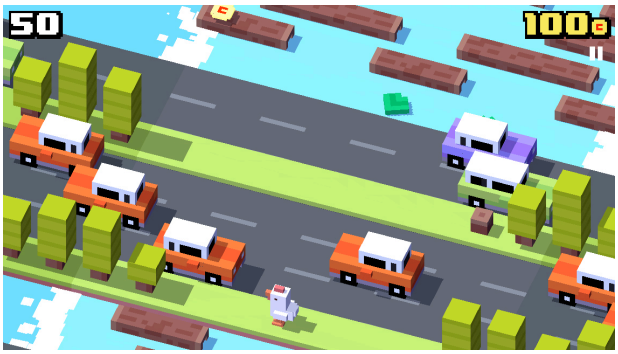
Héritage en folie



L'idée est ici de développer une variante Java du jeu *CrossyRoad*. Pour cela, nous allons utiliser un environnement de développement Java tourné vers le jeu : *Greenfoot*.

Pour ceux qui ne connaissent pas ce célèbre jeu pour mobiles vendu à plusieurs dizaines de millions d'exemplaires :

- https://www.youtube.com/watch?v=Z675yex7Yk8
- https://youtu.be/kQmCK2AdiYE

1. Phase de conception

Listez les différents types d'objets qui interviennent dans ce jeu, essayez de les regrouper par catégories, de séparer ensuite ce qui est commun aux objets d'une même catégorie de ce qui leur est spécifique.

Faite un diagramme de classes UML et faites-le valider par votre enseignant. Attention à bien différencier les relations **d'héritage** des relations de (forte) **dépendance**. Représentez aussi au sein des classes les **attributs** et les **méthodes** envisagées. Indiquez quelles classes sont **abstraites**, etc.

2. Découverte de Greenfoot

Greenfoot est un EDI similaire à *BlueJ* (même auteur), mais incorporant des bibliothèques facilitant le développement de jeux : gestion du déplacement d'objets, de collisions entre objets, de sons et de graphismes, gestion des touches clavier, etc.

Partons d'un exemple pour être plus concret :

1. Chargez le projet Demo pour GreenFoot disponible sur Moodle
2. Une fois chargé dans Greenfoot, lancez une exécution du jeu (touches clavier pour agir) et comprenez ce que permet ou pas ce jeu.
3. Regardez maintenant la hiérarchie simple d'objets prévus dans ce jeu (colonne de droite de la fenêtre), et explorez le contenu des classes pour comprendre comme elles fonctionnent et interagissent.
4. L'API de greenfoot permet bien plus de choses que ce qui est montré dans cet exemple : http://www.greenfoot.org/files/javadoc/ Consultez l'introduction de la doc des classes *Actor*, *World* et *Greenfoot* et scannez rapidement la liste des méthodes proposées dans chacune.

3. Une première version simple

Dans cette première version, on considère que le décor ne défile pas pour l'instant, et que le but du jeu est que le joueur arrive à faire passer son personnage du bas de l'écran jusqu'en haut de l'écran.

1. Ouvrez un nouveau projet Greenfoot et mettez en place une classe `Monde` dérivant de `World` ainsi que des premières classes dérivant d'`Actor` et représentant vos grandes catégories d'acteurs.
2. Pour chaque catégorie d'acteurs, mettez en place au moins une classe la spécialisant.
3. Associez une image à chaque classe d'acteur terminale (non dérivée) (clic droit sur la classe dans le schéma UML simplifié sur la droite).
4. Faites en sorte qu'au démarrage du projet, votre méthode `Monde` charge un acteur correspondant au personnage (un poulet initialement par exemple).
5. Gérez maintenant le clavier pour que votre personnage puisse de déplacer vers le haut, la gauche ou la droite (mais pas revenir en arrière).
6. Ajoutez maintenant un acteur de type véhicule et débrouillez-vous pour qu'il avance (assez doucement) dans une direction (toujours la même) à chaque tour de jeu (déclenché depuis la méthode `act()` de la classe `Monde`). A ce propos, comment maintenez-vous la liste des acteurs dans cette classe ?
7. Faites en sorte que le jeu s'arrête (méthode `stop()` héritée ?) quand votre personnage entre en collision avec le véhicule ou quand il atteint le haut de l'écran

3. Une version jouable

On veut maintenant passer de l'étape prototype à un jeu réel, simple mais dans lequel on peut commencer à s'amuser. Pour cela :

1. Choisissez une thématique pour votre jeu, propre à votre projet (par exemple CrossyRoad a sorti récemment une version Christmas, une version Brésil, une version Australie)...soyez créatifs
2. Ajoutez une image de fond à votre classe Monde. Cette image sera fixe dans un premier temps, composée de plusieurs bandes : certaines sont des routes, l'une d'entre elle est une voie ferrée, et d'autres sont de la prairie. Pour chacune, il faut pouvoir indiquer le sens de circulation (gauche -> droite ou inverse, mais toujours le même pour les véhicules circulant dessus).
3. Mettez en place des images pour vos acteurs qui ont la hauteur d'une bande du décor.
4. Faites en sorte qu'il y ait plusieurs véhicules qui se déplacent sur votre carte (du même type pour l'instant).
5. Mettez en place au moins deux types de véhicules différents (images spécifiques), par exemple des camions et des voitures sur les routes (chaque type de véhicule a une vitesse propre). Les trains se déplacent très vite et uniquement sur les voix ferrées.
6. Ajoutez un bruit (de moins de 1s) à votre personnage à chaque fois qu'il se déplace (un fichier `wav` par exemple que vous créez ou trouvez sur internet) et/ou à chaque fois qu'il passe à proximité d'un véhicule.
7. Le nombre de véhicules sur les routes doit pouvoir être réglé par une variable que vous maintenez. Faites aussi en sorte qu'un véhicule disparaisse quand il atteint le bord du tableau de jeu (méthode `removeObjects(...)` appliquée sur un objet de la classe Monde) et que de nouveaux véhicules apparaissent régulièrement.
8. Mettez en place une animation spéciale (un acteur) quand le personnage du joueur entre en collision avec un véhicule + un son (fichier wav) de circonstance (genre «splotch» ou marche funèbre ;)
9. Faite que quand le personnage atteint le haut de l'écran, il ré-apparaisse en bas de l'écran et que la difficulté augmente (par exemple le nombre moyen de véhicules sur la carte augmente, ou ils vont plus vite).
10. Mettez en place un système de score, qui fait en sorte que plus le personnage survit longtemps, plus il marque de points. Mettez en place un tableau d'affichage du score réalisé quand la partie se finit.

4. Vers une version freeware/shareware de votre jeu

1. Ajoutez des **obstacles** sur les bande de terrain sans véhicules pour que le personnage ne puisse pas les parcourir trop librement d'un côté à l'autre de l'écran. Il faut gérer ces obstacles pour que le joueur ne puisse les franchir quand il se déplace latéralement su une telle bande de terrain. Bien sûr, plus le joueur a déjà traversé de tableaux, plus le nombre de ces objets peut être important.
2. Mettez en place un **scrolling** du terrain : constituez un terrain de jeu assez large qui contient plus de bandes que peuvent être affichées à un moment donné sur l'écran et faites en sorte que ce terrain soit affiché de façon défilante. Pour ça, vous pourrez vous inspirer du 2ème projet Greenfoot présent sur Moodle (nommé `3-Asteroids-plus_d_Acteurs`). Attention, quand le terrain de jeu scroll, il faut que tous les acteurs descendent en même temps de façon à ce que chacun reste dans sa travée horizontale (route, bande de terrain, etc).
3. Faites en sorte que si le personnage du joueur avance, il reste toujours dans le quart inférieur de l'écran, ce que vous pouvez obtenir en faisant que le décor défile plus vite à ce moment là.
4. Faites en sorte que le personnage se fasse ramasser par un aigle ou un autre acteur si jamais il atteint le bas de l'écran (en raison du défilement automatique du décor)
5. Mettez en place des bandes d'eau (ou de lave) sur lesquelles des rondins (ou des rochers) flottent, le personnage est autorisé à se déplacer sur ces îlots de sûreté mais pas de se retrouver sur l'eau (la lave)
6. Adaptez votre système de score aux nouvelles conditions
7. Mettez en place un compte à rebours pour chaque niveau : quand le compte à rebours atteint zéro, le joueur gagne un bonus ou un nouveau personnage qu'il peut choisir.
8. Mettez en place des variantes sorties de votre imagination, donnez du fun à votre variante pour la démarquer du jeu original.