Éditeur graphique vectoriel

CREATION : 2011/01/21
MISE A JOUR : 2012/02/10

Les tutoriels officiels Java montrent comment réaliser un éditeur de texte avec interface Swing. Je vous propose de vous faire réaliser les débuts d'un éditeur graphique vectoriel.

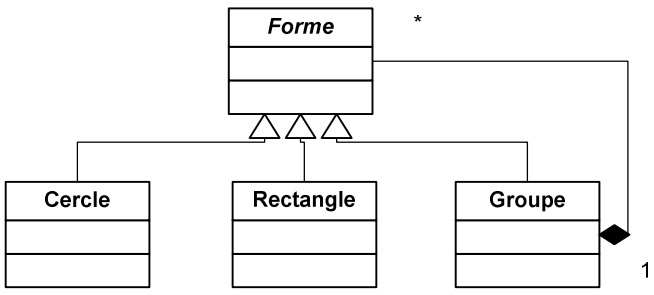
Préparation

- Créer un projet Eclipse.
- Créer une classe Application. Cette classe aura pour attribut une fenêtre qui sera affichée avec le bon titre. On peut utiliser un SwingWorker, invokeAndWait() ou invokeLater() pour créer et afficher l'interface.
- Ajouter à la fenêtre un JCanvas qui pour l'instant ne fait qu'afficher un fond uniforme.
- Ajouter un événement souris pour qu'à chaque clic dans le JCanvas apparaisse un rectangle de taille fixe et de couleur aléatoire.
- S'assurer que chaque nouveau rectangle est stocké dans une liste chaînée (LinkedList) pour être affiché correctement lorsque la fenêtre redevient visible.
- Modifier les événements souris pour que maintenant le rectangle soit dessiné grâce à la souris : Appuyer sur un bouton détermine le premier angle, le deuxième est déterminé lorsque le bouton est relâché. La fenêtre pourra être mise à jour pendant le déplacement de la souris (*drag*).
- Intercaler un JScrollPane pour parcourir la zone de dessin entière si la fenêtre est rétrécie.

Éditeur

On va maintenant ajouter un modèle objet pour être capable de dessiner différents objets. On va ajouter la possibilité de dessiner une ellipse (en donnant le rectangle dans lequel elle s'inscrit)

- Ajouter une barre d'outils (JToolBar) pour permettre le choix du type de figure : cercle ou rectangle. On peut ajouter une aide contextuelle avec un JToolTip. Icones disponibles sur mon site web : http://www.isima.fr/~loic/java/icones
- Créer la hiérarchie d'objets suivante (en oubliant Groupe pour l'instant)



- Faire le lien entre un bouton de la barre d'outils et l'objet à créer
- Ajouter un bouton pour effacer tout l'écran
- Ajouter un bouton pour faire le choix de la couleur de l'objet à dessiner.
- Ajouter un bouton pour permettre la sélection d'un objet pour le déplacer.

On va maintenant s'intéresser à la sauvegarde / lecture de la scène.

- Ajouter une barre de menus et un menu fichier (JMenuBar, JMenu, JMenuItem)
- Ajouter un menu Nouveau qui remet à zéro la liste des objets
- Ajouter un menu sauvegarder qui dans un premier temps sauvegarde la scène sous un nom fixé. Je vous laisse le choix du format : binaire ou xml
- Ajouter un bouton pour recharger la scène à partir du nom fixé.
- Ajouter une boîte de dialogue qui affiche un message d'avertissement avant de quitter si la scène n'a pas été sauvegardée (OPTIONNEL)
- Utiliser des boîtes de dialogues (JFileChooser) pour choisir le nom du fichier à écrire ou à lire (OPTIONNEL)

Aller plus loin

Voilà quelques pistes pour aller plus loin :

- Implémenter une interface MDI (Multiple Document Interface) et non plus SDI
- Permettre la sélection multiple d'objets
- Permettre d'autres actions sur les objets sélectionnés (changement de taille, de couleur, …)
- Ajouter la notion de groupe d'objets (grouper/dégrouper), ce qui a un impact sur la modélisation objet avec l'introduction du pattern composite (un objet qui en contient d'autres)
- Ajouter un copier/coller/couper
- Ajouter un UNDO/REDO
- Gérer l'ordre (Z-index) des objets
- Internationaliser l'application ou i18n (voir la classe Bundle)
- Faire un fichier d'aide (HTML ou autre)

Exemple de lecture d'une icône

```
public ImageIcon createImageIcon(String path, String desc) {
   java.net.URL imgURL = getClass().getResource(path);
   if (imgURL != null) {
      return new ImageIcon(imgURL, desc);
   } else {
      System.err.println("[ERROR]" + "File"+ path + " not found");
   }
   return null;
}
```