto: (0.0,4.0):Punto: (-2.0,2.0):Punto: (0.0,0.0)*
Paralelepípedo*Punto: (0.0,0.0):Punto: (2.0,2.0):Punto: (6.0,2.0):Punto: (4.0,0.0)*
Punto: (2.0,3.0)
```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20