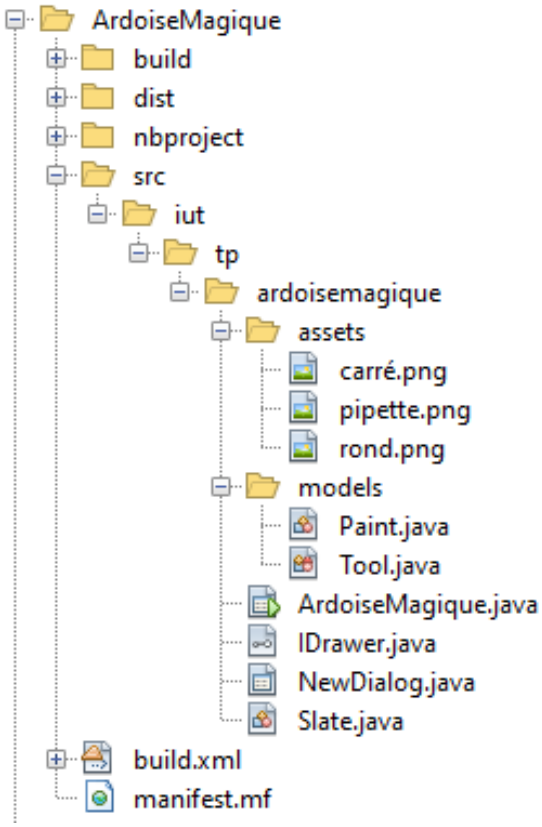
TP3 IHM

L'idée de ce TP est de créer une ardoise magique.

1) Créer un nouveau projet « ArdoiseMagique »
2) Créer l'arborescence du projet



En voici un descriptif sommaire :

iut	package	A créer
iut.**tp**	package	A créer
iut.tp.**ardoisemagique**	package	A créer
iut.tp.ardoisemagique.**assets**	package	A créer
iut.tp.ardoisemagique.assets.**carré.png**	image	Est fourni
iut.tp.ardoisemagique.assets.**pipette.png**	image	Est fourni
iut.tp.ardoisemagique.assets.**rond.png**	image	Est fourni
iut.tp.ardoisemagique.**models**	package	A créer
iut.tp.ardoisemagique.models.**Paint**	class	A créer
iut.tp.ardoisemagique.models.**Tool**	enum	A créer
iut.tp.ardoisemagique.**ArdoiseMagique.java**	class (JFrame form & main)	A créer
iut.tp.ardoisemagique.**IDrawer.java**	interface	A créer
iut.tp.ardoisemagique.**NewDialog.java**	class (JDialog form)	A créer
iut.tp.ardoisemagique.**Slate.java**	class	A créer

L'énumération « Tool » : *énumère les différents outils utilisables (CHOOSER, ROUND, SQUARE) dans l'interface. L'utilisateur peut décider d'utiliser la pipette pour capturer une couleur de l'ardoise, sélectionner un pinceau avec un embout rond ou carré.*

La classe « Paint » : *Correspond au modèle d'un point dessiné sur l'ardoise. Un objet Paint contient la couleur du point dessiné, la position x et y du point, la taille du point, si le point est lisse (utilisation d'un antialiasing), et sa forme (c'est un Tool). Bien sûr, on en conviendra ça n'a pas de sens de créer un objet Paint avec un Tool.CHOOSER.*

La classe « NewDialog » : *Lorsque l'on fait Fichier>Nouveau, cette JDialog modale s'affiche permettant de saisir la taille de l'ardoise à créer. Lorsque celle-ci se referme elle doit renvoyer un objet Dimension représentant la largeur et hauteur de l'ardoise à créer.*

La classe « Slate » : *C'est l'ardoise. En vérité c'est un JPanel sur lequel on redéfini la méthode paintComponent(). L'ardoise a une dimension récupérée par un objet « NewDialog ».*

La classe « ArdoiseMagique » : *C'est la classe principale du projet. Elle hérite de JFrame et possède la méthode main. Graphiquement elle possède une JMenuBar. Son contentPane est un BorderLayout. Sur la droite, il y a un JPanel contenant la liste des contrôles (controlPanel), au centre le panneau principal (mainPanel). Le controlPanel contient un JPanel carré en haut représentant la couleur du pinceau (rouge par défaut). Ce panel a une bordure noire de 1px. En dessous, 3 JToggleButtons qui représentent chacun un « Tool ». En dessous un JPanel de propriétés (propertiesPanel). Sa bordure contient son titre. Il possède un JSpinner pour définir la taille du pinceau [1 ; 50] pixels. Lorsque l'on clique sur l'outil pipette ce panel se grise pour empêcher toutes modifications. A contrario, les autres outils activent le panel et son contenu. Le mainPanel est un BorderLayout. Au sud il y a un JPanel*

(statusBar). Au centre il y a un JScrollPane qui permettra de se déplacer sur l'ardoise si celle-ci dépasse les dimensions de la fenêtre. Dans la JScrollPane, il y a un JPanel (workspace). Ce JPanel est un FlowLayout ancré à gauche. C'est dans ce JPanel que l'on ajoutera un objet Slate.

L'interface « IDrawer » : *Permet de faire le lien entre l'objet Slate et ArdoiseMagique. En effet le dessin se réalisera sur l'objet Slate (donc dans la classe Slate). Mais un Slate a besoin de certaines informations pour dessiner (comme la couleur choisie, la taille et l'outil sélectionné). Ces informations se trouvent dans la classe ArdoiseMagique.*

3) Créer les méthodes de l'interface :

public java.awt.Color getSlateColor(); → permettra à un objet Slate de récupérer la couleur choisie par l'utilisateur dans ArdoiseMagique.

public Tool getSlateTool(); → permettra à un objet Slate de récupérer l'outil choisi par l'utilisateur dans ArdoiseMagique.

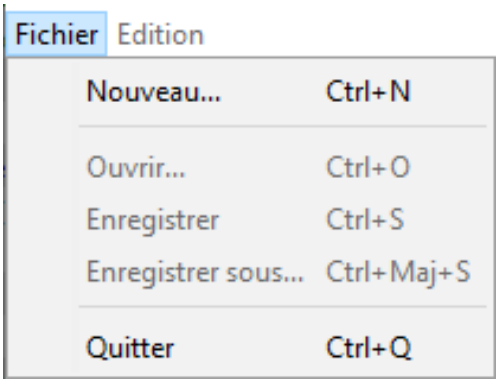
public int getSlateToolSize(); → permettra à un objet Slate de récupérer la taille de l'embout du pinceau choisi par l'utilisateur dans ArdoiseMagique.

public boolean isSlateSmooth(); → permettra à un objet Slate de déterminer si les Paint qui seront créés seront lisses ou pas.

public void newColorChoosen(java.awt.Color newColor); → permettra à un objet Slate de communiquer la couleur capturée par la pipette à l'ArdoiseMagique.

public void newMousePosition(java.awt.Point point); → permettra à un objet Slate de communiquer la position x, y de la souris à l'ArdoiseMagique.

4) Créer l'interface graphique (le plus fidèlement possible) de l'ArdoiseMagique en fonction des explications données précédemment. Ne rien mettre dans la statusBar pour le moment. La menuBar contient un menu Fichier et Edition (grisé pour le moment). Le menu fichier contient les items suivants :



Pour les outils, comme il s'agit de JToggleButton, vous pouvez utiliser un ButtonGroup pour les lier ensemble (comme on le ferait avec des JRadioButton). Par défaut l'outil rond est sélectionné).

5) Créer le contenu de la classe Paint. Constructeur contenant tous les paramètres évoqués précédemment et tous les getters. Aucun setter pour le moment.

6) Créer le contenu de l'énumération Tool. Il n'y a que trois outils : CHOOSER, ROUND, SQUARE.

7) Créer l'interface graphique (le plus fidèlement possible) de NewDialog.

8) Utilisez le cours pour remplir la classe NewDialog. Elle doit contenir une méthode qui ouvre la fenêtre modale (showDialog), reste bloquée (parce que la fenêtre est modale et est ouverte) et attend le choix de l'utilisateur avant de renvoyer un objet Dimension contenant la largeur et la hauteur du futur Slate. Si l'utilisateur choisit d'annuler, alors la méthode renvoie null.

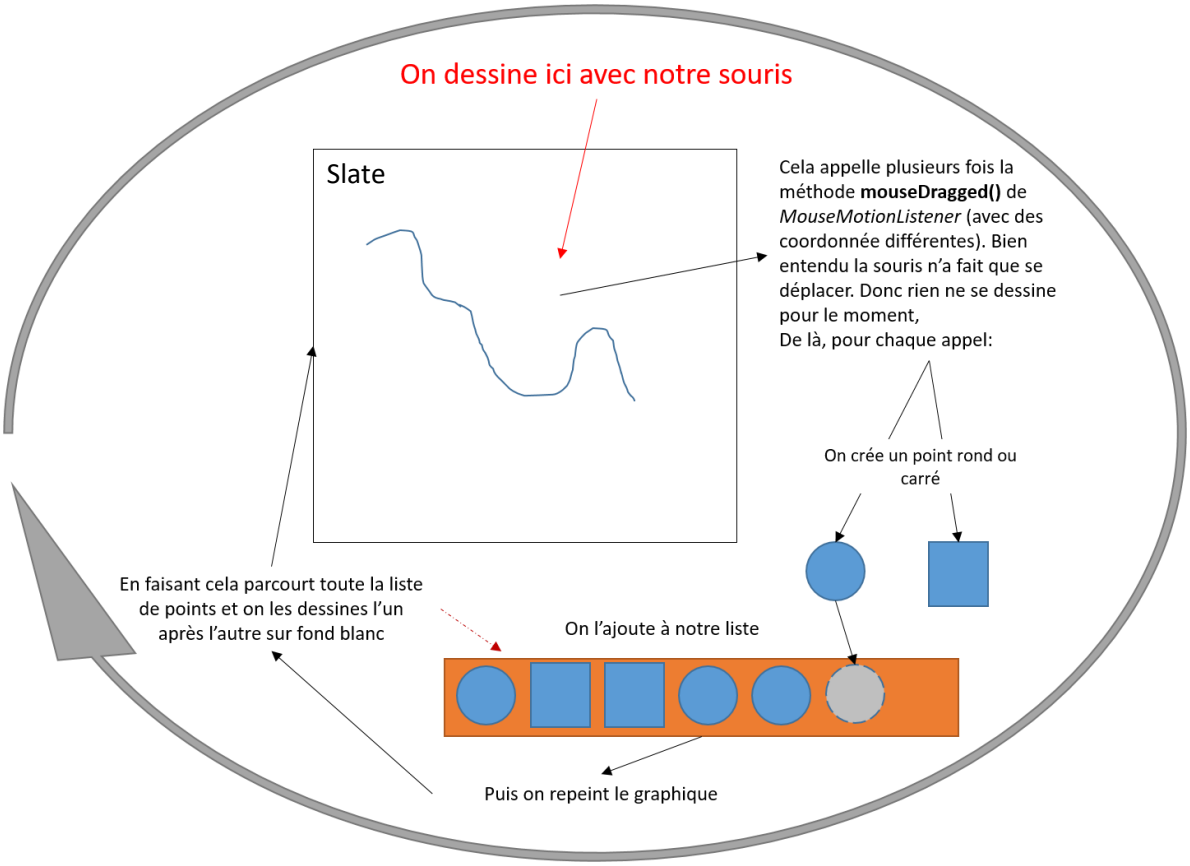
9) Créer l'évènement de l'item « Quitter » de l'ArdoiseMagique. Pour fermer l'application : System.exit(0);

10) La classe ArdoiseMagique doit implémenter IDrawer.

11) La classe Slate a besoin dans son constructeur un objet Dimension et un objet IDrawer. Dans ce même constructeur on initialise un attribut de type List<Paint>. Cette liste contiendra tous les points du dessin (un dessin c'est un ensemble de points) créé par l'utilisateur. Toujours dans le constructeur on définit la taille de l'objet Slate (cf : setSize(), setPreferedSize()…). Attention, il faut que l'objet Slate soit entièrement bloqué dans ses dimensions. Car cela risquerait de déformer le Slate, si on agrandi la fenêtre principale de l'application.

Voici l'idée du fonctionnement du composant Slate :

Puisqu'il s'agit d'un JPanel et que l'on redéfinit sa méthode paintComponent(), celle-ci devra s'occuper de dessiner à chaque fois un background blanc (un rectangle blanc de la taille du Slate), puis au-dessus tous les Paint de sa liste en attribut. Comme un dessin est composé d'une multitude de points, il s'agit de les stocker. C'est le rôle de la liste créée précédemment. Mais comment dessiner à proprement parler ? Il faut placer un listener MouseMotionListener sur le Slate courant. Ainsi lorsque la souris se déplacera dessus (tout en cliquant), on obtiendra une suite d'évènements qui nous suffit de convertir en objet Paint (que l'on stockent ensuite dans la liste). Mais pour autant rien n'est affiché pour le moment. Il faut donc à chaque nouveau point, forcer l'affichage du dessin avec la méthode repaint(). Cela aura pour effet d'appeler automatiquement la méthode paintComponent(). Et comme nous l'avons redéfinie, elle dessine un rectangle blanc sur toute la surface du Slate et redessine chaque point contenu dans la liste. Voici un schéma récapitulatif :



Pour créer un Paint, on a besoin de sa coordonnée (on l'a avec l'évènement mouseDragged()), la forme du pinceau. On l'obtient avec la méthode getSlateTool() de l'objet IDrawer passé en paramètre du constructeur. De même pour déterminer si le point est lisse ou pas et sa taille.

12) Mettre en œuvre l'explication du point 11. Essayer de le réaliser tout seul avant de demander l'aide du prof.

Pour rendre un point lisse il faut (dans la méthode paintComponent()) caster l'objet Graphics en Graphics2D et lui activer ou pas le render avec les méthodes suivantes :

Pour activer

setRenderingHint(java.awt.RenderingHints.KEY_ANTIALIASING, java.awt.RenderingHints.VALUE_ANTIALIAS_ON);

Pour désactiver

setRenderingHint(java.awt.RenderingHints.KEY_ANTIALIASING, java.awt.RenderingHints.VALUE_ANTIALIAS_OFF);

13) Créer l'évènement de l'item « Nouveau… » dans la classe ArdoiseMagique. Pour cela on crée un objet NewDialog et on appelle sa méthode showDialog. En retour dès que l'utilisateur a choisi la taille, celle-ci renvoie un objet Dimension qui nous sert (s'il n'est pas null) à créer un objet Slate. L'objet Slate a besoin de 2 paramètres, une dimension (que l'on vient d'obtenir) et un objet IDrawer (ça tombe bien ArdoiseMagique implémente IDrawer).

14) Maintenant que nous avons notre objet Slate il s'agit de l'ajouter au panel workspace. Comme celui-ci ne l'affiche pas dynamiquement il faut lui faire comprendre qu'il y a bien un nouvel objet en interne et qu'on souhaite le voir. On utilise la méthode revalidate() pour forcer workspace à prendre en compte la modification.