Projet POO : PetRescue Saga

Développé par HU Wei et KE Zong-You (M1 Ling)
janvier 2021

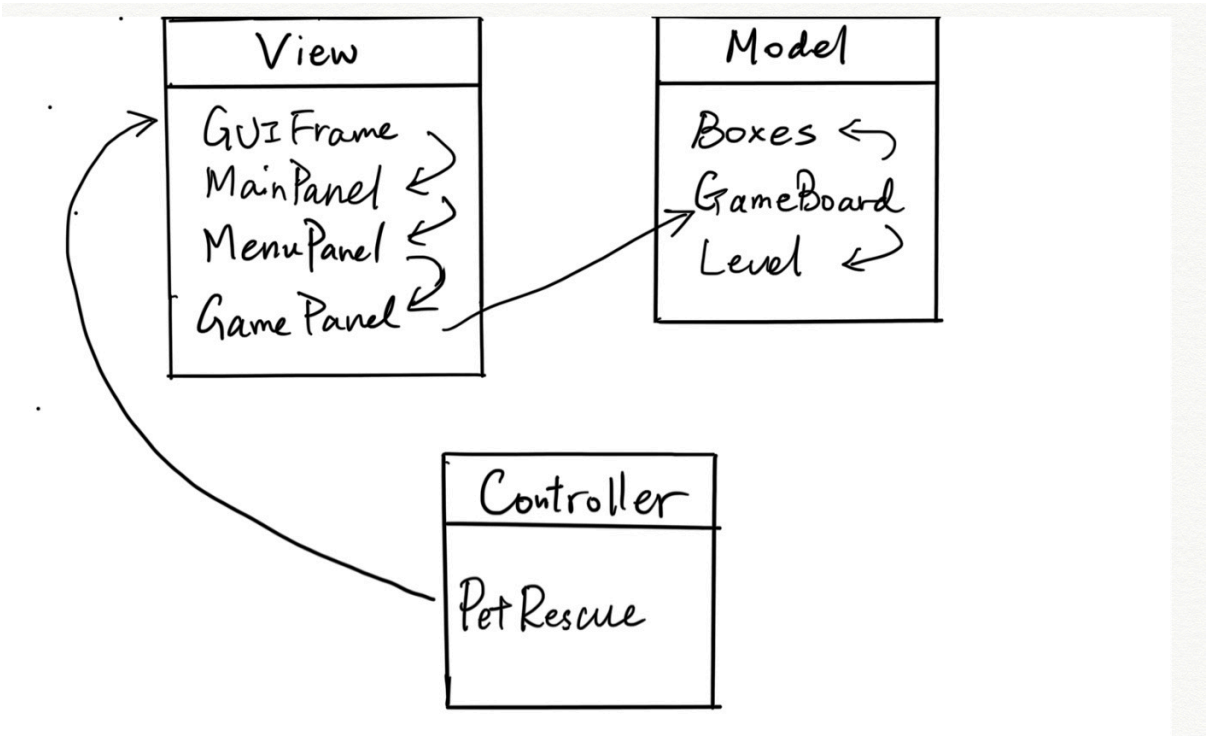
I. Introduction

Ce rapport explique la conception du jeu avec des diagrammes et les codes.

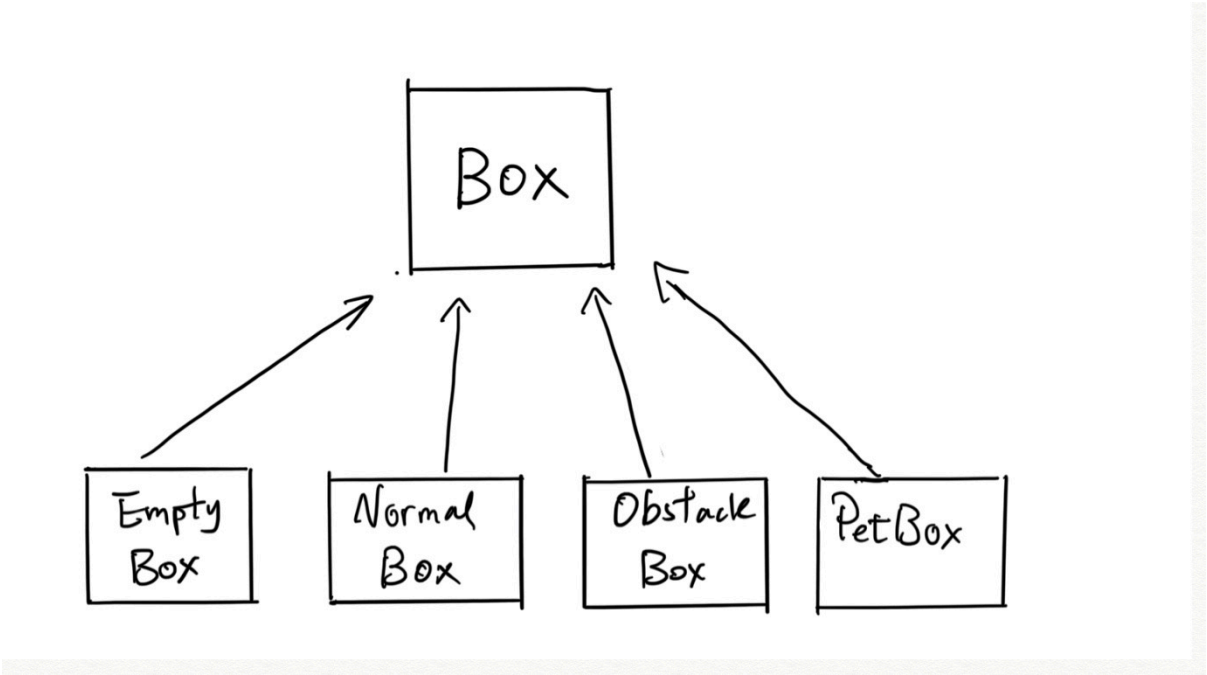
II. Diagrammes de classes

On a divisé les classes en trois catégories : View (Vue), Model (Modèle) et Controller (Controleur), chacune contenant une ou plusieurs classes qui s'interagissent.

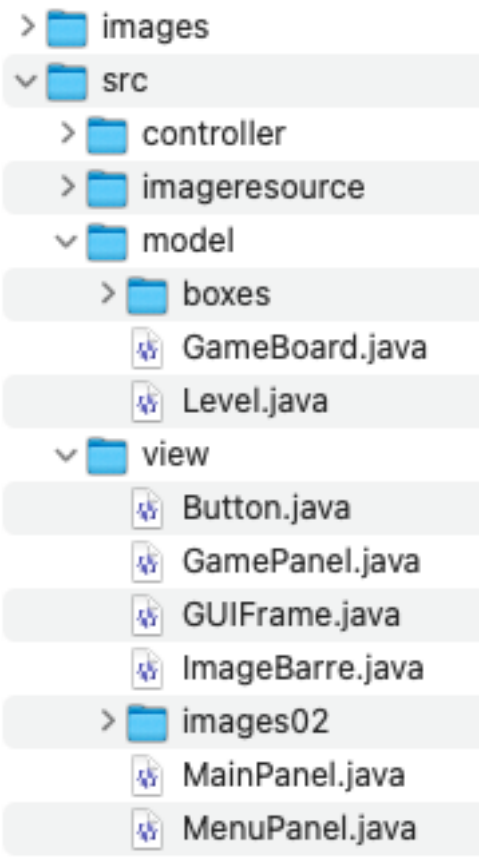
Voici un diagramme qui indique les classes importantes et comment chacune appelle l'autre (la notation $A \rightarrow B$ signifie que la classe A appelle ou utilise la classe B) :



Voici un autre diagramme qui indique les héritages entre la classe BOX et d'autres classes de types de BOX :



La structure des classes sous forme d'un arbre est comme suit :



III. Fonction expliquée

On choisit la fonction *Image* pour expliquer en détail :

Il existe plusieurs façon à introduire des images dans nos fichiers, Dans notre travail, nous avons choisi la méthode BufferedImage et la façon URL à introduire des images dont nous avons besoins.

a) L'image est une sous-classe directe du paquet java.awt, classe abstraite.

b) BufferImage est une sous-classe directe d'image avec une fonctionnalité de mise en mémoire tampon supplémentaire. L'image générée par BufferedImage possède une mémoire tampon d'image en mémoire, en utilisant cette mémoire tampon nous pouvons facilement manipuler cette image, généralement utilisée pour faire des opérations de modification d'image telles que le changement de taille, le grisonnement de l'image, le réglage de l'image transparente ou opaque, etc.

Pour charger une image en mémoire :

```
BufferedImage image = ImageIO.read(new FileInputStream(imgPath));
```

//Utilisez la fonction getGraphics() pour obtenir le contexte graphique pour les opérations ultérieures.

```
Image image = imageToolkit.getDefaultToolkit().getImage(imgPath) ;
```

Cette méthode ne charge pas l'image en mémoire, elle est simplement référencée.

c) ImageIcon se trouve directement sous le paquet javax.swing

La classe publique ImageIcon étend l'Icône d'implémentation des objets, sérialisable, accessible.

La relation entre imageicon et image est que l'imageicon dessine une icône en utilisant une image. Cependant, l'image est généralement trop grande pour être utilisée comme icône (seule une partie de l'image est affichée lorsqu'une grande image est utilisée comme icône) et doit être traitée. Les lignes de codes sont comme suit :

```
ImageIcon imageIcon = new ImageIcon(new File(path)) ;
Image image = imageIcon.getImage() ;
image = image.getScaledInstance(30,20,Image.SCALE_FAST) ;
ImageIcon icon = new ImageIcon(image);
```

//Utiliser le constructeur de l'imageicon public ImageIcon(Image i)

IV. Des bugs

Pourquoi la méthode statique ne peut pas consulter la méthode non-statique ?
Les méthodes statiques appartiennent à des classes, c'est-à-dire que les méthodes statiques sont chargées avec la classe et que le programme alloue de la mémoire pour les méthodes statiques lorsque la classe est chargée

Les méthodes non statiques appartiennent aux objets, qui sont créés après le chargement de la classe.

Les méthodes statiques existent avant les objets. Lorsque vous créez un objet, le programme lui alloue de la mémoire dans le tas, généralement au moyen d'un pointeur vers l'objet. Les méthodes statiques ne dépendent pas de l'objet appelé, il est appelé via le "nom de classe". Nom de la méthode statique" de manière à ce qu'il soit invoqué. Pour les méthodes non statiques, le programme leur alloue de la mémoire uniquement lorsque l'objet est créé, puis accède à la méthode non statique par le biais de l'objet de classe. Ainsi, les méthodes non statiques n'existent pas lorsque l'objet n'existe pas, et les méthodes statiques ne peuvent naturellement pas appeler une méthode qui n'existe pas.

a) Il s'agit ici d'un appel de classe à une méthode, et non d'un appel d'objet à une méthode.

b) La méthode est une méthode statique, il suffit d'utiliser "classe". méthode", car l'utilisation de méthodes statiques ne dépend pas du fait que l'objet soit créé ou non. null peut être forcé à convertir un type en n'importe quel type (pas n'importe quel type d'objet, lorsque le statique sera supprimé lors de l'exception du pointeur nul, si le type d'objet peut être appelé), vous pouvez donc l'utiliser pour exécuter la méthode statique.

c) Les méthodes non statiques utilisent l'"objet". méthode", doit dépendre de l'objet est créé à utiliser, si la méthode testMethod() avant la suppression de statique, va signaler une exception de pointeur nul . Cela permet également de vérifier le point 2)

V. Captures d'écran

Figure 1 : Menu principal

Figure 2: Niveau 1

Figure 3 : Fenêtre affichée après la complétion d'un niveau

Figure 4 : Niveau 2

Figure 5 : Fenêtre demandant le joueur de quitter le jeu ou non


Pet Rescue Saga
score : 7500
Wei
Zong-you
Pet
Rescue
saga
Terminer
Refaire le jeu ?
No
Yes