Universidad de Barcelona

Arthur Font

Cristian Rodríguez

Proyecto de Prácticas

Programación II

Práctica 2 – 31/03/2019

Barcelona

2019

Index

1. Introducción

El objetivo de la práctica es implementar un programa que haga las veces de menu para poder manipuar archivos, tanto de tipo audio, como de tipo video. A su vez, trabajamos el modus operandi de modelo, vista y controlador. También trabajamos el tema del input/outputstreams con objetos y el tratamiento de excepciones. Además profundizamos el estudio y manejo de las heréncias y las interfícies.

2. Análisis

Al principio discutimos las funciones de las clases a implementar y definimos las relaciones entre ellas.
Hemos querido realizar esta práctica yendo desde el terreno mas ambigüo hasta lo más específico. Comenzando así por el menú, que al principio no realizaba nada ya que era solo era estructural. Continuamos con la implementación de la estructura para guardar los diferentes tipos de ficheros. Ahora ya podemos empezar a implementar los propios tipos de archivos, es decir, el audio y el video, pasando antes por la clase abstracta fitxerReproduible que servirá como base para crearlos.

3. Desarrollo

Hemos creado las clases Audio y Video las cuáles heredan de FitxerReproduible. Los contructores de estas clases los hemos definido usando los atributos comunes para el método super(), y por otra parte dando valor a los atributos específicos de cada clase.

Constructores utilizados:

```
public Video(String cami, String nom, String codec, float durada, int alcada,
        int amplada, float fps, Reproductor r) {
    super(cami, nom, codec, durada, r);
    this.alcada = alcada;
    this.amplada = amplada;
    this.fps = this.fps;
}
```

```
public Audio(String cami, File fitxerImatge, String nom, String codec, float
            durada,  int kbps, Reproductor r) {
    super(cami, nom, codec, durada, r);
    this.fitxerImatge = fitxerImatge;
    this.kbps = kbps;
}
```

Además, implementamos los métodos para guardar y recuperar los datos en una copia de seguridad en el disco.
Sigue abajo el funcionamento:

```
public void guardarDadesDisc(String camiDesti) throws AplicacioException,
            FileNotFoundException, IOException, ClassNotFoundException {
    this.fitxer = new FitxerMultimedia(camiDesti);
    FileOutputStream fout= new FileOutputStream(fitxer);
    ObjectOutputStream oos = new ObjectOutputStream(fout);
    oos.writeObject(this.biblio);
    oos.close();
}
```

```
public void carregarDadesDisc(String camiOrigen) throws AplicacioException,
            FileNotFoundException, IOException, ClassNotFoundException {
    FileInputStream fin= new FileInputStream(this.fitxer);
    ObjectInputStream ois = new ObjectInputStream(fin);
    this.biblio = (BibliotecaFitxersMultimedia)ois.readObject();
    ois.close();
}
```

Por tener que trabajar con el una classe como que utilize las funciones de las clases ObjectInputStream/ObjectOutputStream y las clases FileInputStream/ FileOutputStream hemos tenido que implementar la interfície Serializable en todas las clases implicadas en el proceso. Al no haber trabajado nunca con las clases que utilizan el iostream tuvimos algun que otro problema, pero con insisténcia y ganas conseguimos arreglarlos y hacer que la cópia de seguridad funcione correctamente.

4. Respuestas

1) Reutilizamos las clases CarpetaFitxers, FitxerMultimedia y Aplicació UB1. En la clase CarpetaFixters y FitxerMultimedia, adicionamos el tratamiento de excepciones ya que ahora el proyecto se adapta al esquema model-vista-controlador y implementamos la interfície Serializable, para poder utilizar las funciones de ObjectStream.
En la nueva clase AplicacioUB2, reutilizamos el codigo de AplicacioUB1 como base y la modificamos para que atenda las nuevas opciones del menú.

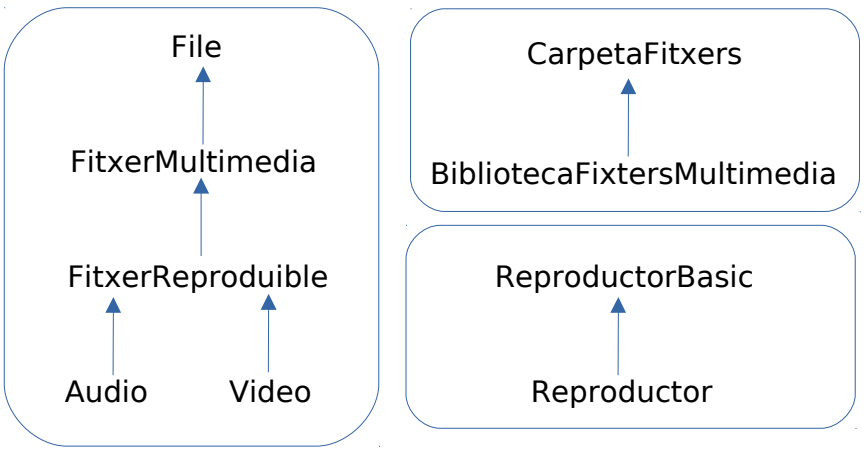
2) Creamos primeiramente un objecto de la clase AplicacioUB2, donde se crea un objecto de la clase Controlador, donde se crea un objecto de la clase Dades, donde se crea un objecto de la clase BibliotecaFitxersMultimedia. Como BibliotecaFitxersMultimedia herda de CarpetaFitxers, inicializamos por final objecto de la clase ArrayList que esta definido por el constructor de la clase madre.

3) Implementamos el metodo equals() para verificar si dos archivos són iguals. La utilizamos en el método addFitxer() de BibliotecaFitxersMultimedia para prohibir que hayan archivos duplicados en nuestra biblioteca.

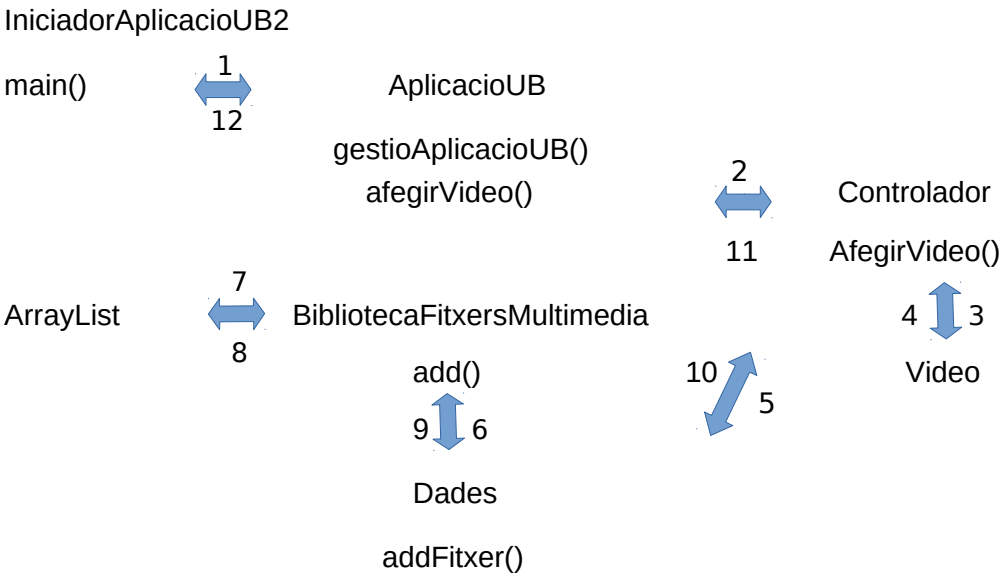
4) Utlizamos la clase AplicacioException para controlar las excepciones de nuestro progama, llançandolas desde las clases del model, después por el controlador y cogendo-las en la vista.

5) No hemos utilizado sobrecarga en controlador, simplemente hemos usado de parámetros los atributos necesarios para llamar al método deseado.

6) Diagrama de clase



7) Diagrama de flux



8) Para que imprima un mensaje diferente dependiendo de si es un audio o video:

FitxerReproduible fr1 = new Audio(repro);

FitxerReproduible fr2 = new Video(repro);

fr1.reproduir();

fr2.reproduir();

9) Hemos realizado las pruebas propuestas por la propia práctica y cada una de las pruebas ha salido correctamente.

10) Ha sido una práctica interesante ya que hemos empezado a utilizar archivos guardados en el disco con el input/outputstream. También cabe recalcar el sistema de separación modelo/vista/controlador.

5. Resultados

Los resultados de la práctica fueron los resultados esperados.

Todas las pruebas fueron superadas.

1) Añadimos un fichero tipo audio

```
	try {

		control.afegirAudio("/home/arthurfont/Downloads/aud.mp3", "aaa", "aud.mp3",
"codec", 11, 11);

	}
	catch (AplicacioException except) {
		except.getMessage();
	}
```

2) Añadimos otro fichero, ahora de tipo Video

```
	try {
	   control.afegirVideo("/home/arthurfont/Downloads/vid.mp4", "vid.mp4", "doced", 11, 11,
11, 11);
	}
	catch (AplicacioException except) {
	   except.getMessage();
	}
```

3) Añadimos otro fichero más, de tipo Video otra ve

```
	try {
	   control.afegirVideo("/home/arthurfont/Downloads/ala.mp4", "ala.mp4", "doced", 11, 11,
11, 11);
	}
	catch (AplicacioException except) {
	   except.getMessage();
	}
```

4) Imprimimos los datos guardados de la biblioteca por pantalla

```
	System.out.println("Carpeta Fitxers:\n==========\n" + control.mostrarBiblioteca());
```

5) Añadimos otro fichero, esta vez es un fichero que ya existe en la biblioteca, por lo tanto deberia saltar una excepción y no guardar nada

```
	try {
	   control.afegirVideo("/home/arthurfont/Downloads/ala.mp4", "ala.mp4", "doced", 11, 11,
11, 11);
	}
	catch (AplicacioException except) {
	   except.getMessage();
	}
```

6) Volvemos a imprimir los datos guardados de la biblioteca por pantalla y si ha salido bien solo deben haber tres ficheros

```
	System.out.println("Carpeta Fitxers:\n==========\n" + control.mostrarBiblioteca());
```

7) Añadimos otro fichero, esta vez el fichero no existe en el disco, por lo tanto deberia saltar una excepción y no guardar nada

```
try {
   control.afegirVideo("/home/arthurfont/Downloads/sf.mp4", "sf.mp4", "doced", 11, 11, 11,
11);
}
catch (AplicacioException except) {
   except.getMessage();
}
```

8) Volvemos a imprimir los datos guardados de la biblioteca por pantalla y si ha salido bien solo deben haber tres ficheros

```
System.out.println("Carpeta Fitxers:\n==============\n" + control.mostrarBiblioteca());
```

9) Ahora guardamos los datos en la copia de seguridad en el disco

```
try {
   control.guardarDadesDisc("/home/arthurfont/Music/copia.dat");
}
catch (AplicacioException except) {
   except.getMessage();
}
```

10) Ahora borramos el segundo fichero guardado en la biblioteca

```
try {
   control.esborrarFitxer(2);
}
catch (AplicacioException except) {
   except.getMessage();
}
```

11) Verificamos que solo tenemos dos ficheros en la biblioteca

```
System.out.println("Carpeta Fitxers:\n==============\n" + control.mostrarBiblioteca());
```

12) Cargamos la copia de seguridad en la biblioteca para recuperar el fichero borrado

```
try {
   control.carregarDadesDisc("/home/arthurfont/Music/copia.dat");
}
catch (AplicacioException except) {
   except.getMessage();
}
```

13) Al final volvemos a mostrar por pantalla los datos de la biblioteca para ver si la carga de la copia ha funcionado correctamente

```
System.out.println("Carpeta Fitxers:\n==============\n" + control.mostrarBiblioteca());
```