Licence Informatique 1ère année

Algorithmique et Programmation

Projet 2016-2017

1 Organisation

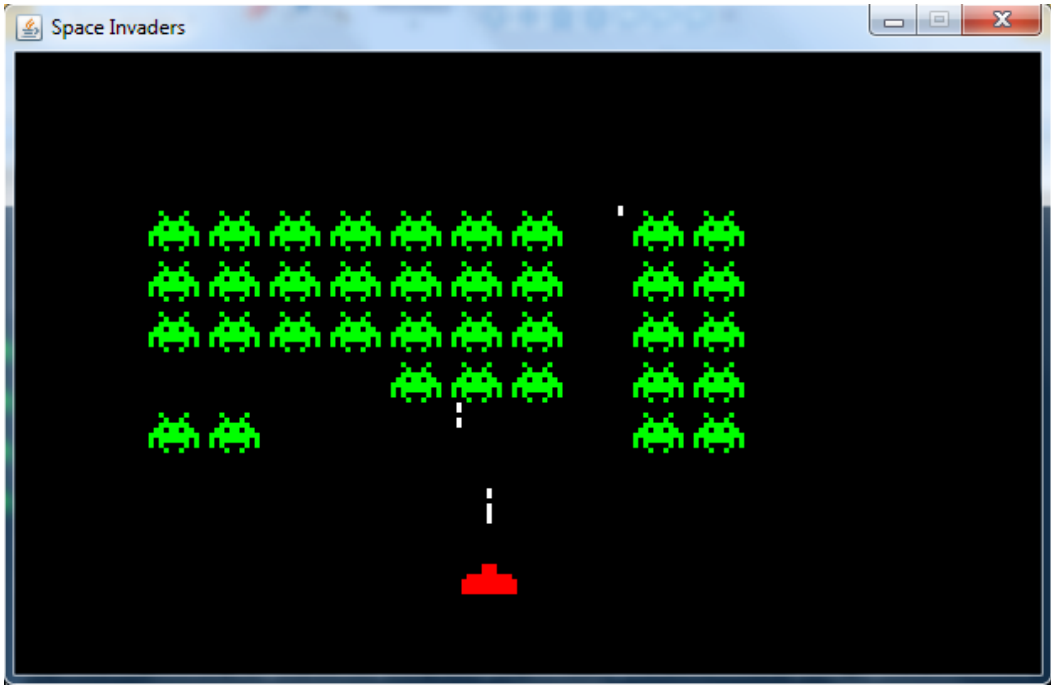
Le projet est à réaliser en binôme. En fin de semestre, les binômes présenteront leur travail aux encadrants de TP. Lors des présentations, chaque binôme rendra un rapport qui présentera le programme réalisé (structures de données utilisées, principe des algorithmes implémentés, ...), les résultats obtenus, les problèmes rencontrés, les solutions apportées et tout élément permettant d'évaluer le travail réalisé. Chaque binôme devra également, au moment des soutenances, envoyer le code des programmes écrits aux responsables de TP (*mariem.benfadhl@u-picardie.fr*, *frederic.furst@u-picardie.fr*, *clement.lecat@u-picardie.fr*, *yu.li@u-picardie.fr*, et *labib.yousef@u-picardie.fr*).

2 Sujet

Le projet consiste à programme le jeu Space Invaders, dont une version est disponible en ligne sur la page freeinvaders.org. Le programme réalisé doit permettre de jouer au jeu avec les règles suivantes :

- les vaisseaux des envahisseurs (en vert) doivent bouger vers la droite et vers la gauche régulièrement et descendre, à un rythme plus faible que les mouvements droite-gauche. Les vaisseaux qui sont les plus bas laissent tomber des bombes de façon aléatoire.

- le canon (en rouge) doit pouvoir être déplacé à droite et à gauche par le joueur et tirer des missiles. Si un missile touche un vaisseau envahisseur, le vaisseau disparait, et le missile aussi. Si le canon est touché par une bombe, soit il reste au moins une vie au joueur et le jeu redémarre au début, soit il ne lui reste plus de vie et le jeu est terminé.



La partie interface graphique n'est pas à votre charge. Une classe Java *InterfaceSpaceInvaders.java* est fournie sur la page du module. Son mode d'emploi est donné plus bas. Vous devez coder tout le reste.

Remarque importante : des programmes de type Space Invaders écrits en Java peuvent surement être trouvés sur le Web, mais ils seront écrits en Java objet. Aucun projet écrit dans le paradigme objet ne sera pris en compte lors des soutenances.

3 Fonctionnalités à réaliser optionnellement

Ces extensions du programme de base apportent des points en plus au projet **à condition que les fonctionnalités obligatoires aient été réalisées**.

- Vous pouvez gérer les données du jeu à l'aide de fichiers. Ces données peuvent concerner les parties (sauvegarde d'une partie en cours, chargement d'une partie sauvegardée) et/ou les joueurs (sauvegarde des meilleurs scores de chaque joueur, de l'historique des parties de chaque joueur, ...).

- Vous pouvez complexifier le jeu, en introduisant des types de vaisseaux différents, avec des comportements différents, en ajoutant des obstacles qui permettent de mettre le canon à l'abri des bombes, etc.

- Vous pouvez introduire un mode multi-joueur où un joueur contrôle un vaisseau envahisseur et l'autre joueur contrôle le canon.

- Vous pouvez ajouter de la musique (voir le tutoriel java sound).

4 Démarche générale pour programmer le jeu

La première chose à faire est de comprendre le jeu, en jouant. Il faut ensuite bien identifier les données à manipuler : quelles données sont nécessaires pour représenter la grille du jeu et les contenus des cases? Une fois les données à représenter identifiées, vous devez choisir des structures de données à utiliser. Par exemple, il est possible de représenter un vaisseau par le point où il se trouve et les positions de chacune de ses cases relativement à ce point. Mais il est aussi possible de le représenter par les positions absolues de toutes ses cases. Cependant, dans ce cas, bouger le vaisseau oblige à modifier les coordonnées de toutes les cases. Il faut donc bien réfléchir à la façon dont on va représenter les données, pour faciliter ensuite l'écriture du programme.

Une fois les structures de données choisies, vous pouvez déjà écrire le code Java correspondant à la déclaration de vos structure et à leur initialisation. Ensuite, vous pouvez utiliser l'interface graphique pour afficher l'état initial du jeu.

L'étape suivante est d'écrire l'algorithme qui fait tourner le jeu. Le mieux est d'y aller étape par étape. Par exemple, commencez par écrire un algorithme qui déplace les vaisseaux envahisseurs de façon régulière, de droite à gauche, et vers le bas. Ensuite, ajouter le mouvement du canon. Puis ajouter la gestion des bombes et des missiles. Finalement, on peut écrire l'algorithme général du jeu avec la détection de la fin du jeu (le joueur a perdu toutes ses vies) et la gestion des points.

Pour gérer la vitesse du jeu, on peut utiliser l'instruction *try{Thread.sleep(n);} catch(InterruptedException e){}* qui met le programme en attente *n* millisecondes.

5 L'interface d'affichage

Du code est fourni pour l'affichage du jeu. Il est obligatoire de l'utiliser. Pour l'utiliser, placez simplement le fichier *InterfaceSpaceInvaders.java* dans le répertoire où se trouve votre programme, vous pourrez alors utiliser dans votre programme les instructions données plus bas. Il est inutile de lire le contenu de cette classe, et encore moins de le comprendre, pour l'utiliser. Les explications qui suivent suffisent pour l'utiliser.

Instruction pour créer une interface : *InterfaceSpaceInvaders toto = new InterfaceSpaceInvaders(l, h,c)* où *l* et *h* sont respectivement la largeur et la hauteur de la grille (en nombre de cases) et *c* est la couleur de fond (une valeur du type énuméré *Color*, par exemple *Color.BLACK*).

Instruction pour changer la couleur d'une case : *toto.modifieCase(x,y,c)* où *toto* est la variable contenant l'interface créée, *x* et *y* (de type *int*) sont les abscisses et ordonnées de la case et *c* est la couleur de type *Color* à afficher dans la case.

Instruction pour effacer une case : *toto.effaceCase(x,y)* où *toto* est la variable contenant l'interface créée, *x* et *y* (de type *int*) sont les abscisses et ordonnées de la case. Cette instruction colore en fait la case avec la couleur de fond définie lors de la création de l'interface.

Instruction pour savoir quelle touche l'utilisateur a tapée : *toto.toucheTapee()* où *toto* est la variable contenant l'interface créée, renvoie un *int* qui vaut 0 pour la barre d'espace, 1 pour la flèche gauche, 2 pour la flèche droite et 5 pour la touche Pause.

Instruction pour afficher un message dans une boite de dialogue : *toto.afficheMessage(m)* où *toto* est la variable contenant l'interface créée et *m* la chaine de caractères à afficher.