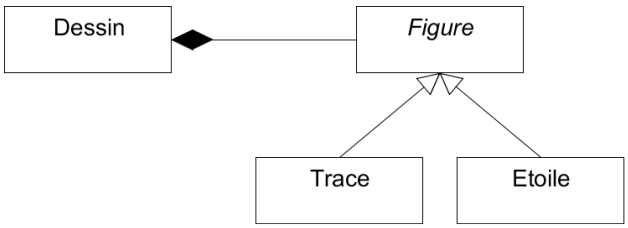
Travaux dirigés – séance n°4 : Architecture MVC

Nous avons maintenant une applette de dessin avec 2 outils de dessin (crayon et étoile) et un modèle dans lequel un dessin est composé de traces et d'étoiles que l'on généralise en une classe abstraite Figure comme l'illustre le diagramme de classes suivant :



Le modèle permet la mémorisation des données, la vue permet l'affichage et les contrôleurs permettent la prise en compte des sollicitations de l'utilisateur, qu'elles consistent en frappes de touche au clavier ou utilisation de la souris. Pour cette raison, on introduit un contrôleur pour prendre en compte la frappe au clavier des touches c (pour crayon) et e (pout étoile) en changeant d'outil par un remplacement d'écouteur souris (MouseListener et MouseMotionListener).

Etudier et comparer les versions « applet de dessin v1 avec 2 outils et modèle » et « applet de dessin v1 avec 2 outils, contrôleur principal et modèle ».

La séparation des rôles est maintenant bien réalisée mais le redessin ne se fait pas. Pourquoi ?

Proposer une solution.

Développement d'une application graphique avec swing

Une application graphique utilise l'architecture MVC (modèle-vue-contrôleur).

Le modèle est le même que pour l'applette.

Les contrôleurs (EcouteurSouris, EcouteurCrayon et EcouteurEtoile) sont pratiquement les mêmes que pour l'applette (il faut les adapter en remplaçant les références à l'applette par des références à une fenêtre).

La vue sera une fenêtre, instance d'une classe **`FenetreDeDessin`** qui hérite de **`JFrame`** et qui comporte :

- une zone de dessin dont la largeur et la hauteur sont ajustées à la taille de la fenêtre. Quel agenceur de composants est approprié ? Où ajouter cette zone de dessin dans la fenêtre ?
- une barre d'état qui tient l'utilisateur informé par un affichage actualisé, toujours situé en bas de la fenêtre. Où ajouter cette barre d'état dans la fenêtre ?

Dans cette classe de fenêtre, on définira les méthodes suivantes :

```
public void dessineTrait(int x1, int y1, int x2, int y2);
public void afficheCoordonnées(int x, int y);
public void afficheOutil(String nomOutil);
public void afficheCouleur(String nomCouleur);
```

La première délègue le traitement à la zone de dessin, les trois dernières méthodes tiennent l'utilisateur informé par des mises à jour de la barre d'état. Elles délèguent donc les traitements à la barre d'état.

Travaux dirigés – séance n°4 : Architecture MVC

Le dessin se fait dans une instance de **`ZoneDeDessin`**, classe qui hérite de **`JPanel`** dans laquelle on définira les méthodes suivantes :

```
public void dessineTrait(int x1, int y1, int x2, int y2);
public void dessineFigure(Figure figure, Graphics g);
public void dessineFigure(Trace trace, Graphics g);
public void dessineFigure(Etoile etoile, Graphics g);
public void paint(Graphics g);
```

Cette zone de dessin utilise les écouteurs souris. Il faut les compléter pour mettre à jour l'affichage de la position de la souris dans la barre d'état.

La barre d'état, qui est toujours affichée en bas de la fenêtre, comporte 3 zones régulièrement espacées sur la largeur de la fenêtre. Quel agenceur de composant est approprié ?

Ces zones d'affichage sont des étiquettes, instances de **`JLabel`**. Elles affichent respectivement les coordonnées de la souris, l'outil de dessin (crayon, étoile), la couleur.

La barre d'état est définie dans une classe qui hérite de **`JPanel`** avec les méthodes suivantes :

```
public void afficheCoordonnées(int x, int y);
public void afficheOutil(String nomOutil);
public void afficheCouleur(String nomCouleur);
```

Pour terminer, la fenêtre comporte une barre de menus faite d'un seul menu déroulant intitulé Dessin, comportant 3 éléments :

- une option Effacer (qui efface le dessin en appelant la méthode vider – à définir, de la classe Dessin) ;
- un séparateur ;
- une option Quitter (qui quitte l'application).