Memòria #4

INTRODUCCIÓ I DESENVOLUPAMENT

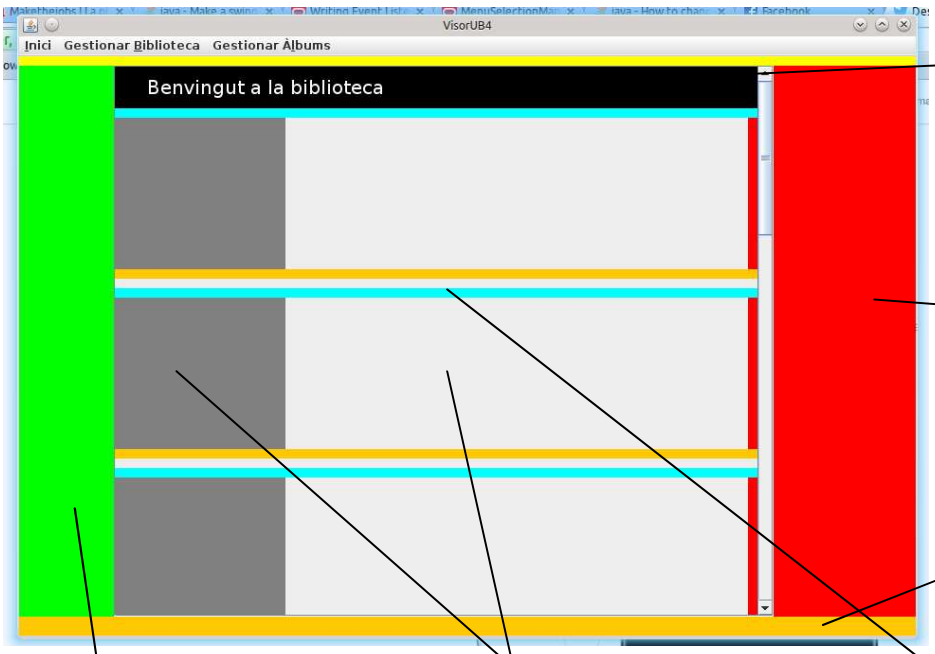
Per dur a terme el VisorUB4 no he utilitzat les ajudes del Netbeans d'arrossegar components. He preferit tenir el 100% de control sobre tot el codi i dissenyar i establir per mi mateix tot el potencial de Swing i la creació d'espais i zones a partir dels diferents tipus de Layouts.

He dissenyat el visor en Displays (pantalles). Cada display és una funcionalitat diferent del visor, per exemple, en el display biblioteca es poden realitzar totes les accions sobre la biblioteca. I així per la resta.

Cada display està configurat per un BorderLayout amb 5 zones (Nord, sud, centre, est i oest).

Aniré mostrant l'evolució de display biblioteca com a exemple de com he anat desenvolupant el visor.

La primera forma de disseny que vaig obtenir és la que mostro seguidament. D'entrada per separar bé cada zona i cada funcionalitat utilitzo colors diferents per marcar bé el disseny:



Zona nord: només contindrà el títol del display actual.

Zona est: contindrà els botons relacionats amb el seu display actual. De momento la pinto vermella.

Zona sud: contindrà el nom del creador. Jo.

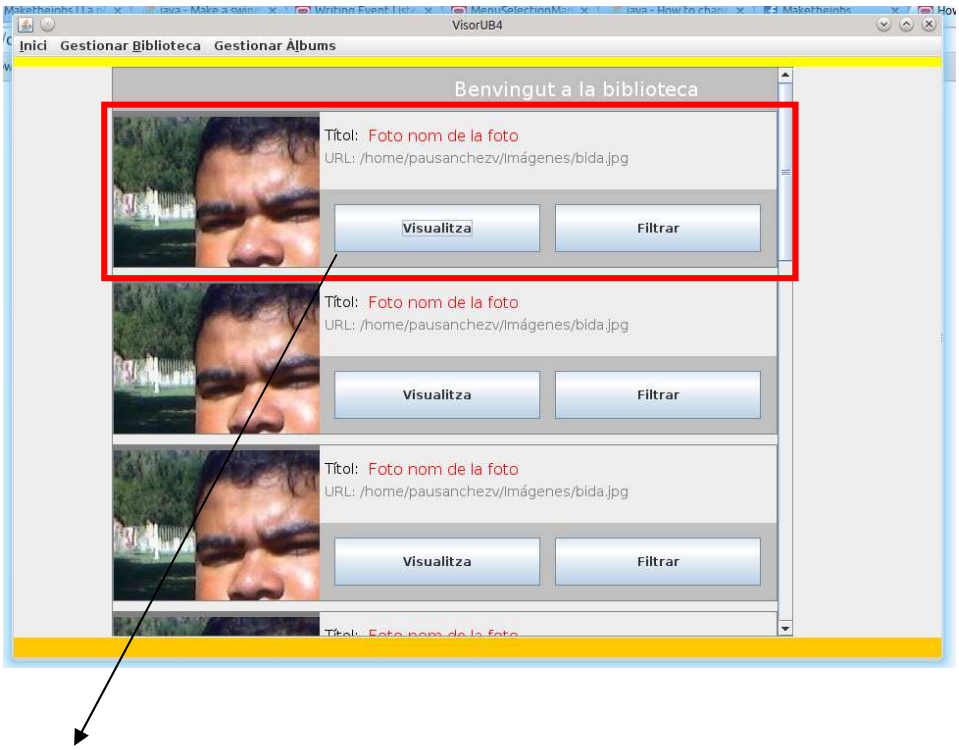
Zona oest: no contindrà res, la utilitzo de simple marge.

Box (Dins la zona centre): és un BorderLayout del qual només aprofito la zona est (contindrà la imatge en miniatura) i la zona centre (zona elástica que s'adapta al frame) que contindrà 2 botons i la info de la imatge.

Zona centre: és un BoxLayout Y_AXIS on cada box representa una imatge.

Aquest disseny és aprofitable per 4 Displays diferents:

1. La biblioteca → Mostrarà les imatges actuals i permetrà interaccioar amb elles
2. Un àlbum → Mostrarà el títol de l'àlbum a la zona nord i totes les seves imatges a la zona central.
3. La llista dels àlbums → Cada box serà un àlbum i mostrarà la seva imatge de portada.
4. Afegir una imatge de la biblioteca a un àlbum → Mostrarà la biblioteca per la única acció que permetrà fer sobre una imatge serà afegir-la a l'àlbum des del qual hem fet la crida.

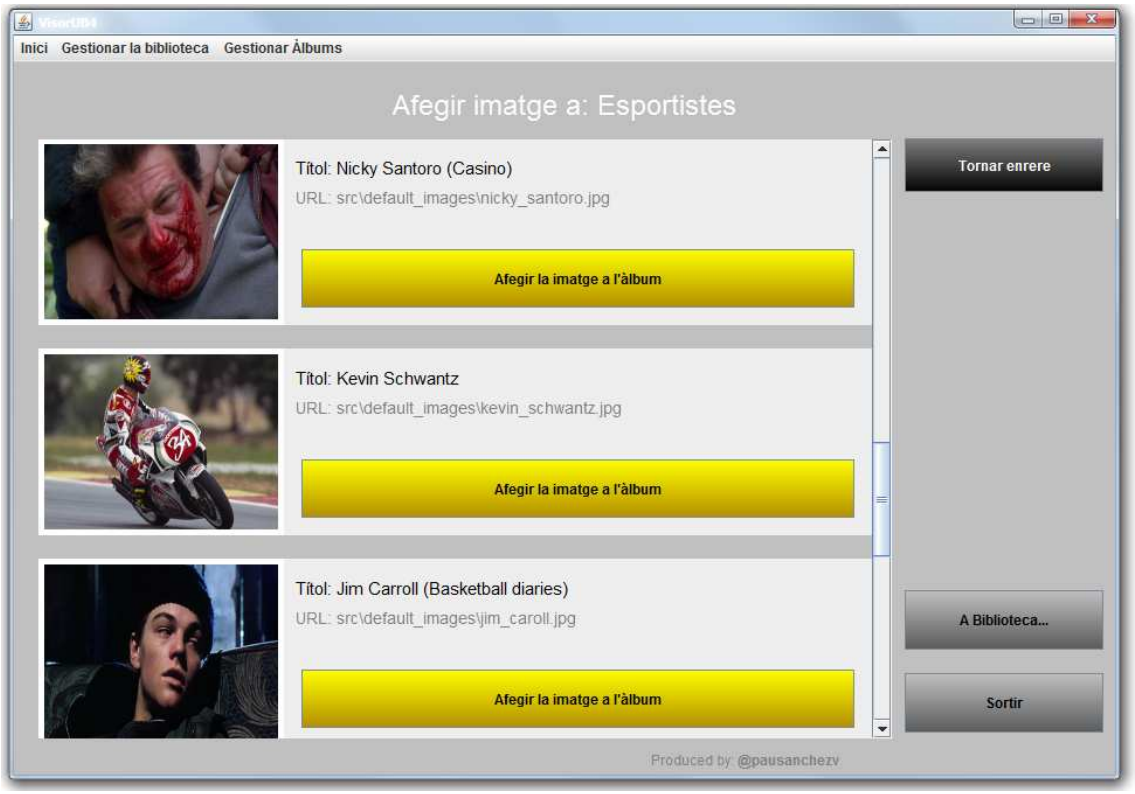


Cada BOX és un BorderLayout on la seva zona est representa la imatge i la seva zona central representa la info i opcions de la imatge. A la vegada la zona central és un GridLayout (matriu) formada per dues files i una columna. La fila superior del grid és un FLowLayout que conté el títol de la imatge i la seva URL. La fila inferior és a la vegada un nou GridLayout on d'una fila i dues columnes on cada columna conté un botó. Tot es basa en JPanel dins de JPanel.

Com es pot apreciar, al principi, no aconseguia adaptar el tamany de la foto al tamany del JPanel, doncs la foto original és així:

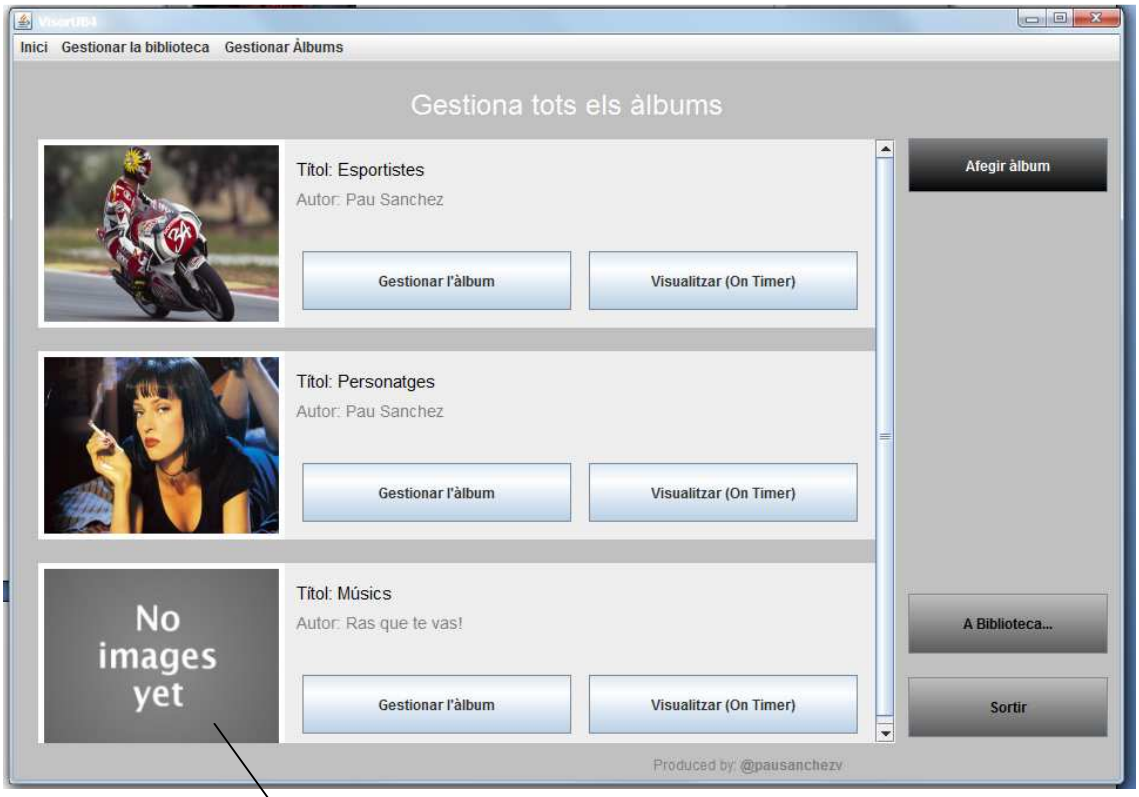
Evolució definitiva del disseny:

Ara veiem què passa quan d'es d'un àlbum cridem a la biblioteca per afegir una imatge a l'àlbum:



Totes les opcions han marxat i només ens permet introduir una imatge a l'àlbum des del qual hem fet la crida, en aquest cas l'àlbum "Esportistes".

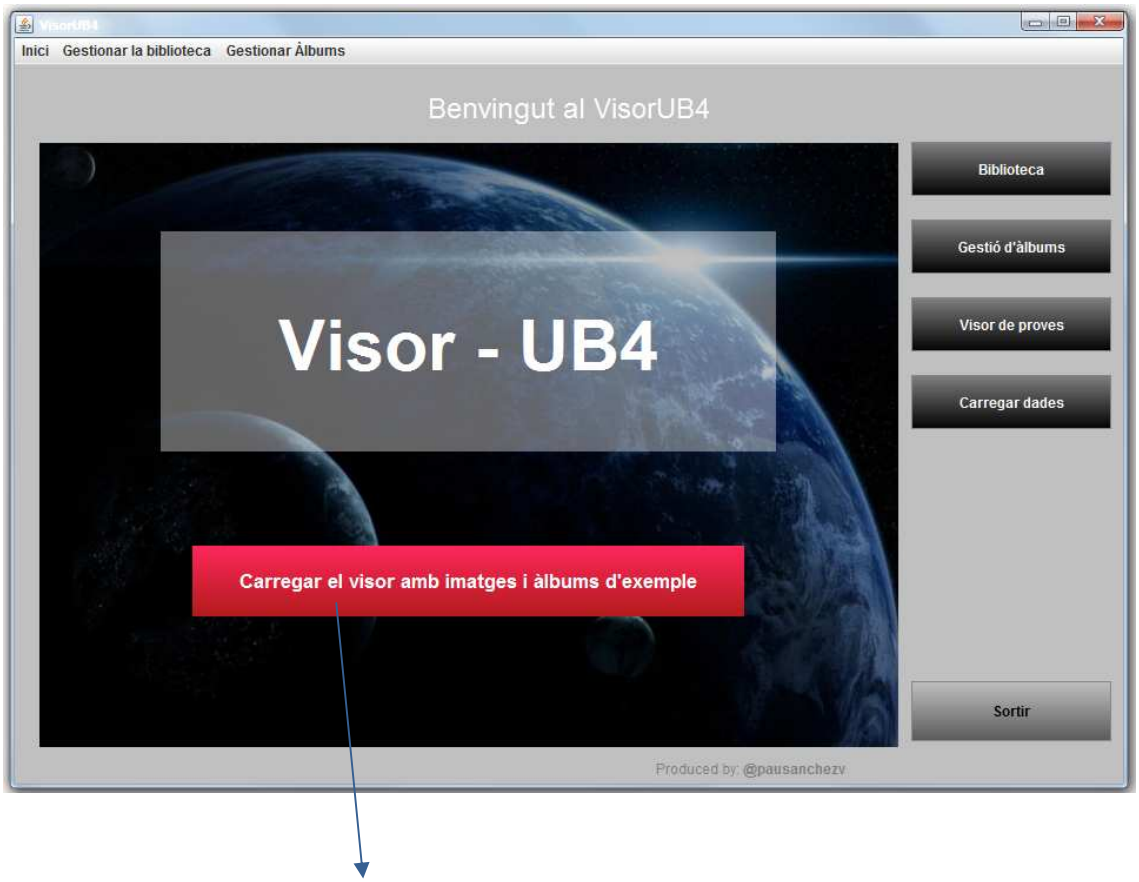
Veiem per últim com es mostra el display dedicat als àlbums i què passa quan un àlbum encara està buit.



Podem apreciar que si un àlbum encara no té imatges i la seva portada és NULL mostra una imatge per defecte.

Quan transformem una imatge de qualsevol dels àlbums o de la biblioteca, l'efecte de la transformació es veu reflectit immediatament en la miniatura de la imatge sense necessitat d'haver d'esperar a visualitzar-la mitjançant un show.

Per últim he preparat un visor d'exemple i una opció per carregar-lo en la mateixa pantalla de benvinguda. Quan carregues aquest visor d'exemple, l'aplicació s'omple amb 14 imatges i 3 àlbums evitant així haver de crear i introduir imatges i àlbums en el visor per pode evaluar-lo i posar-lo en marxa.



Aquest és el botó que carrega el visor de proves de manera automàtica. De vegades, segons la qualitat de la computadora, triga uns segons a carregar-lo!

PREGUNTES A RESPONDRE

1. Introduïu el problema tractat en el lliurament (No s'acceptarà una còpia directa de l'enunciat).

Es demana programar una interfície gràfica tot i continuant la pràctica anterior. El desenvolupament del meu visor ha estat el descrit a l'apartat anterior. En el meu cas m'he desviat una mica de les exigències i els noms dels formularis "FrmNomformulari" ja que jo no utilitzo formularis d'aquests per al meu visor i he programat tot jo des de zero.

Bàsicament es demanen totes les funcionalitats de l'exercici visor 3 però ara tot fet amb events i intrfície gràfica.

A més es demana l'opció d'augmentar i disminuir la velocitat de visualització amb l'onTimer.

2. Expliqueu les classes implementades.

2.1 - Classes ja implementades en pràctiques anteriors

Nota: Ha desaparescut la classe <GestióVisorUB>. Ara la classe que conté el main() és la classe ViorUB4 . El mètode main es troba dins aquesta classe i crida al mètode run per posar en marxa l'aplicació.

<Imatge> : Representa un obj Imatge i hereda de <ImageFile> que a la vegada hereda de File.

<Llistaimatges> : És la classe base per gestionar les llistes d'imatges, hereda de la classe abstracta <ImageList> i ens imposa el diseny de l'interface <InImageList>.

<AlbumImatges> : Un obj AlbumImatges i és fill descendent de <LlistaImatges> ja que un àlbum és una llista d'imatges.

<BibliotecaImatges> : És filla descendent de <LlistaImatges>, fa servir algunes de les funcions "mare" i sobreescriu les que són necessaries tot i fent ús del polimorfismse.

<DadesVisor> : És la classe on es crea l'array dels àlbums (jo l'hagués creat en una classe a part però un disseny imposat és un disseny imposat). Aquesta classe, a més, conté totes les dades del visor. Des d'aqui, a més, és d'es d'on guardem i recuperem les dades ja que l'objecte serialitzable que guardem/recuperem és precisament un obj <DadesVisor>. Aquesta classe doncs és la que portarà implements Serializable.

<CtrlVisor> : Exerceix d'intermediari entre el model i la vista. Crea un obj <DadesVisor> i el fa servir per relacionar-se amb les llistes d'imatges i la llista dels àlbums. Aquesta classe és cridada a través d'un obj per la vista.

<VisorUB4> : Classe que conté el main() i que posa en marxa el VisorUB4.

<ImatgeSepia>: És filla directa d'Imatge. Hereda tots els mètodes d'Imatge i té creats 3 mètodes propis que serviran per gestionar la classe:

- color2Sepia() → Mètode que aplica un filtre sípia sobre la imatge
- save() → Guarda la imatge utilitzant el mètode saveImage() de la classe <ImageFile>. En aquesta classe és un mètode sobreescrit ja que també l'he implementat a Imatge().
- Show() → sobreescriu el show proporcionat per la llibreria utils que ens heu fet implementar. Aquest mètode es limita a cridar al'anterior (save) i després crida al show() de la classe mare <Imatge>, aquest és qui dona tamanys genèrics a la imatge i després crida al show() d'<ImageFile> que ja ensenya la imatge.

<ImatgeBN>: És exactament idèntica que la classe <ImatgeSepia>.

<Filtrable> ***Interface*** : És una interfície creada amb tots els mètodes necessaris per filtrar una imatge o visualitzar-la amb filtre. És un patró imposat per mi el qual han de seguir tots els elements que siguin filtrables. En aquest cas imatges.

2.2 - Classes implementades per al VisorUB4

<Settings>: És la classe que crea les zones i que fa les crides per formar el display que toca segons l'acció de l'usuari sobre la interfície gràfica. Crea el frame principal l'omple amb el display de benvinguda quan obrim el visor. Crea els objs dels 10 displays (pantalles) diferents que conformen el visor i totes les accions es fan sobre els objectes (creats només una vegada per evitar solapaments de pantalla).

<TopBarMenu>: Crea la barra de menú de la part superior del VisorUB4.

<Functions>: És una classe que conté mètodes diversos. Majoritàriament conté tots aquells mètodes de suport que a la pràctica anterior es trobaven a la classe <VisorUB3>. Molts d'aquells mètodes no m'han fet falta i en canvi n'he necessitat de nous. En fi, és una classe de mètodes de suport.

<Display> ***Abstract Class*****:** Aquesta és una classe abstracta mare de tots els displays (pantalles del visor). Està declarada com a abstracta perquè no s'ha de crear cap instància de Display sense saber de quid display es tracta (Biblioteca, àlbums, inici, etc). No conté mètodes abstractes degut a que les 10 classes filles tenen molts mètodes en comú (mateix nom i funcionalitat) però en alguns subclasses reben paràmetres i a les altres no cal, així que no puc imposar un patró fix de disseny. En canvi aquesta classe si que conté mètodes comuns i dels quals es beneficien i poden utilitzar totes les classes filles (estalvi de codi inútil).

<DisplayLibrary> ***extends Display*****:** és el display (pantalla) de la biblioteca. Permet tota la interacció amb la biblioteca i les seves imatges.

<DisplayAlbums> ***extends Display*****:** és el display (pantalla) dels àlbums. Permet tota la interacció amb els àlbums (visualitzar, canviar títol, actualitzar portada...)

<DisplayAlbum> ***extends Display*****:** és el display (pantalla) d'un àlbum. Permet tota la interacció amb l'àlbum i les seves imatges.

<DisplayWelcome> ***extends Display*****:** és el display (pantalla) que dona la benvinguda al VisorUB4 quan posem en marxa l'aplicació.

<DisplayAddImage> ***extends Display*****:** és el display (pantalla) que permet afegir una imatge a la biblioteca.

<DisplayAddAlbum> ***extends Display*****:** és el display (pantalla) que permet afegir un àlbum.

<DisplayAddImageToLibrary> ***extends Display*****:** és el display (pantalla) que permet afegir una imatge de la biblioteca a un àlbum.

<DisplayChangeFront> ***extends Display*****:** és el display (pantalla) que permet actualitzar la portada d'un àlbum.

<DisplayEditTitleAlbum> ***extends Display*****:** és el display (pantalla) que permet actualitzar el títol de l'àlbum.

<DisplayVisualize> *extends Display*: és el display (pantalla) que permet visualitzar una llista d'imatges ja sigui la biblioteca o un àlbum.

<GradientButton> Accessoris package: és una classe on he fet servir el paintComponent() explicat l'últim dia de classe per pintar els botons a dos colors amb efecte degradat.

3. Expliqueu quines classes has pogut reutilitzar del primer lliurament per a fer aquest. Quins canvis sobre les classes reutilitzades heu necessitat fer i perquè.

Totes les classes de la pràctica anterior han estat lògicament reutilitzades i continuen fent servei menys una. He destruït la classe <GestióVisorUB> que contenia el main() i ara he passat aquest mètode a la classe <VisorUB4> que conté el main() i el mètode run().

Majoritàriament les modificacions de les classes anteriors han estat al controlador. Com que abans treballàvem amb menús, l'usuari entrava un enter a través de la consola i nosaltres enviàvem aquest enter al controlador i allà es creava l'obj a partir d'aquest enter que era la posició dins l'array. Ara treballant amb botons i events podem prescindir de la petició d'aquest enter i un click sobre un determinat obj ja ens pot retornar tota la seva informació.

4. Expliqueu quin és el model de delegació d'events que es fa servir en el botó de visualització de la biblioteca.

Els events que es desencadenen quan volem visualitzar la biblioteca són els següents:

1. L'usuari està al display <Biblioteca> interactuant amb la biblioteca quan de sobte decideix prémer el botó de visualització.
2. L'actionPerformed crida al mètode de la classe <Functions> slideImages() el qual rep una llista com a paràmetre, en aquest cas, rep la llista biblioteca, però podria ser un àlbum si estiguéssim en el display d'un àlbum.
3. Suposem el pitjor cas: No hi ha imatges! No passa res pq salta l'excepció i ens obre un diàleg dient que s'ha produït un error. No passa res, afegim imatges i ja està.
4. Ara suposem que si que hi ha imatges: el try de slideImages() activa el play() del controlador i comença l'espectacle, (el catch ja l'he explicat)
5. Una vegada al controlador estem dins el mètode play i aquest mostra la primera imatge i activa l'onTimer()
6. L'onTimer() va mostrant les imatges i quan acaba torna al principi de la llista.
7. Com es pot observar si estem visualitzant la biblioteca i engrandim la finestra s'aprecia que les imatges s'adapten a la nova mida de pantalla oferint un efecte "responsive" i reajustant automàticament el seu tamany.

5. Indiqueu quins tipus d'events heu fet servir al vostre codi.

He fet servir sobretot el actionPerformed que és escoltat per les classes internes anònimes heretades d'actionListener. En el meu cas particular tots els esdeveniments del meu codi estan fets amb classes internes anònimes però a mi no me les ha fet el netbeans, sinó que les he fet jo, i he triat fer-les així pq és la manera més còmode i la qual em permet optimitzar millor el codi. Cada botó té un actionPerformed per dur a terme totes les accions que cal fer quan premem sobre el botó. Aquestes accions poden ser, visualitzar una imatge, eliminar una imatge, etc etc etc...

6. A quina classe heu implementat les accions derivades dels controls d'activar, pausar i aturar una reproducció?

Els botons de control estan situats a la zona est de la classe <DisplayVisualize> i els events d'aquests botons criden a mètodes del controlador. Llavors tots els mètodes de control de la visualització d'imatges (play, pausa, stop, incrementa velocitat, decrementa ... etc...) estan al controlador).

7. Proves realitzades per comprovar el correcte funcionament de la pràctica, resultats obtinguts i accions derivades.

Les proves més dures han estat sens dubte la familiarització amb els layouts (molts d'ells encara em costa d'entendre com es comporten). He trobat a faltar i no puc entendre com no existeix una espècie de CSS per java. Construir estructures ho trobo excessivament difícil i tot és massa rígid. Però m'he divertit construint-lo.

De la part interna com que he arribat fins aquí amb un codi molt sòlid i sense gaires errors, no m'he trobat cap "sorpresa" i tot el que he fet ha estat substituir els enters entrats per l'usuari al triar una opció de l'antic menú a la construcció directe de l'objecte en qüestió per part meva. Per tant han sobrat molts mètodes.

8. Observacions generals.

Aqui és on a les altres entregues et comentava les reparacions en el codi que m'havies proposat en les retroaccions anteriors.

Malauradament, en aquest cas, estic fent aquest informe sense haver rebut la retroacció de la pràctica 3 així que no puc utilitzar aquest apartat amb la mateixa finalitat.