Astor Prieto Dehghan Pour (aprietde10)

Programació II

10 – 03 – 2014

Lliurament 1, ReproductorUB1

Index

Introducció

L'objectiu d'aquesta practica es organitzar i mostrar les cançons dins de la biblioteca musical. Cada canço es represantada per un fitxer que la guarda amb el seu respectiu "*path*", i les dades propies d'una canço:

· Nom de la cançó
· Autor
· Disc
· Any d'edició
· Número de cançó dins del disc
· Discogràfica
· Duració de la cançó

Aquesta biblioteca musical ha de ser una llista de fitxers de audio.

Juntament amb aquesta practica, s'ha de implementar un menu per a interactuar amb la biblioteca musical, que inclou les següents opcions:

1. **Afegir fitxer:** Demanarà les dades d'un fitxer i l'afegirà a la llista de fitxers.
2. **Mostrar llista:** Mostra el contingut de la llista de fitxers, mostrant davant de cada fitxer, el nombre de la seva posició a la llista començant per 1.
3. **Eliminar fitxer:** Elimina el fitxer en "una posició determinada" de la llista.
4. **Guardar llista:** Guarda el contingut de la llista en un fitxer.
5. **Recuperar llista:** Carrega una llista prèviament guardada en un fitxer.

Anàlisi

Clases Implementades

Per a aquesta entrega, i per a poder implementar un menu que gestioni una llista de fitxers de audio, caldrà implementar una Clase FitxerAudio, que hereti els mètodes de File, per a poder definir tota la informació que demana una pista de audio.

També cal definir una Clase TaulaFitxers, que juntament amb els seus mètodes, ens servirà per a gestionar i modificar la taula de fitxers, es a dir, la nostra biblioteca musical. Ja que la practica esta dividida en dos reproductor diferents (UB1A, UB1B), aquesta clase només la farem servir per a la opció A, ja que a la opció B la llista de fitxers s'implementarà amb ArrayList, que ja conté els seus mètodes per a modificar la biblioteca musical.

Posteriorment, s'haurà de implementar dos menus amb les cinc opcions de la interficie de usuari, un per a la opció A, que crida als mètodes de TaulaFitxers, i un per a la opció B, amb els mètodes propis de ArrayList.

Desenvolupament

Canvis entre A i B

Inicialment, el ReproductorUB1A ha sigut implementat com una llista de arxius del tipus FitxerAudio, donant a la taula una mida estandard de 100 posicions com a màxim. Ja que al ReproductorUB1B la taula es implementada com un ArrayList, ja de entrada no cal presuposar que la taula té una mida finita.

El menú de la opció A, tot implementat amb aquesta taula limitada, fa servir els mètodes implementats a la pròpia clase TaulaFitxers (tamany(), afegirFitxer(), eliminarFitxer(), netejaTaula()...), a diferència de la opció B, que ja que es implementada amb ArrayList, fà servir els mètodes d'aquesta (add(), remove()...).

Per a implementar les opcions 4 i 5 dels menus, s'han creat dos mètodes a la clase TaulaFitxers per a la opció A, i dos mètodes a la clase del menu per a la opció B (guardarLlista(), recuperarLlista()).

Exemples A i B

Tot seguint el format donat al lliurament de la pràctica, la llista de cançons s'ha de mostrar de la mateixa forma per a cada opcio. Ha de seguir el seguent format:

```
[1] | Titol: By the way | Autor: Red Hot Chili Peppers | Disc: By the way | Pista: 1 | Discogràfica: Sony | Duració: 3.37 | Any: 2002 | Fitxer: <C:\canco1.mp3> |

[2] | Titol: Universally Speaking | Autor: Red Hot Chili Peppers | Disc: By the way | Pista: 2 | Discogràfica: Sony | Duració: 4.19 | Any: 2002 | Fitxer: <C:\canco2.mp3> |
```

A la part A, per a donar aquest format a la llista de fitxers, s'ha implementat un metode toString() ón es concatenen tots els elements de cada pista separats amb “|” i amb un comptador que els asigna la seva posicio a la biblioteca musical.
A la part B, degut a que cada pista es un element sencer del ArrayList, s'ha implementat un element iterador que comproba si hi han elements en la taula i els va imprimint mentre quedin fitxers a la llista (hasNext(), next()...).

Problemes i Proves

Q: segons la implementació de la part B, si tenim dos fitxers d’àudio corresponents al mateix fitxer, quan cridem al mètode per eliminar un d’aquests fitxers eliminarà el altre també o no?

A: No, degut a que, gràcies al ArrayList, el que elimina es un objecte asociat a una posició de la llista, i no el “*path”* al que apunta el fitxer.

El principal problema ha sigut la implementació de les opcions 4 i 5 del menu per la part A i la posterior adaptació per a la part B.
Els metodes implementats (guardarLlista() i recuperarLlista()), a la part A, fan ús dels ObjectOutputStreams per a guardar la llista sencera codificada i despres poderla recuperar amb un ObjectInputStream, que estan referint-se a un objecte del tipus TaulaFitxers, cosa que el fa incompatible amb el ArrayList<FitxerAudio>. D'aquesta forma, també s'han hagut de implementar dos mètodes més (guardarLlista() i recuperarLlista()), que es refereixin a un altre fitxer “.dat” (sent el primer “listaReproUB1A.dat” i el segón “listaReproUB1B.dat”).

Resultats

Ja que les tres primeres opcions del menú són fàcilment testejables (Afegir Fitxer, Mostrar Llista i Eliminar Fitxer), s'ha comprovat si el mètode toString() seguia el format adecuat demanat al lliurament, i si els mètodes afegiFitxer() i eliminarFitxer() afegia fitxers i eliminava fitxers de la llista sense deixar forats buits enmig i afegia fitxers al final de la llista.

Posteriorment, i amb les opcions 4 i 5 implementades, s'ha comprovat que importar i exportar fitxers "*.dat*" (en el root del projecte de NetBeans) no donava problemes i el objecte es recuperava correctament, mostrant la taula amb el seu correcte format.
Això és possible degut a que el ObjectsStreams codifiquen en bytes els objectes amb tots els seus paràmetres, fent que sigui de fàcil utilització i portabilitat.

Finalment, comparant i trobant errors a les diferents parts del A i B, s'ha comprovat que els seus fitxers "*.dat*" no son compatibles entre sí, ja que es refereixen a objectes de diferents tipus (un a TaulaFitxers i altre a ArrayList<FitxerAudio>). De forma que ens ha forçat a implementar un parell de funcions de importació i exportació de la llista diferents per a cada part. També s'ha pogut veure si al recuperar una llista des de un arxiu, la taula recuperada era modificable al programa i si es podia exportar novament.

En resum, la opció A és un programa en el que s'ha hagut de implementar més mètodes i clases per al seu correcte funcionament, mentre que a la opció B, amb el menú i la majoria de mètodes per a modificar la taula implementats, necesita de crear menys mètodes i clases, ja que aprofita tot el codi implementat a la clase ArrayList de la API de Java.