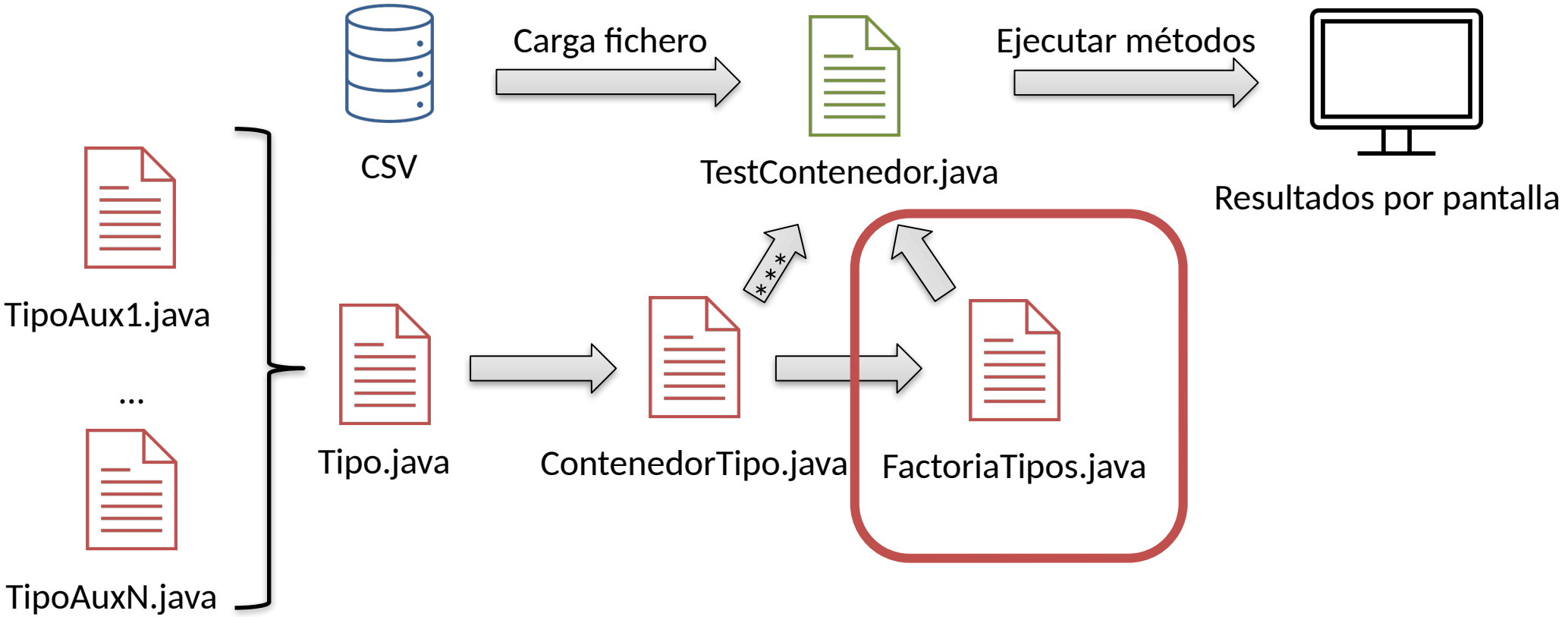
FUNDAMENTOS DE PROGRAMACION

4 – Tipo Factoría

UNIVERSIDAD DE SEVILLA

¿Qué se ve en este tema?



ÍNDICE

ÍNDICE

Lectura de un fichero y carga en una coleccion
Método 1

En general manejaremos colecciones que suelen ser el resultado de cargar en una colección (*una lista*) los registros de un fichero.

*Esquema 1: Una **clase con un método** y se necesita un **constructor a partir de String** en la clase que implementa el Tipo.*

```
public class FactoriaTipo {
   public static List<Tipo> leer«Tipos» (String
   nombreFichero){
         …
   }
}
```

Lectura de un fichero y carga en una coleccion
Método 1


```
public class FactoriaTipo {
   public static List<Tipo> leer«Tipos» (String nombreFichero){
   List<Tipo> res=new ArrayList<Tipo>();

    try {
    List<String> lineas = Files.readAllLines(Paths.get(nombreFichero));

    for (String linea:lineas){
     Tipo t=  new Tipo(linea);
    res.add(t);
       }
    } catch (IOException e) {
    e.printStackTrace();
    }
    return res;
   }
```

El método *readAllLines* lee de una vez todas las líneas y devuelve una lista de String en que cada registro se almacena en un elemento de la lista.

Lectura de un fichero y carga en una coleccion
Método 1: Libros


```
public class FactoriaLibros {

public static List<Libro> leerLibros(String nombreFichero){
	List<Libro> res=new ArrayList<Libro>();
	try {
		List<String> lineas=Files.readAllLines(Paths.get(nombreFichero));
		for (String linea:lineas) {
			res.add(new Libro(linea)); //Constructor a partir de String
		}
	}catch (IOException e) {
		e.printStackTrace();
	}
	return res;
}

}
```

Lectura de un fichero y carga en una coleccion
Método 1: Libros


```
public class Libro {
        . . .
                public Libro(String linea){
                        String [] trozos=linea.split(",");

                        Checkers.check("La cadena no se trocea bien", trozos.length==5);

                        this.título = trozos[0].trim();

                        String autor = trozos[1].trim();
                        Checkers.check(“Autor no puede estar vacío”, !autor.equals(“”));
                        this.autor = autor;

                        this.númeroPaginas = Integer.valueOf(trozos[2].trim());
                        this.fechaAdquisicion = LocalDate.parse(trozos[3].trim());
                        this.gênero = Genero.valueOf(trozos[4].trim());

                        }
}
```

ÍNDICE

1. Método 1: método lectura + constructor a partir de String
2. **Método 2: método lectura + método de parseo a partir de String**

Lectura de un fichero y carga en una coleccion
Método 2

*Esquema 2: Una **clase con dos métodos** y no hay constructor a partir de String en la clase que implementa el Tipo*

```
public class FactoriaTipo {
public static List<Tipo> leer«Tipos» (String nombreFichero){
…
}
private static Tipo parsearTipo(String linea){
…
aquí se “parseará” cada linea del fichero
…
}
```

Lectura de un fichero y carga en una coleccion
Método 2


```
public class FactoriaTipo {

public static List<Tipo> leer«Tipos» (String nombreFichero){
     List<Tipo> res=new ArrayList<Tipo>();
     try {
     List<String> lineas=Files.readAllLines(Paths.get(nombreFichero));
     for (String linea:lineas){
     res.add(parsearTipo(linea)); //parsear devuelve un objeto
     }
     } catch (IOException e) {
     e.printStackTrace();
     }
return res;
}
```

Lectura de un fichero y carga en una coleccion
Método 2


```
public class FactoriaTipo {
      ...
  private static Tipo pasearTipo(String linea){
    String [ ] trozos= linea.split("separador");
    Checkers.check("La cadena no se trocea bien",trozos.length==X);
    Tipo1 param1= ...trozos[0].trim()...;
    Tipo2 param2= ...trozos[1].trim()..;
    Tipo3 param3= ...trozos[2].trim()..;
    …
    return (new Tipo (param1, param2, param3, …));
   }
}
```

Lectura de un fichero y carga en una coleccion
Método 2


```
public class FactoriaLibros {

	public static List<Libro> leerLibros(String nombreFichero){
		List<Libro> res=new ArrayList<Libro>();
		try {
			List<String>lineas=Files.readAllLines(Paths.get(nombreFichero));
			for (String linea:lineas) {
				res.add(parsearLibro(linea));
		}
		} catch (IOException e) {
			e.printStackTrace();
		}
		return res;
	}
}
```

Lectura de un fichero y carga en una coleccion
Método 2


```
public class FactoriaLibros {
    …
        private static Libro parsearLibros(String linea){
        String [] trozos=linea.split(",");

        Checkers.check("La cadena no se trocea bien", trozos.length==5);

        String título=trozos[0].trim();
        String autor=trozos[1].trim();
        Integer númeroPaginas=Integer.valueOf(trozos[2].trim());
        LocalDate fechaAdquisicion=LocalDate.parse(trozos[3].trim());
        Genero genero=Genero.valueOf(trozos[4].trim());

        return new Libro(título, autor, númeroPaginas, fechaAdquisicion, genero);
        }
}
```