

ESTRUCTURA DE DATOS I

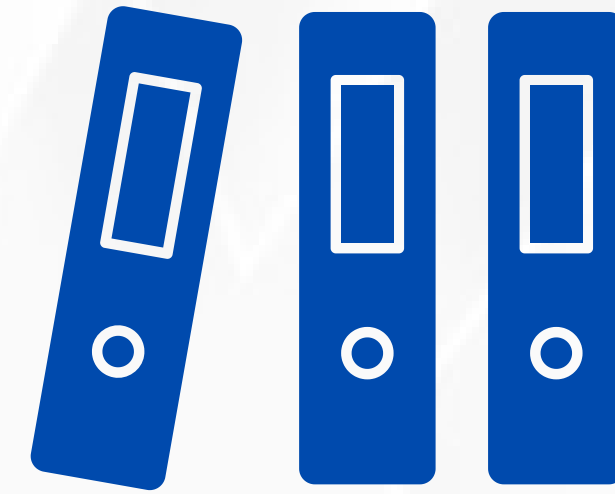
Lectura y escritura de archivos



MEng. Andrés Castillo

06/02/24

Archivos



Se considera un archivo (fichero) en informática a un conjunto de datos que se encuentran almacenados en un dispositivo.

Este conjunto de datos viene agrupado por un nombre, una ruta de acceso y una extensión.

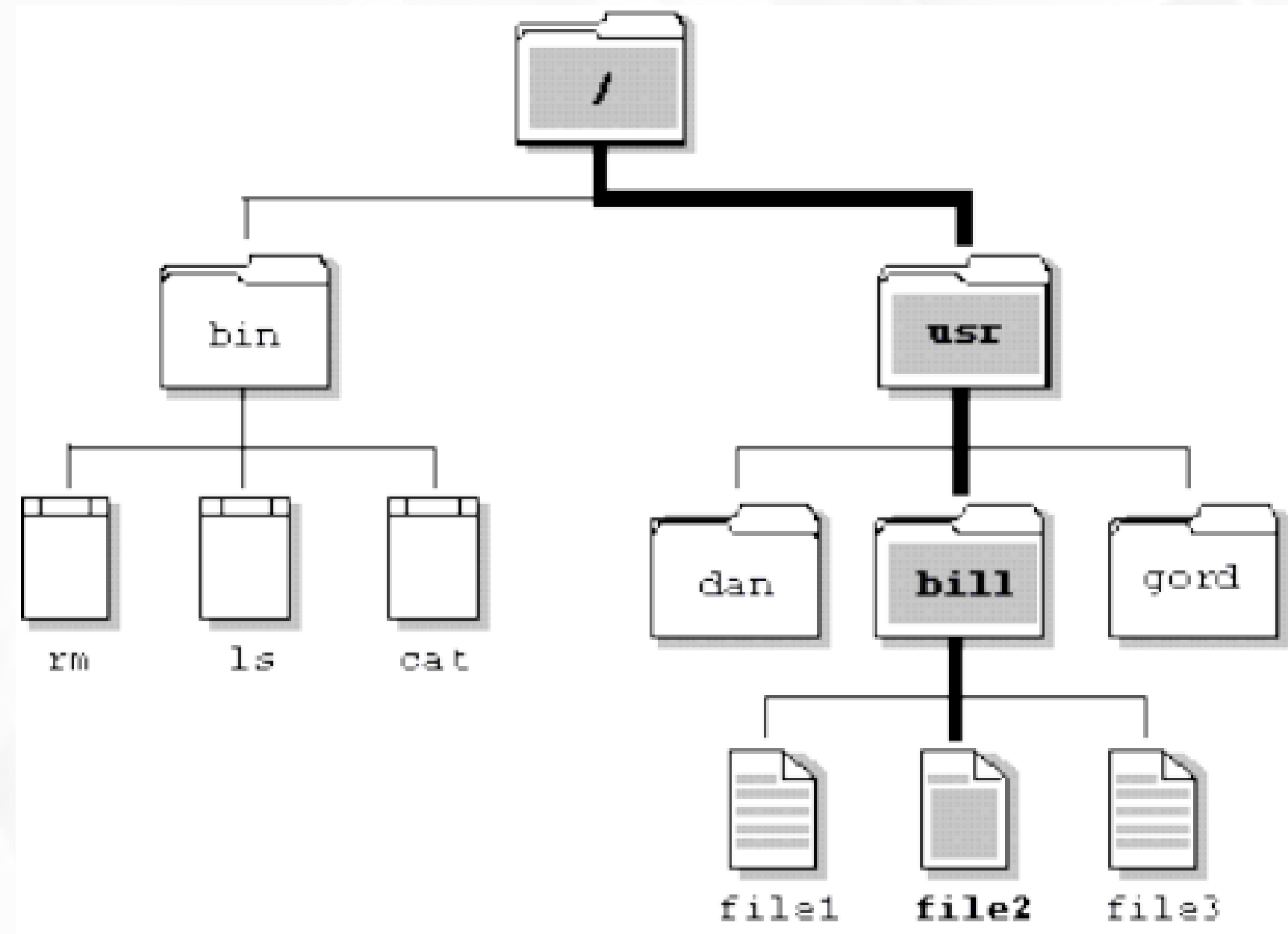
Los ficheros se guardan en dispositivos de almacenamiento fijo como discos duros, pendrives, tarjetas, etc.

Sistema de ficheros

Los ficheros se suelen organizar de forma jerárquica.

No pueden existir ficheros con el mismo:

- Nombre
- Ruta
- Extensión



Rutas



Para acceder a un determinado fichero, se utiliza la ruta (path)

En una ruta, cada nivel de la jerarquía se representa delimitado por el símbolo /.

Existen 2 tipos de rutas:

- Absoluta: Ruta al fichero desde el directorio principal (root). Ej: `/home/Documentos/ejemplo.txt`
- Relativa: Ruta al fichero desde el directorio actual. Ej: Estando en home, `./Documentos/ejemplo.txt`

Archivo



En un ordenador, lo único que se es capaz de manejar es un conjunto de bits (0, 1).

Estos bits se agrupan en la unidad mínima, el byte.

Podemos considerar a todo fichero en nuestro ordenador como un conjunto de bytes, independientemente de su extensión

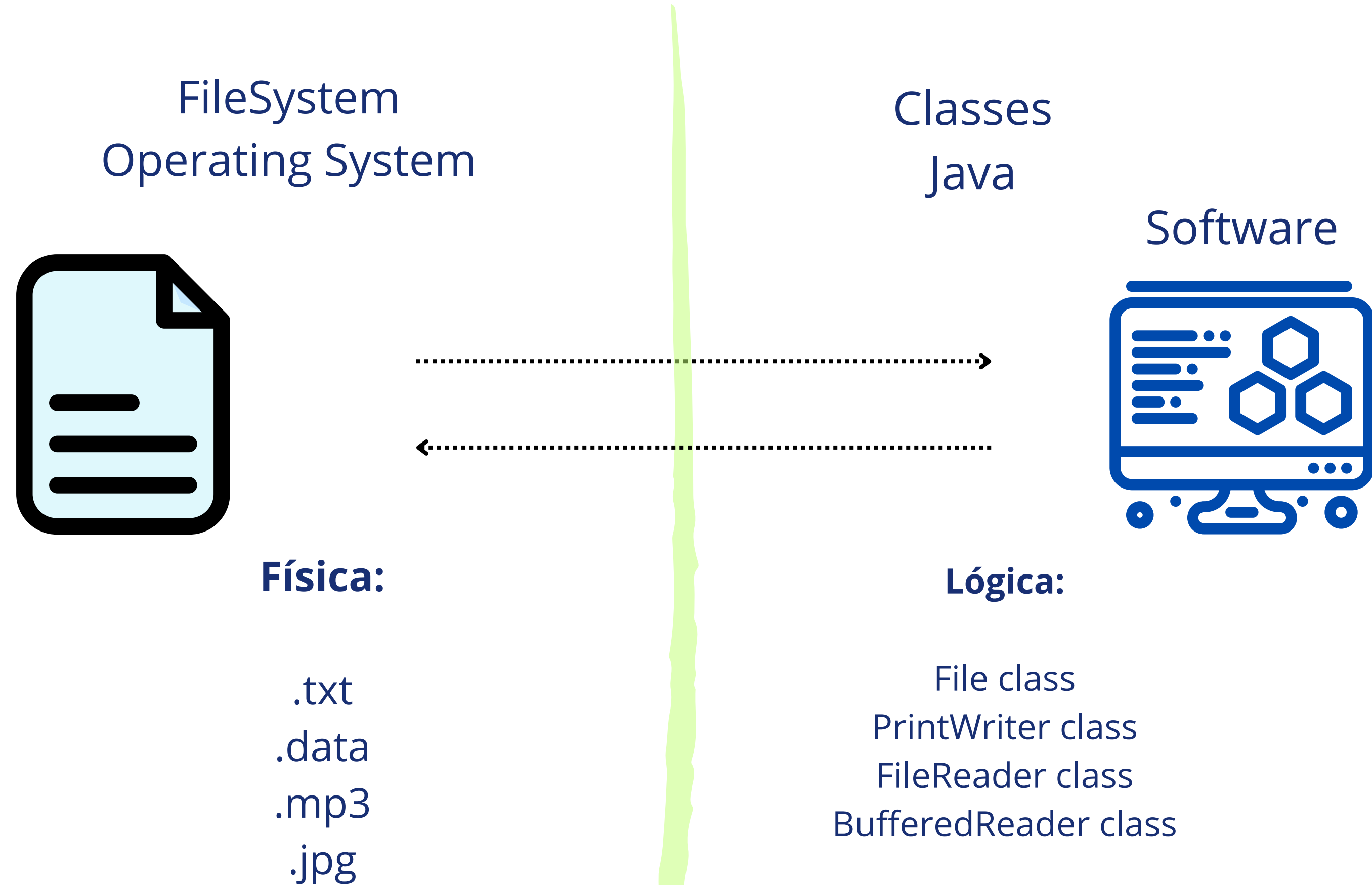
Interpretar el significado de estos bytes es responsabilidad del programa que manipula este fichero.

Operaciones con ficheros

- _____
- _____
- _____
- _____

Apertura.
Cierre.
Lectura.
Escritura.
Ejecución.
Creación.
Eliminación.

Archivos en Java



Apertura y cierre de un fichero



- Cuando se abre un fichero en Java, se “reserva” el fichero para operar con el.
- Se establece un flujo de datos desde el fichero a una **variable en Java**, que representa al fichero.

La clase File

La clase que manipula los ficheros en Java se llama File.

Con esta clase se pueden hacer un gran número de operaciones sobre un fichero y sus propiedades, pero no se permite leer ni escribir.

También permite obtener datos del fichero, como rutas, nombres, permisos e incluso si existe.

El resto de clases que manipulan ficheros parten de la existencia de una clase File, por lo que es la base de cualquier operación de manipulación de ficheros.

La clase File

El constructor de la clase File tiene el siguiente esquema:

```
File variableFichero = new File("ruta fichero");
```

Al crear el constructor, la variable **variableFichero** es un objeto con los datos del fichero que se encuentra en la ruta pasada por parámetro, si es que existe. La ruta puede ser absoluta o relativa.

Para cerrar un fichero:

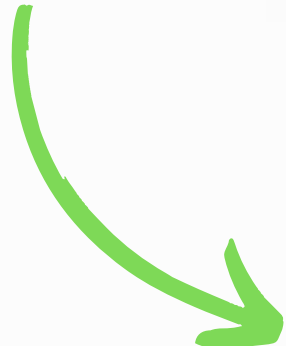
```
variableFichero.close();
```

Documentación de clase File:

<https://docs.oracle.com/javase/7/docs/api/java/io/File.html>

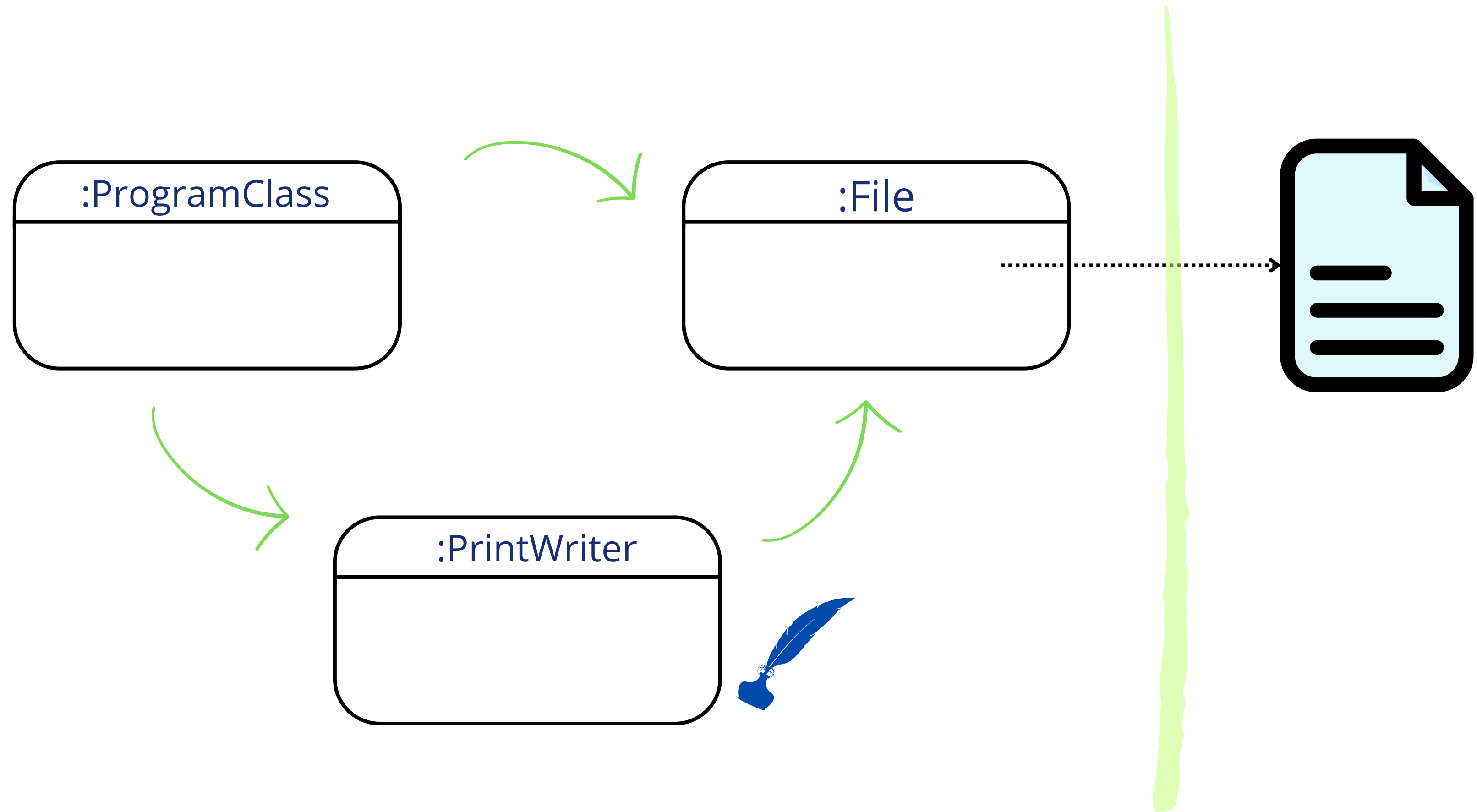
La clase File

```
public static void main(String args[]) {  
    File fichero = new File("FicheroEjemplo.txt");  
  
    if (fichero.exists()) {  
        System.out.println("Nombre del archivo "+ fichero.getName());  
        System.out.println("Ruta "+ fichero.getPath());  
        System.out.println("Ruta absoluta "+ fichero.getAbsolutePath());  
        System.out.println("Se puede escribir "+fichero.canRead());  
        System.out.println("Se puede leer "+fichero.canWrite());  
        System.out.println("Tamaño "+fichero.length());  
    }  
  
    fichero.close();  
}
```



Nombre del archivo: FicheroEjemplo.txt
Ruta: FicheroEjemplo.txt
Ruta absoluta: C:\DirectorioEjemplo\FicheroEjemplo.txt
Se puede escribir: false
Se puede leer: true
Tamaño: 1366

Escritura de archivos en Java



Escritura de datos de un fichero

- Clase **File**: Para representar el fichero que se quiere leer

```
File fichero = new File("ruta fichero");
```

- Clase **FileWriter**: Establece el stream de datos de escritura del fichero. Tiene una serie de métodos para escribir en ficheros. Al constructor del `FileWriter` recibe el objeto `File`.

```
FileWriter writer = new FileWriter(fichero);
```

- Clase **PrintWriter**: Crea un buffer a través del `FileWriter`, que permite extender los métodos del `FileWriter` por otros similares a los que tenemos en la salida de pantalla. El constructor recibe el `FileWriter` como parámetro.

```
PrintWriter pw = new PrintWriter(writer);
```

Escritura de datos de un fichero

Entre las funciones que tenemos en el **PrintWriter**, las mas comunes son:

- **print**("texto"). Imprime el texto pasado por parámetro
- **println**("texto"). Imprime el texto pasado por parámetro y hace un salto de línea
- El constructor del **FileWriter** puede recibir un segundo parámetro boolean que indica si queremos escribir el fichero desde cero (false) o si queremos añadir texto al existente (true)

```
FileWriter writer = new FileWriter(fichero);
```

```
FileWriter writer = new FileWriter(fichero, true);
```

```
FileWriter writer = new FileWriter(fichero, false);
```

Desde cero

Añadir texto

Escritura de datos de un fichero

```
import java.io.*;

public class EscribirFichero
{
    public static void main(String[] args)
    {
        File fichero = null;
        FileWriter writer = null;
        PrintWriter pw = null;
        try
        {
            fichero = new File("C:\\directorioArchivo\\archivo.txt");
            writer = new FileWriter(fichero);
            pw = new PrintWriter(writer);

            for (int i = 0; i < 10; i++) {
                pw.println("Linea " + i);
            }

        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            try {
                if (null != fichero) {
                    fichero.close();
                }
            } catch (Exception e2) {
                e2.printStackTrace();
            }
        }
    }
}
```

Lectura de datos de un fichero

Para leer datos de un fichero de texto, utilizaremos las siguientes clases:

- Clase **File**: Para representar el fichero que se quiere leer.

```
File fichero = new File("ruta fichero");
```

- Clase **FileReader**: Establece el stream de datos de lectura del fichero. Tiene una serie de métodos para leer carácter a carácter. Al constructor del FileReader recibe el objeto File.

```
FileReader reader = new FileReader(fichero);
```

- Clase **BufferedReader**: Crea un buffer a través del FileReader, que permite leer mas de un carácter. El constructor recibe el FileReader como parámetro.

```
BufferedReader buffer = new BufferedReader (reader);
```


Lectura de datos de un fichero

Utiliza la función del **BufferedReader** llamada **readLine()**, la cual:

- Devuelve la siguiente línea de texto si existe, si no existe, devuelve null.

```
String linea = buffer.readLine();
```

- Teniendo en cuenta el funcionamiento de `readLine()`, se puede leer todo el fichero utilizando un bucle `while`.

```
String linea;  
while((linea=buffer.readLine()) != null) {  
    System.out.println(linea);  
}
```

Lectura de datos de un fichero

```
import java.io.*;

class LeeFichero {
    public static void main(String [] arg) {
        File archivo = null;
        FileReader reader = null;
        BufferedReader buffer = null;

        try {
            archivo = new File("C:\\directorioArchivo\\archivo.txt");
            reader = new FileReader (archivo);
            buffer = new BufferedReader(reader);

            String linea;
            while( (linea=buffer.readLine()) != null) {
                System.out.println(linea);
            }
        }
        catch(Exception e){
            e.printStackTrace();
        }finally{
            try{
                if( null != fr ){
                    fr.close();
                }
            }catch (Exception e2){
                e2.printStackTrace();
            }
        }
    }
}
```

Borrado de un fichero

Para borrar un fichero, se utiliza la clase **File**.

- Existe la función **delete()** que elimina el fichero.

```
public static void main(String args[]) {  
    File fichero = new File("FicheroEjemplo.txt");  
  
    if (fichero.exists()) {  
        System.out.println("Nombre del archivo "+ fichero.getName());  
        System.out.println("Ruta "+ fichero.getPath());  
        System.out.println("Ruta absoluta "+ fichero.getAbsolutePath());  
        System.out.println("Se puede escribir "+fichero.canRead());  
        System.out.println("Se puede leer "+fichero.canWrite());  
        System.out.println("Tamaño "+fichero.length());  
    }  
  
    fichero.delete();  
}
```

Reto

Apropiación de lectura de archivos - Numeral 3.6
páginas 107-110

Apropiación de Escritura de archivos - numeral 3.8
páginas 115-116

A practicar!!

“El arte se opone a la tecnología y la tecnología inspira el arte”

John Lasseter