Astor Prieto DehghanPour (aprietde10)

Programació II

28 – 04 – 2014

Lliurament 3, ReproductorUB3

Índex

Introducció

En aquesta tercera entrega, l'objectiu era ampliar i millorar el reproductorUB2, afegint funcionalitats per a reproduïr fitxers i llistes per una banda, i gestionant possibles errors amb excepcions pròpies per una altra. Per a dur a terme aquesta tasca, s'ha d'afegir al controlador del reproductor una classe que ens permeti reproduïr pistes una a una, la biblioteca sencera, o be una llista de reproducció determinada.

Anàlisi

Classes Implementades

Per a aquesta entrega, i fent ús de la biblioteca *UtilsProg2.tar*, hem creat aquest nova classe del controlador que conté tots els mètodes necessàris per a reproduir de diferents formes els fitxers de audio. Aquesta classe anomenada ReproductorAudio, és una classe que hereta de ReproductorBasic i controla tant la reproducció de arxius com les opcions de reproducció possibles. Dins la classe tenim uns “flags” de estat (rCiclic i rAleatoria) que ens determinen si les opcions de reproducció ciclica i reproducció aleatòria estan activades o no, respectivament. També trobem en aquesta classe un vector de booleans que es crea amb la excecució dels mètodes per a reproduïr llistes (ja sigui la biblioteca o una LlistaReproduccio) i que té la mateixa mida que la llista en qüestió que serveix per a comprovar si una pista de la llista ja ha sigut reproduída o no.
Per altra banda, les altres classes que han sigut implementades foren les excepcions pròpies. Les classes ExcepcioFitxerRepetit i ExcepcioFitxerNoExisteix, hereten de FitxerAudioErrorException i es defineixen com classes filles de la superclasse “Exception”. La primera excepció pròpia, ExcepcioFitxerRepetit, salta quan s'intenta introduïr un fitxer ja existent a la biblioteca i retorna el missatge “Fitxer de audio ja existent.”. La segona, ExcepcioFitxerNoExisteix, comprova, mitjançant la funció exists(), si el fitxer que volem introduïr existeix al nostre disc dur o no, retornant en cas negatiu el missatge “No existeix el fitxer al disc.”.

Codi Reutilitzat

Com en aquesta entrega es reutilitza el codi de la entrega segona, mes aviat explicarem les modificacions que han sigut realitzades al codi previ.
Després de adaptar la vista i el menú per a mostrar les opcions noves necessàries (3. Control Reproducció), la classe que més modificacions ha sofert ha sigut la classe CtrlReproductor, que ha sigut adaptada, incloent-hi les crides a tots els mètodes de ReproductorAudio i modificacions als mètodes existents, com guardarDades() i recuperarDades(), que han canviat per a que ara també guardin i recuperin el objecte de tipus ReproductorAudio que hem creat al controlador. Ara en comptes de guardar i recuperar únicament el objecte de tipus DadesReproductor, ara guarda un sol objecte de tipus CtrlReproductor, que conté al seu interior dos objectes mes (un DadesReproductor i un ReproductorAudio). De aquesta forma no solsament ens asegurem de guardar la biblioteca i les llistes de reproducció, sino que a sobre estem guardant la informació de les opcions de reproducció (el aleatori i el ciclic).

Desenvolupament

Proves Realitzades

En aquesta entrega, i ja que és la primera entrega que reprodueix fitxers de audio, s'han realitzat un conjunt de proves per a guarantir el correcte funcionament de cada opció de reproducció. Per tal de que el reproductor reproduís llistes, s'ha hagut de sobreescriure el mètode de event onEndFile usant el mètode de ReproductorAudio, seguent(), per a que al acabar una cançó, es crides automàticament a la següent, ja que d'una altra forma, les cançons es reprodueixen alhora i no una darrera l'altra. És a dir, quan es crida al metode playFilesList(), es reprodueix la primera canço i al acabar, amb el event onEndFile(), s'executa el mètode seguent(), que comprova els *flags* de reproducció i cirda i executa el metode playAudioFile() amb una pista o altra en consequència.

Una altra prova realitzada ha sigut la reproducció usant els *flags* ciclic i aleatòri, que per diferents combinacions de activació, el mètode seguent() ha de comportar-se de una forma o de altra. Per a resoldre el mode aleatori, hem creat una variable “index” que en cas de estar desactivat es incrementa en una unitat, però en cas de estar activat el aleatori, aquest index s'obté mitjançant una funció random entre el 0 i el length de la llista a reproduïr. D'altra forma, per a resoldre el mode ciclic, es comprova la taula de booleans de la mida de la llista, i reproducció rere reproducció, es comprova si tota la taula conté “true” a cada posició, en cas afirmatiu, la taula es reseteja amb “false” a totes les posicions, i es reinicia la reproducció per a donar el efecte de ciclic.

Resultats

Finalment, en acabar aquesta entrega i corretgir els errors i excepcions, s'observen els seguents objectius assolits:

- S'ha creat un reproductor que reprodueix musica i llistes de musica.
- S'ha après a crear i gestionar excepcions pròpies i creades per nosaltres.
- S'ha vist el que significa programació orientada a events, i se n'ha fet ús de events.