Los flujos de datos

Toda la información que se transmite a través de un ordenador fluye desde una entrada hacia una salida. Para transmitir información, Java utiliza unos objetos especiales denominados streams (flujos o corrientes).

Los stream permiten transmitir secuencias ordenadas de datos desde un origen a un destino. El origen y el destino pueden ser un fichero, un String o un dispositivo (lectura de teclado, escritura en pantalla).

Java dispone de dos tipos de flujos:

- Flujos de entrada o lectura (input streams).
- Flujos de salida o escritura (output streams).

Las clases que representan flujos de datos se dividen además en dos tipos:

- Flujos de datos en formato Unicode de 16 bits: derivados de las clases abstractas Reader y Writer.
- Flujos de bytes (información binaria): derivados de las clases abstractas InputStream y OutputStream.

Los flujos de datos además se clasifican en:

- **Iniciadores:** vuelcan o recogen datos directamente del dispositivo.
- **Filtros:** se sitúan entre el *stream* iniciador y el programa. Aportan una funcionalidad más especializada facilitando la lectura o escritura.

Escribir líneas en un fichero de texto

Toda operación de salida o escritura consta de tres pasos:

1. Abrir fichero para escritura.
2. Escribir líneas en el fichero.
3. Cerrar el fichero.

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

public class Principal {
	public static void main(String args[]) throws IOException {
		// Abrir fichero para escritura
		FileWriter file; // Iniciador
		try {
			file = new FileWriter("C:\\cine\\peliculas.txt");
		} catch (IOException e) {
			System.out.println("No se puede abrir el fichero");
			System.out.println(e.getMessage());
			return;
		}

		// Abrir buffer y escribir tres líneas
```

```
		BufferedWriter buffer = new BufferedWriter(file); // Filtro
		try {
			buffer.write("¡Bienvenido, Mister Marshall!");
			buffer.newLine();
			buffer.write("Con la muerte en los talones");
			buffer.newLine();
			buffer.write("Muerte de un ciclista");
			buffer.newLine();
		} catch (IOException e) {
			System.out.println("Error al escribir en el fichero");
			System.out.println(e.getMessage());
		}

		// Cerrar el buffer y el fichero
		try {
			buffer.close();
			file.close();
		} catch (IOException e) {
			System.out.println("Error al cerrar el fichero");
			System.out.println(e.getMessage());
		}
	}
}
```

Este programa crea el fichero cada vez que se ejecuta sobre escribiendo el anterior, si lo que queremos es añadir líneas a un fichero existente, lo que necesitamos es pasar al constructor de la clase FileWrite el valor true como segundo argumento.

```
file = new FileWriter("C:\\cine\\peliculas.txt", true);
```

Leer un fichero de texto

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

public class Principal {
	public static void main(String args[]) {

		// Abrir fichero para lectura
		FileReader file; // Iniciador
		try {
			file = new FileReader("C:\\cine\\peliculas.txt");
		} catch (IOException e) {
			System.out.println("No se puede abrir el fichero");
			System.out.println(e.getMessage());
			return;
		}

		// Abrir buffer y escribir tres líneas
		BufferedReader buffer = new BufferedReader(file); // Filtro
		String linea = "";
		try {
			linea = buffer.readLine();
```

```
                while (linea != null) {
                        System.out.println(linea);
                        linea = buffer.readLine();
                }
        } catch (IOException e) {
                System.out.println("Error al leer el fichero");
                System.out.println(e.getMessage());
        }

        // Cerrar el buffer y el fichero
        try {
                buffer.close();
                file.close();
        } catch (IOException e) {
                System.out.println("Error al cerrar el fichero");
                System.out.println(e.getMessage());
        }

    }
}
```

Escritura de un fichero binario

Queremos guardar en un fichero datos de artículos, para que sea más sencillo, crearemos una clase Producto:

```
public class Producto {
        private String nombre;
        private float precio;
        private float unidadesEnExistencia;

        public Producto(String nombre, float precio, float unidadesEnExistencia) {
                this.nombre = nombre;
                this.precio = precio;
                this.unidadesEnExistencia = unidadesEnExistencia;
        }

        public String getNombre() {
                return nombre;
        }
        public float getPrecio() {
                return precio;
        }
        public float getUnidadesEnExistencia() {
                return unidadesEnExistencia;
        }

        @Override
        public String toString() {
                return nombre + " Stock: " + this.unidadesEnExistencia + " Precio: " +
                        this.precio;
        }
}
```

Ahora ya podemos crear un programa que grabe tres artículos en un fichero:

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

public class Principal {

        public static void main(String[] args) {
```

```
                // Creación de 3 objetos producto
                Producto p1 = new Producto("Manzanas Royal Gala",2.50f,7f);
                Producto p2 = new Producto("Dátiles de la tía Julita",3.25f,12f);
                Producto p3 = new Producto("Mandarinas Clementinas",2.20f,25f);

                FileOutputStream fichero;
                DataOutputStream escritor;

                // Apertura del fichero almacen.dat
                try {
                        fichero = new FileOutputStream("almacen.dat", true);
                        escritor = new DataOutputStream (fichero);
                } catch (IOException e) {
                        System.out.println("No se ha podido abrir el fichero almacen.dat");
                        System.out.println(e.getMessage());
                        return;
                }

                // Escribir datos en el fichero almacen.dat
                try {
                        escritor.writeUTF(p1.getNombre());
                        escritor.writeFloat(p1.getPrecio());
                        escritor.writeFloat(p1.getUnidadesEnExistencia());

                        escritor.writeUTF(p2.getNombre());
                        escritor.writeFloat(p2.getPrecio());
                        escritor.writeFloat(p2.getUnidadesEnExistencia());

                        escritor.writeUTF(p3.getNombre());
                        escritor.writeFloat(p3.getPrecio());
                        escritor.writeFloat(p3.getUnidadesEnExistencia());

                } catch (IOException e) {
                        System.out.println
                        ("Ha ocurrido un error al escribir datos en el fichero");
                        System.out.println(e.getMessage());
                }

                try {
                        escritor.close();
                        fichero.close();
                } catch (IOException e) {
                        System.out.println("Ha ocurrido un error al cerrar el fichero");
                        System.out.println(e.getMessage());
                }
        }
}
```

Lectura de un fichero binario

```
import java.io.DataInputStream;
import java.io.EOFException;
import java.io.FileInputStream;
import java.io.IOException;

public class Principal {

        public static void main(String[] args) {
                FileInputStream fichero;
                DataInputStream lector;
                try {
                        fichero = new FileInputStream("almacen.dat");
                        lector = new DataInputStream (fichero);
                } catch (IOExceptione) {
                        System.out.println("Ha ocurrido un error al abrir el fichero");
                        System.out.println(e.getMessage());
                        return;
                }

                boolean eof = false;
```

```
                while (!eof) {
                        try {
                                String pro = lector.readUTF();
                                float pre = lector.readFloat();
                                float uni = lector.readFloat();
                                Producto p = new Producto(pro, pre, uni);
                                System.out.println(p);
                        } catch (EOFException e1) {
                                eof = true;
                        } catch (IOException e2) {
                                System.out.println
                               ("Ha ocurrido un error al leer los registros");
                                System.out.println(e2.getMessage());
                                break; // sale del bucle while
                        }
                }

                try {
                        lector.close();
                        fichero.close();
                } catch (IOException e) {
                        System.out.println("Ha ocurrido un error al cerrar el fichero");
                        System.out.println(e.getMessage());
                }
        }

}
```

Lectura de datos con DataInputStream

DataInputStream provee métodos para la lectura secuencial de tipos de datos elementales desde un fichero binario:

- ***readBoolean():*** lee un valor de tipo *boolean* del fichero.
- ***readByte():*** lee un valor de tipo *byte* del fichero.
- ***readChar():*** lee un valor de tipo *char*del fichero.
- ***readDouble():*** lee un valor de tipo *double* del fichero.
- ***readFloat():*** lee un valor de tipo *float* del fichero.
- ***readInt():*** lee un valor de tipo *int* del fichero.
- ***readLong():*** lee un valor de tipo *long* del fichero.
- ***readShort():*** lee un valor de tipo *short* del fichero.
- ***readUTF():*** lee una cadena en formato UTF del fichero.