

Programmation orienté objets et événementielle

Projet à rendre pour le (bonne question mais autour du 15 janvier 2010)

Si vous ne recevez pas d'accuser et que la date limite d'envoi est passée, manifester vous rapidement (mon numéro de téléphone est 06 09 74 47 85, cas d'urgences seulement).

L'objectif du projet est de développer un visualiseur d'images type slide show. L'utilisateur pourra ouvrir des images, puis demander l'affichage des images. Les images seront afficher les unes après les autres un certain temps, des transitions seront utiliser pour passer d'une image à l'autre.

La première étape du projet est la lecture du code source qui vous est fourni, comprenez son fonctionnement et familiariser vous avec l'architecture proposée.

Trois packages sont présents : mainProgram, slideShow, slideShowInterface

mainProgram contient une seule classe avec la méthode main

slideShow est le package du modèle, il contient une classe SlideShowImageCollection permettant de stocker en mémoire une liste d'image et de la parcourir. Cette classe utilise un Vector. Référez vous à la documentation en ligne des Vector en Java pour savoir comment fonctionne ce type de classe. La classe SlideShow à pour objectif de calculer l'image à afficher à partir de la listes d'images chargées et d'un temps (représentant le temps depuis le début de la lecture). La méthode update(int time) met à jour l'image mixedImage. La classe ImageAdapter est une classe static boite à outil, elle sert pour créer des méthodes de manipulation d'images. N'hésiter pas à rajouter des méthodes dans cette classe dès que nécessaire

slideShowInterface est le package de la vue, il contient une classe MainWindow qui prépare la fenetre principale, cette fenêtre contient un widget de la classe ImageViewer également défini dans ce package.

De plus les classes Utils et ImageFileFilter (récupérer dans la documentation de Sun) permette le filtrage des fichiers images lors de l'ouverture d'une image (pour n'afficher que les *.extition_de_type_image). A priori vous n'avez pas à modifier ces deux classes.

N'hésitez pas à regarder la javadoc et le site de sun pour comprendre le fonctionnement du code fourni. Certains commentaires dans le code complémente cette documentation.

Travail à faire

D'une manière générale vous devez commenter votre code pour chaque classe, méthode et donnée membre que vous créez : quel est la raison d'être d'une classe, que fait la méthode, que représente une

donnée membre. Par contre l'intérieur d'une méthode ne doit être commenté que si il y a un intérêt réel : exemple de commentaire qui ne sert à rien :

```
// compute sourceImage aspect ratio by divide width/height
    float biAspect =
(float)sourceImage.getWidth()/ (float)sourceImage.getHeight();
// compute output aspect ratio by divide width/height
    float viAspect = (float)width/ (float)height;
    int w = width;
    int h = height;
    if(biAspect>viAspect){
// if sourceImage aspect is greater than output aspect, change h
        h = (int) ((float)w*1.0/biAspect);
    }
    else{
// else change width
        w = (int) ((float)h*biAspect);
    }
}
```

Les commentaires que vous devriez écrire pour ce morceau de code se trouvent dans les fichiers sources fournis.

Stop

Ajouter un bouton stop, qui arrête le défilement des images. Commencez par ajouter le bouton à l'interface. Associez une action à l'activation du bouton (qui affiche simplement un message). Enfin implémenter l'arrêt du défilement.

Transitions

Modifier la méthode update(int) de la classe SlideShow pour permettre une transition entre deux images (ajouter une donnée membre correspondant au temps de transition). Pour mixer deux images, le plus simple est de copier la première image dans mixedImage puis de dessiner par-dessus l'autre image en utilisant un filtre RescaleOp changeant l'alpha de la deuxième image. Vous trouverez un exemple sur le site <http://java.sun.com/docs/books/tutorial/2d/images/drawimage.html>

Libre à vous d'ajouter d'autre transition (fondu au noir, au blanc, transformation diverse etc ...)

Si nécessaire ajoutez des classes pour la gestion des transitions.

Interface

Ajouter des widgets permettant le contrôle de la durée d'affichage d'une image, de la durée de transition, et de toutes options que vous voulez offrir pour sur les transitions que vous avez définies. Voir <http://java.sun.com/docs/books/tutorial/uiswing/components/componentlist.html>

NB : le code fourni n'est pas parfait, peu débiter. Si vous avez un doute sur l'utilité d'une partie du code, commentez le, testez, faites mieux (et posez moi la question pour être sûr, c'est avec joie que je publierai des patches)