Examen Final

Développement de programmes dans un environnement graphique – SIM – A21

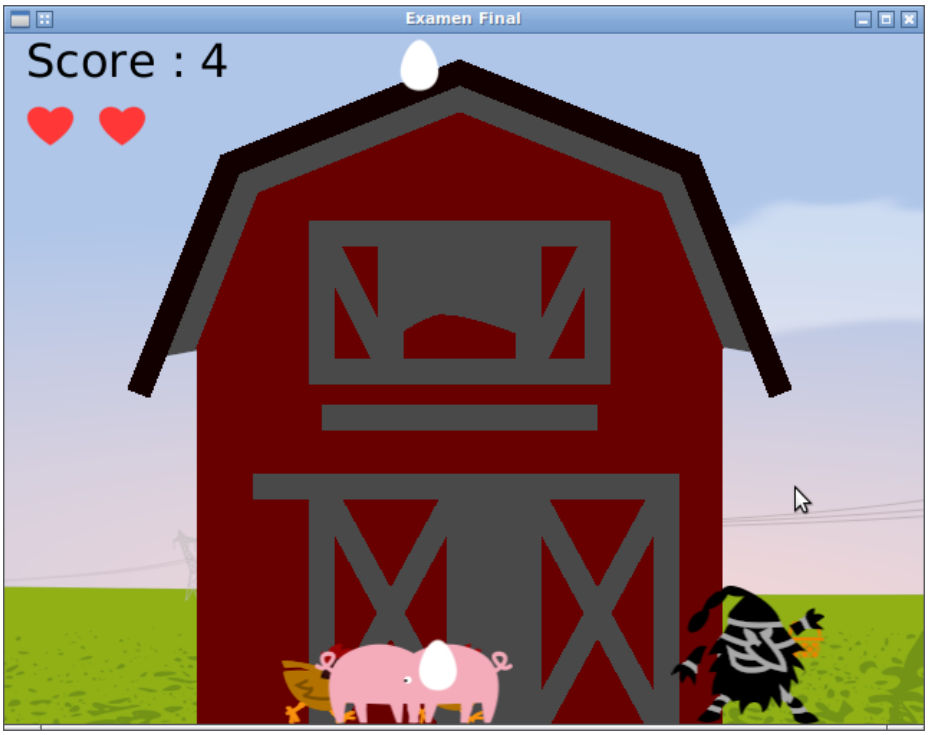
Nicolas Hurtubise

Barème

Pondération	Élément d'évaluation
50%	Instructions 1 à 4
30%	Instructions 5 à 7
20%	Qualité du code

Contexte

Votre tâche sera de coder un petit jeu dans un `Canvas` avec JavaFX.



Projet de base

Je vous fournis déjà une petite base de code :

- `Main` : le point d'entrée de l'application
- `Input` : une petite classe utilitaire pour savoir quelle touche est appuyée à chaque instant (même classe qu'on a vu en cours)
- `GameObject` : une classe abstraite qui définit la base de ce qu'un objet du jeu peut faire
- `Partie` : qui ne contient presque rien pour le moment, mais qui devrait contenir la logique de votre code

Je vous fournis également les images de base à utiliser, qui sont déjà dans le dossier `resources/` du projet.

Instructions

➢ Vous avez **3 heures, incluant le temps pour remettre le projet sur Léa**

Si vous n'arrivez pas à tout faire, vous serez évalués sur ce que vous avez complété.

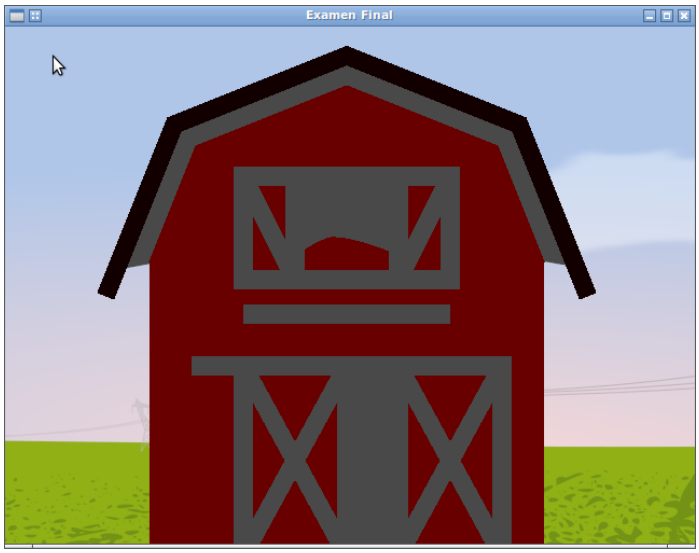
Mieux vaut un code qui marche mais qui ne fait pas tout plutôt qu'un code qui ne marche pas mais qui essaie de (mal) tout faire.

Allez-y une seule étape à la fois, dans l'ordre où c'est écrit dans les instructions.

Si vous bloquez sur une des choses à faire, c'est ok de passer à la prochaine, mais assurez-vous simplement de ne pas tout essayer de faire en même temps.

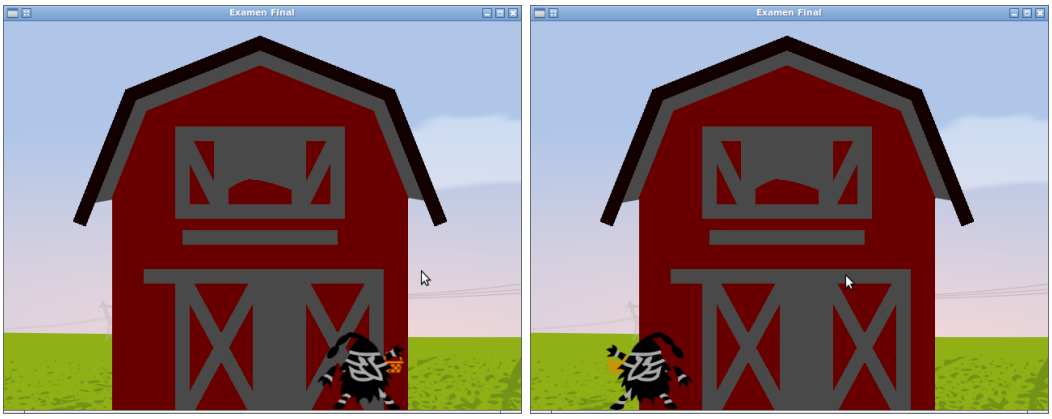
1. Arrière-plan

La ferme est affichée avec l'image `arriere-plan.png`.



2. Gnome

Le personnage contrôlé au clavier est un petit Gnome, qui peut marcher de gauche à droite sur le sol :



Le Gnome a une taille de Largeur=107px, Hauteur=96px et peut marcher de gauche à droite à une vitesse de 250px/s.

Le Gnome ne peut pas sauter. Le Gnome **ne peut pas aller plus loin que les côtés de l'écran**.

3. Animaux

Il y a différents animaux qui se promènent sur la ferme :

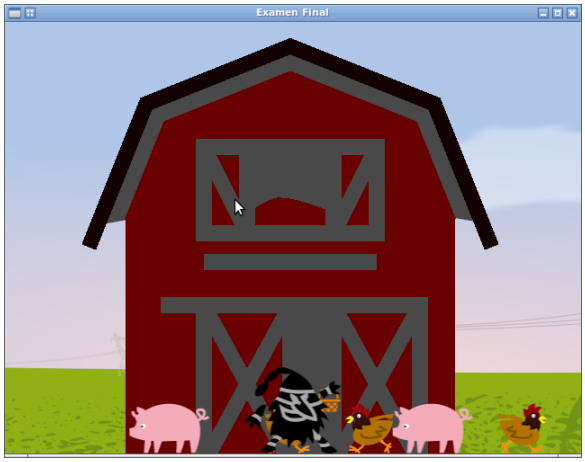
- Les poules, qui marchent horizontalement à 50px/s
- Les cochons, qui marchent horizontalement à 85px/s

Autant les poules que les cochons avancent en ligne droite jusqu'à atteindre le côté de l'écran. Quand ils frappent le côté de l'écran, ils changent de sens.

Le jeu doit commencer avec **3 poules** et **2 cochons**, posés sur le sol, à des positions en x choisies au hasard.

Les poules ont une taille de Largeur=54px, Hauteur=56px.

Les cochons ont une taille de Largeur=90px, Hauteur=56px.



4. Oeufs

Pour une raison quelconque, des oeufs tombent du ciel à toutes les 3 secondes.

Les oeufs sont créés au-dessus du haut de l'écran (donc on ne les voit pas encore) à une position horizontale choisie au hasard, et avec une vitesse initiale de 0px/s.

Les oeufs subissent une gravité de $50\text{px}/s^2$ vers le bas.

Quand le *bas* d'un oeuf touche le sol, il disparaît.

5. Score

Le but du jeu est de ramasser les oeufs avant qu'ils ne s'écrasent parterre.

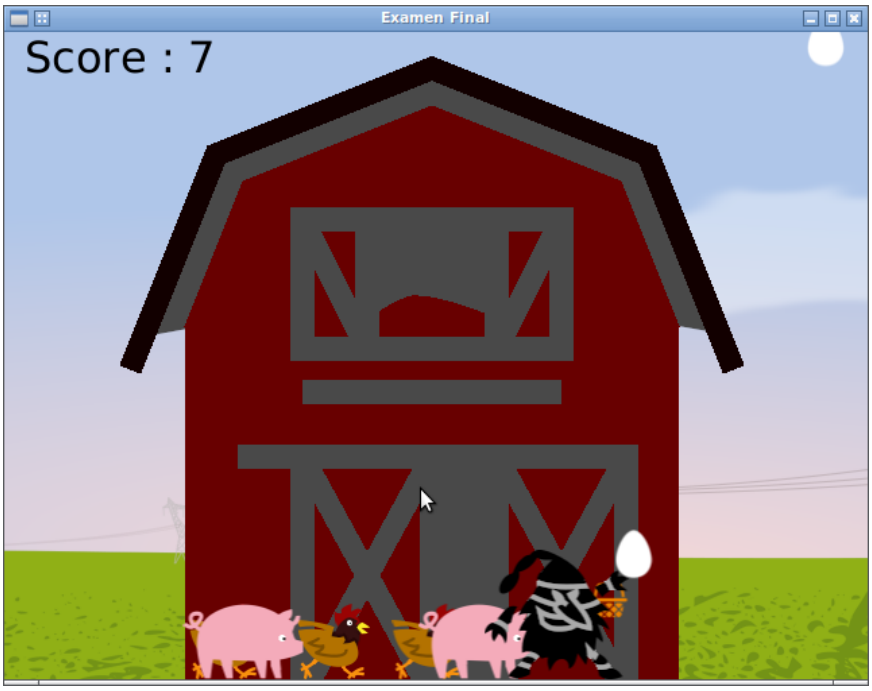
Quand le gnome est à une distance de moins de 50px d'un oeuf, il le ramasse (donc l'oeuf disparaît du jeu).

On calcule cette distance **depuis le centre** du Gnome **jusqu'au centre** de l'oeuf. Utilisez la distance euclidienne :

$$distance(gnome, oeuf) = \sqrt{(gnome_x - oeuf_x)^2 + (gnome_y - oeuf_y)^2}$$

Chaque oeuf collecté fait augmenter le score de +1.

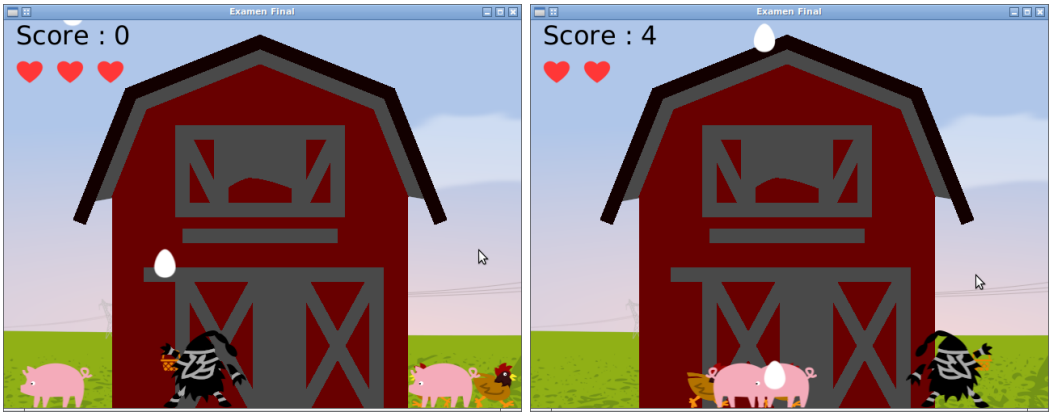
Le score est affiché **en noir** en haut à gauche de l'écran.



6. Vies

On commence avec 3 coeurs, qui sont affichés en haut à gauche de l'écran.

À chaque fois qu'un oeuf touche le sol avant que le Gnome n'ait pu l'attraper, ça fait descendre de 1 le nombre de coeurs.



7. Fin de partie

Si on tombe à 0 coeurs, on perd la partie.

À ce moment, on affiche "Perdu!" **en rouge** sur l'écran pendant 3 secondes, puis on recommence le jeu avec 3 coeurs et 0 points de score. Les oeufs qui étaient déjà sur le jeu sont supprimés, on recommence une nouvelle partie de zéro.

Précisions

Utilisez la structure du projet fourni pour ne pas perdre de temps.

Vous serez évalués sur le respect du découpage MVC, mais si vous suivez la structure proposée, ça ne devrait pas être très difficile à faire.

Si quelque chose n'est pas spécifié dans l'énoncé, c'est que ce n'est pas grave. Par exemple : je ne serai pas sévère sur la position au pixel près du mot "Perdu!" quand on perd la partie.

Annexe : classes et méthodes utiles

Mis à part les classes fournies, vous ne devriez pas avoir besoin d'utiliser beaucoup de choses différentes.

La documentation à même IntelliJ vous sera utile.

Vous aurez principalement besoin des méthodes du `GraphicsContext` :

```
// Charger une image depuis les ressources du projet
Image image = new Image("truc.png");

// Dessiner l'image avec son coin haut-gauche en (x, y)
context.drawImage(image, x, y);
```

Si vous voulez dessiner du texte à l'écran directement dans le `canvas`, vous pouvez utiliser :

```
// Changer la couleur du texte
context.setFill(Color.BLUE);

// Changer la taille de la police d'écriture utilisée :
context.setFont(Font.font(55));

// Changer l'alignement du texte
context.setTextAlign(TextAlignment.CENTER); // LEFT et RIGHT existent aussi

// Dessiner du texte en (x, y)
context.fillText("Bonjour tout le monde!", x, y);
```