

Rapport Projet Paint

Introduction



Réalisations d'une application de dessin distribué sous une architecture MVC avec un modèle tournant sur un serveur accessible via RMI.

Les Vues Clients

Nous avons réalisé plusieurs vues dans ce Projet, celles-ci sont décrites ci-dessous :

❖ La Vue Edition :

Cette Vue permet de modifier le dessin de l'application et d'avoir accès à toutes les fonctionnalités de celle-ci, la vue éditable est donc composée de plusieurs éléments :

-  **Une feuille de dessin** : Support de la fenêtre permettant d'accueillir les formes que l'on va dessiner, chaque forme est un fils de la feuille de dessins.
-  **Une barre de tache** : Barre à outils de l'application qui permet de choisir la fonctionnalité en cours d'utilisation :
 - *Mode de dessin* : ComboBox permettant de choisir l'action réalisable sur la feuille : dessiner un cercle, rectangle, triangle, ligne, déplacer une forme, changer la couleur de fond d'une forme ou même dessiner à main levée à l'aide du mode libre.
 - *Épaisseur du trait* : ComboBox permettant de modifier l'épaisseur du trait de dessin c'est à dire le contour d'une forme ou l'épaisseur d'une ligne ou du mode libre.
 - *Couleur du trait et du fond* : ColorPicker permettant de choisir la couleur du trait ou du fond, l'option « transparent » est disponible également.
 - *Undo – Redo* : Bouton permettant d'annuler une action ou de répéter une action préalablement annulée.
 - *Grouper – Dégroupier* : Bouton permettant de grouper et dégroupier des formes ensemble, ainsi elles seront liées les unes aux autres et pourront se déplacer de manière unie.
 - *Sauvegarder et charger* : Permet de sauvegarder sous un nom choisi, le dessin réalisé dans la base de donnée, le dessin pourra également être chargé depuis la base de donnée.

❖ La Vue Lecture seule :

Cette Vue donne un aperçu des formes dessinées depuis la vue édition à travers le modèle se trouvant sur le serveur aucune modification n'est possible depuis cette vue elle est composée d'une feuille de dessin étant le fond de la fenêtre comme pour la vue édition mais est dépourvue de barre de taches.

❖ La Vue Statistique :

Cette Vue affiche les proportions de manière dynamique pour chaque type de formes sur le dessin sur un graphique.

❖ La Vue Terminal :

Cette Vue représente sous formes textuelle les différentes formes présentent dans une figure.

Les Fonctionnalités

Voici les fonctionnalités disponibles dans notre application:

Mode de dessin : Formes disponibles.

- *Cercle* : Les coordonnées du clic seront le centre du cercle.
- *Rectangle* : Les coordonnées du clic seront, selon l'orientation, l'un des quatre coins d'un rectangle.
- *Triangle* : Les coordonnées du clic seront le sommet du triangle.
- *Ligne* : Les coordonnées du clic seront celle du point de départ de la ligne.
- *Libre* : Les coordonnées du clic seront celle du point de départ de la ligne du dessin.
- *Groupe* : Lie les formes sélectionné ensemble pour qu'elle ne forme plus qu'une forme.
- *Taille du trait* : modifier la taille du trait de dessin c'est à dire le contour d'une forme ou l'épaisseur d'une ligne ou du mode libre.
- *Couleur du trait et du fond* : Modifie la couleur du trait ou du fond, l'option « transparent » est disponible également.

Undo – Redo : Bouton permettant d'annuler une action ou de répéter une action préalablement annulée.

Sauvegarder et charger : Permet de sauvegarder sous un nom le dessin réalisé dans la base de donnée ou de charger un dessin via une liste depuis la base de donnée.

La Base de données

Description de la base de données :

DTO :

- *FigureDto* : il s'agit du dessin « Figures » qui contient toutes les formes « Shapes ».
- *ShapeDto* : il s'agit d'une forme, celle-ci peut être un ShapeBlock qui contient d'autres formes ou qui contient d'autre Shapeblock.
- *PointDto* : il s'agit de l'ensemble des points qui définissent une forme.



Les tables :

- Figure : Table représentant le dessin : 3 attributs
 - Id => id en Long – clé primaire – Non nul.
 - Figurename => le nom sauvegardé pour le dessin – unique – Non nul.
 - Datemaj => la dernière date de mise à jour du dessin
- Shape : Table représentant l'ensemble des formes sauvées.
 - Id => id en Long – clé primaire – Non nul.
 - Shapetype => le type de la forme qui se trouve dans ShapeEnum – Non nul.
 - Strokecolor => la couleur du trait de la forme.
 - Fillcolor => la couleur du contenu de la forme.
 - Strokewidth=> l'épaisseur du trait de la forme.
 - Figure=> la figure dans laquelle se trouve cette forme – clef étrangère qui pointe vers l'id de la figure dans laquelle elle se trouve
 - Shape => le conteneur de la forme ou la forme père (si la forme se trouve dans un group sinon cet attribut est à nul) – clef étrangère qui pointe vers l'id du Shape père s'il existe.
 - Radius = > le radion s'il s'agit d'un cercle sinon il sera nul.
- Point : Table représentant l'ensemble des points sauvés.
 - Id => id en Long – clé primaire – Non nul.
 - X => la position du point sur l'abscisse – Non nul.
 - Y => la position du point sur l'ordonnée – Non nul.
 - Shape => l'id de la forme auquel le point appartient – clef étrangère qui pointe vers l'id du Shape auquel le point appartient – Non nul.
 - Listorder => l'ordre du point dans la forme auquel le point appartient – Non nul.

L'architecture RMI

Le modèle est situé dans le serveur via la classe Paint.java qui a l'objet FigureModel qui est notre état, la classe Paint implémente l'interface PaintInterface ce qui nous permet de récupérer via la méthode « FigureModel getState() » l'état du modèle du serveur.

FigureModel et tous les objets en découlant sont reconstruit du côté du client, à part l'instance de la classe Paint car elle étant « UnicastRemoteObject » et non « Serializable » elle est donc donnée par référence.



Modèle :

- FigureModel représentant notre dessin, il contient l'ensemble des formes dessinées
 - Il sera envoyé à chaque mise à jour du dessin
 - Le FigureModel constitue l'état du modèle

✚ Parcours d'une modification :

- Action sur le client : A chaque action réalisé sur le client la méthode `setState(« ACTION DESIREE »)` sera appelée elle prendra en paramètre l'action que l'on exécute, afin d'informer le serveur du changement réalisé sur le dessin (`FigureModel`).
- Action sur le serveur : Le serveur récupère l'action et l'exécute ensuite notifie les vues distantes grâce à « `notifyChangementDistant()` » qui lance le notifie changement de la classe `PaintClient` qui implémente `EcouteurDistant`.
- Notification sur le client : `PaintClient` notifie à son tour toutes les vues qui lui sont liés.
- Mise à jour des vues : Les vues demandent au serveur l'état du modèle via « `getModel().getState()` », les vues vont ensuite être redessiné en fonction de cette état.