MEMORIA DEL LLIURAMENT 1

PROGRAMACIO 2

DATA DEL LLIURAMENT: 7-3-2013

NOM: VICTOR GOMEZ FARRUS

GRUP: B, DNI: 49184353 A

INDEX:

Introducció

Anàlisi(mòduls i funcionalitat)

Desenvolupament

Resultats

INTRODUCCIO:

L’objectiu principal és desenvolupar un software mitjançant programació orientada a objectes i esdeveniments.

El software serà un visor d’imatges, el qual faré guiat per la nostra professora, Laura Igual.

Aquesta aplicació contindrà una biblioteca d’imatges i permetrà definir àlbums d’imatges, així com gestionar-los i mostrar-los amb la possibilitat de guardar i recuperar les nostres llistes.

En el lliurament 1 de moment només hem d’implementar un menú que permeti gestionar fitxers dins una llista.

Aplicaré els conceptes d’algorísmica i programació del semestre passat per poder sortir-me’n.

ANALISI:

Per tal d’assolir els objectius, el programa estarà dividit en tres mòduls: vista, model i controlador. A continuació entraré en detalls:

**VISTA:**

Conté les classes principals i les que permetran una relació interactiva amb l’usuari. És la manera que tenim de veure les funcionalitats i dades de la nostra aplicació.

**MODEL:**

Conté les classes que serviran per manejar les dades, és a dir, les que contindran les estructures necessàries per poder gestionar la biblioteca d’imatges.

**CONTROLADOR:**

Conté les classes que controlen l’execució i les encarregades de visualitzar les imatges, etc.

DESENVOLUPAMENT:

La universitat em proporciona un arxiu .jar que conté unes classes que em seran d’utilitat. Les aniré mencionant a mesura que les utilitzi.

Dividiré el meu desenvolupament en els mòduls dels que he parlat a l’anàlisi. Ressaltaré els elements de la manera següent:

***Mètodes Tipus de variable Atribut de classe Classe***

**VISTA:**

**#VisorUB1:**

En el lliurament 1 de pràctiques, serà la classe principal(conté el main). Aquesta classe proporciona un menú via consola, el qual permet les 6 opcions demanades a l’enunciat.

Els atributs d’aquesta classe seran tots privats:

-***enum OpcionsMenuVisorUB1***: És el **enum** corresponent al menú principal. És un atribut estàtic.

-***ImageList imageList***: Serà la llista de imatges amb la que tractarem.

- ***String[]*** ***descMenuVisorUB***: Cada part de l’array correspon a una opció del menú principal. És un atribut estàtic.

-***Menu<OpcionsMenuVisorUB1>*** ***menu***: Utilitzant ***Menu*** proporcionada al .jar, serveix per crear un menú.

Els mètodes d’aquesta classe seran tots privats:

***-void gestioVisorUB(Scanner sc):*** Permetrà gestionar per consola el programa començant pel menú. Ho de manera interactiva amb l’usuari gràcies al paràmetre tipus ***Scanner***. Mentre no triem l’opció sortir, el programa es mantindrà en funcionament. Cada opció cridarà els següents mètodes segons convingui:

- ***void addImage(Scanner sc)*** : Serveix per afegir una imatge a la llista.

- ***void showList()*** : Serveix per mostrar la llista

- ***void deleteImg(Scanner sc)*** : Serveix per eliminar una imatge ubicada en un lloc introduït per ***Scanner*** de la llista.

- ***void saveList(Scanner sc)*** : Serveix per guardar la llista en un arxiu, mitjançant “Streaming”, ubicat en un lloc introduït per ***Scanner***. Conté “try-catch-finally”, necessari per gestionar el “Streaming”.

- ***void retrList(Scanner sc)*** : Serveix per recuperar una llista en un arxiu, mitjançant “Streaming”, ubicat en un lloc introduït per ***Scanner***. Conté “try-catch-finally”, necessari per gestionar el “Streaming”.

- ***void clear()*** : Serveix per posar a null la llista.

**(·)** El **main** inicialitza un ***Scanner*** i un ***VisorUB1***, i utilitza el mètode ***gestioVisorUB*** amb els paràmetres inicialitzats.

**(·)** El **constructor** d’aquesta classe inicialitza ***menu*** i li assigna com a descripció ***descMenuVisorUB****.* Ens fa escollir entre mode d’implementació(Llista o taula), encara que els dos siguin compatibles. Més endavant aprofundirem en els modes.

**MODEL:**

**#Imatge:**

Heretarà la classe abstracta ***ImageFIle***. S’ha d’aclarir que s’entén per ruta completa d’un arxiu com: (ex: “C:\Users\VICTOR\Pictures\Sample Pictures\Koala.jpg”).

Els atributs d’aquesta classe seran tots privats:

*-****File******arxImatge***: Correspondrà a l’arxiu que manipularem.

A continuació tenim els mètodes abstractes(i també públics) de ***ImageFile***:

***- String getFullPath()***: Serveix per obtenir la ruta completa de l’imatge.

***-String getExtension()***: Serveix per obtenir l’extensió(ex: “.jpg”). En el lliurament 1 aquest mètode no s’utilitza.

**(·)** El **constructor** d’aquesta classe té com a paràmetre un ***Scanner****,* mitjançant el qual posem la ruta completa de l’imatge en concret i la guardem dins ***arxImatge***.

**#TaulaImatges:**

Heretarà la classe abstracta ***ImageList*** i serà ***Serializable***, ja que la funció d’aquesta classe serà gestionar arxius d’imatges i necessitarem “serialització”.Servirà per gestionar les imatges en mode ***Imatge[]***. Quan escollim una opció dins el menú principal, estarem cridant un mètode d’aquesta classe.

Els atributs d’aquesta classe seran tots privats:

- ***Imatge[]***  ***taula***: Contindrà totes les imatges que vulguem afegir.

- ***int***  ***size***: Serà el número d’imatges afegides.

- ***int***  ***maxtamany***: Serà el màxim nombre d’imatges que puguem afegir.

A continuació tenim els mètodes abstractes de ***ImageList***:

NOTA: Són tots públics excepte ***isFull()***, que és protegit.

- ***int getSize()***: Serveix per obtenir el la grandària de la ***taula***, és a dir, ***size***.

- ***void addImage(ImageFile i)*** : Serveix per afegir un element a ***taula***, incrementant ***size*** en 1. Farem un cast de ***ImageFile*** a ***Imatge***.

- ***void removeImage(ImageFile i)*** : Serveix per eliminar un element de ***taula***, disminuint ***size*** en 1.

- ***ImageFile getAt(int i)*** : Serveix per obtenir un ***ImageFile*** fent un cast de l’element de ***taula*** corresponent al paràmetre.

- ***boolean isFull()***: Serveix per comprovar si no s’ha excedit ***maxtamany***.

**(·)** El **constructor** d’aquesta classe inicialitzarà una ***taula*** de grandària ***maxtamany*** i posarà ***size*** a 0.

**#LlistaImatges(i reutilització del punt 2.3.2 per 2.6):**

Serà absolutament el mateix que ***TaulaImatges***, però implementat amb ***ArrayList <Imatge>*** de Java en comptes de ***Imatge[].***Conté alguna modificació:

-No necessitem tenir com a atribut ***size***, perquè la nova implementació, es pot fer de manera dinàmica(sense determinar la grandària de ***ArrayList <Imatge>***) .

- No inicialitzarem al **constructor** amb una grandària ***maxtamany*** .

- El mètode ***getSize()*** seguirà existint ja que és abstracte, llavors aquest només farà que cridar el ***size()*** de ***ArrayList***.

-Dins els mètodes on s’ha d’iterar, en comptes de fer-ho amb “for”, hem utilitzat un ***ListIterator*** de Java, ja que és compatible amb ***ArrayList***.

**CONTROLADOR:**

En el lliurament 1 no ens han dit d’implementar res dins aquest mòdul.

RESULTATS:

En el lliurament 1 hem dissenyat i implementat una aplicació amb un menú que ens permet gestionar fitxers dins una llista.

**NOTA IMPORTANT(Referent a proves per comprovar el funcionament correcte):** Donat a una ocurrència d’última hora després d’haver parlat amb un professor de programació, m’he adonat que ***TaulaImatges*** , ***LlistaImatges*** i ***Imatge*** només haurien de tenir els mètodes abstractes. He corregit la memòria i el lliurament. Ara mateix, en el programa i a la memòria tot està implementat de la forma que se m’ha donat a entendre(o he cregut entendre).