Introduction

Ce petit jeu en java est inspiré d'un célèbre jeu d'arcade des années 80 qui porte le même nom: **Space Invaders**.

Le jeu consiste à diriger un petit vaisseau spacial pour défendre sa position. Le but est d'empêcher les envahisseurs de progresser. Ceux-ci avancent progressivement par vagues successives face au vaisseau dont le seul but est de détruire tous les envahisseurs avant qu'il n'arrivent jusqu'à lui.

Présentation

Le jeu n'est pas compliqué, il se compose d'une unique fenêtre. Au lancement du jeu l'utilisateur a juste à appuyer sur la touche *Entrée* comme le message l'indique, pour commencer une partie. Le déroulement est classique, le joueur dirige le vaisseau horizontalement à l'aide des flèches *gauche* et *droite* et tire avec la *barre d'espace*.

Le but est de survivre le plus longtemps possible en détruisant à chaque niveau la totalité des monstres à l'écran. Les envahisseurs ne tirent pas mais si l'un d'eux arrive à la hauteur du vaisseau, la partie est perdue. Le joueur dispose de 5 vies en tout, c'est à dire qu'il peut perdre 5 fois en tout. Plus les niveaux avancent, plus les monstres rapportent des points mais tous les deux niveaux ils accélèrent.
Le nombre de niveaux est donc infini, juste limité par la capacité du joueur à detruire tous les envahisseurs.

Le graphisme des monstres est volontairement sommaire, il est très proche ce celui du Space Invaders original ce qui apporte une petite touche amusante.

Le programme

Le programme est composé de 6 classes:

- **Invaders**: classe principale
- **Element**: classe de laquelle vont dériver tous les objets affichés à l'écran
- **Monstre**: classe dérivant d'**Element** qui permet de construire les envahisseurs
- **Vaisseau**: classe dérivant d'**Element** qui permet de construire le vaisseau du joueur
- **Tir**: classe dérivant d'**Element** qui permet de construire un tir du vaisseau
- **SpriteCache**: classe permettant de mettre en cache toutes les images

Pour gérer les objets qui se répètent à l'écran j'ai choisi d'utiliser des listes, cela facilite l'ajout et la suppression d'éléments pas rapport à un tableau.

Je vais maintenant détailler chaque classe pour faciliter la compréhension du code et des commentaires.

Invaders

La classe principale, elle dérive de Japplet et contient l'essentiel du code. C'est à dire l'initialisation du jeu et de chaque partie. La créaction des groupes d'objets, comme les monstres et les tirs.
Elle comporte aussi la gestion des collisions ainsi que la capture des touches pour les actions du vaisseau. L'affichage de tout ce qui est à l'écran, les messages, les informations sur la partie en cours et bien sur les objets que sont les monstres, le vaisseau et les tirs.

Element

C'est la classe mère des classes **Monstre**, **Vaisseau** et **Tir**. Elle comporte les données de base qui peuvent être attribuées à tous ces objets. Elle peut retourner la position d'un élément sous forme de point.

Monstre

Elle contient le constructeur de l'élément monstre qui est crée à l'aide de coordonnées (x, y) et de la vitesse de déplacement du monstre. Elle contient aussi la méthode **deplacement** qui effectue le déplacement d'un monstre en fonction de deux paramètres: **sens** qui est son sens de déplacement (gauche ou droite) et **descente** qui defini de combien de pixel le monstre doit descendre lors de son déplacement.
Enfin le classe **Monstre** contient la méthode de dessin d'un monstre.

Vaisseau

Cette classe est assez semblable à la classe **Monstre**. Elle possède un paramètre qui lui est propre et qui est la vitesse de déplacement du vaisseau. Le vaisseau est construit à l'aide de trois paramètres: sa position (x, y) et le nombres de vies du joueur, étant donné que le vaisseau représentele joueur.
Comme pour la classe **Monstre**, **Vaisseau** possède une méthode **deplacement** qui prend en paramètre le sens de déplacement du vaisseau. Cette méthode teste si le vaisseau atteint les bord de la fenetre pour maitriser son déplacement.

Tir

La méthode Tir possède un paramètre fixe qui, comme pour **Vaisseau**, défini sa vitesse de déplacement. Un tir est construit à partir de deux paramètres: ses coordonnées (x,y).
Sa méthode deplacement ne prend pas de paramètre et déplace le tir tant que celui-ci est dans la fenetre de jeu. Dans le cas contraire, il supprime ce tir. Ce n'est pas cette méthode qui gère les collisions avec les monstres.

SpriteCache

Cette méthode permet de mettre en cache les images utilisée lors du jeu afin de n'avoir qu'un accès au disque dur pout charger ces images à l'initialisation du programme.
Les images sont stockées dans un buffer, puis, toutes les images buffered sont rangées dans un tableau.
Il y a une méthode se retour par image, qui permet de récuperer une des trois images buffered (**getShipSprite**, **getMonsterSprite**, **getShotSprite**),

Remarques

Les scores ne sont pas enregistrés car cela pose quelques problèmes en tant qu'applet, que je n'ai pu résoudre.
De plus, aucune aide n'est présente dans le programme en java. Le programme étant une applet, le code html ds la mage qui affiche l'applet est bien plus approprié pour ecrire les quelques instructions nécessaires comme les touches pour jouer au jeu.