Universidad de Barcelona

Arthur Font

Cristian Rodríguez

Proyecto de Prácticas

Programación II

Práctica 4 – 02/06/2019

Barcelona

2019

Index

1. Introducción
2. Análisis
3. Desarrollo
4. Respuestas
5. Resultados

1. Introducción

El objetivo de la práctica es hacer una interfície gráfica para el programa de administración de archivos mp3 y mp4 que hemos realizado en las anteriores prácticas, usando el entorno de desarrollo integrado Netbeans. Ademas hemos tenido que hacer pequeñas modificaciones en el código para que este funcionara correctamente.

2. Análisis

Planteamos el problema de crear la interfície gráfica de tal forma que solo tuvieramos que adaptar el menú anterior a interfície. Es decir, usar el menú anterior como modelo, siempre pensando en la comodidad del usuario. Nuestra aplicación tiene como base una ventana principal que contiene tanto la gestión de archivos y álbumes como la gestión de la reproducciń de archivos, siempre enseñando al usuario la biblioteca y el álbum que esta usando en el momento. Para añadir un archivo o álbum utilizamos una ventana emergente para que no interfiera en la ventana principal, haciendolo mas intuitivo y elegante.

3. Desarrollo

Hemos usado las clases ya definidas que nos proporciona Netbeans para hacer la interfície, como por ejemplo Jframe, Jbutton, Jlist, Jpanel, JcomboBox, etc.

Para conectar el código y la interíce gráfica hemos creado la clase AplicacioUB4 la cual hereda de la clase Jframe y contiene un atributo de la clase Controlador.

Para gestionar la conexión que hay entre la parte visual del programa y la parte de los datos primero trabajamos en la modificación de los datos y despues en la modificación visual.

En el caso de querer añadir un archivo creamos una ventana emergente creando la clase FrmAfegirFitxerMultimedia que hereda de Jdialog, esta clase te pide diferente información del archivo que quieres añadir en función de si es un archivo mp3 o mp4, también puedes escoger el arcivo de forma buscandolo en tu propio ordenador.

Para mostrar la biblioteca hemos utilizado un Jlist de tipo String que va añadiendo o eliminando los nombres de los archivos en función de lo que haga el usuario, para que la Jlist se actualizara correctamente hemos usado un

atributo de parámetro de en su constructor de tipo DefaultListModel que es lo que realmente vamos actualizando. La Jlist tiene la funcionalidad de seleccionar uno o más archivos para gestionarlos con el menú de la interfície. Con los albumes hemos utilizado el mismo método que con la biblioteca pero mostrando el album que el usuario haya escogido en un JcomboBox. Para actulizar el JcomboBox hemos implementado la opción del menú de crear álbum, esta funciona creando una ventana emergente al igual que al añadir un archivo a la biblioteca pero con la diferencia de que este utiliza un atributo de tipo JoptionPane, que sirve solo para pedir el nombre que el usuari le quiere poner al álbum.

Todos las demas opciones del menú a excepción de las opciones que empiezan la reproducción funcionan como un simple botón para hacer su trabajo pero en función de los archivos seleccionados en la biblioteca, o en el álbum en su defecto.

Las opciones que empiezan la reproducción funcionan de dos maneras, las que gestionan solo un archivo, que reproduce solo el archivo seleccionado en la biblioteca o álbum, respectivamente, y las que gestionan una lista de archivos, que reproduce directamente la biblioteca o el álbum seleccionado, respectivamente.

Finalmente, debajo de la interfície esta situado el menú de control del reproductor que esta compuesto por botones, además añadimos a este menú el botón de “Atura” que cierra el reproductor.

4. Respuestas

1. Hemos podido reutilizar todas las clases del paquete model y controlador. Del paquete vista solo utilizamos la clase IniciadorAplicacióUB.

Hemos realizado las siguientes modificaciones:

En la clase Dades: hemos añadido los métodos getSize() y getSizeAlbum() para conseguir el tamaño de la biblioteca o de un álbum específico.

En la clase Controlador: añadimos los métodos getSize() y getSizeAlbum() para conectar model y vista.

En la clase CarpetaFitxer y AlbumFitxersMultimedia: hemos adaptado los métodos toString para que funcione corectamente.

2. Es un ActionEvent, funciona detectando si el usuario lo selecciona usando el click izquierdo o la tecla Space del teclado.

3. Hemos usado ActionEvent para todas las funciones del menú excepto la función del botón Play/Pause en la que hemos usado ItemEvent, actua como palanca, es decir, tiene dos estados, en uno pausa la repdroducción y en el otro la reemprende.

4. Comprobamos las funciones de la interfície gráfica y todas actuaron como deberian.

5. Aprendimos lo básico de la implementación gráfico pero nos hemos dado cuenta de que es un campo muy extenso en el que se pueden hacer cosas interesantes y más complejas.

5. Resultados

Como ya hemos dicho en el apartado 4.4 todas las funciones tuvieron éxito y el programa fluye correctamente.