Jeu de Star Wars
Projet de programmation en Ada
année 2015-2016

1 Le jeu

Vous êtes au commade d'un vaisseau spatial (disons un X-Wing). Pourchassé par les habituels ennemies votre propulsion principale a été endommagé dans les combats. Il vous reste heureusement les propulsions laterales. Vous preferrez alors naviguer dans un nuage d'asteroides au milieu duquel vous serai furtif.
Le jeu commence dans ce nuage d'asteroide avec une vue a la premiere personne. Vous pouvez piloter votre vaisseau sur 2 axes (X et Y) la vitesse sur l'axe Z etant constante (dans l'espace pas de frottement).
Le pilotage se fait a l'aide du clavier, les touches haut et bas pour modifier la vitesse sur l'axe Y et les touches gauche et droite pour modifier la vitesse sur l'axe X.
Chaque niveau durera pendant un temps defini en secondes (disons 1 minute). Vous devrez eviter les asteroides sur votre chemin, une collision etant fatale.

Regles

- si le vaisseau entre en collision avec un asteroide c'est perdu (exception)
- le niveau se fini une fois que le temps defini est ecoule
- pour commencer le niveau le joueur doit appuyer sur la touche espace
- si le niveau est en cours on peut utiliser la touche espace pour entrer et sortir de pause (attention a mettre a jour les references de temps)
- si le vaisseau sort du nuage d'asteroides il se fait immediatement repere par l'ennemi et c'est perdu (exception)
- le temps defini pour un niveau doit etre aussi precis que l'horloge du systeme et ne depend pas des performances de la machine

Afin de simplifier les choses:

- les asteroides sont des spheres.
- le vaisseau est une spheres et n'est pas visible a l'ecran dans la vue a la premiere personne
- la vitesse sur les axes X et Y est limite par un maximum
- le nuage d'asteroide est un cylindre autour de l'axe Z contenant tous les asteroides
- les asteroides ne bougent pas dans l'espace
- il n'y a pas de rotation du vaisseau dans l'espace
- les asteroides sont places dans le nuage d'asteroides aleatoirement
- la taille des asteroides est aleatoire mais limmite par un maximum

Conseils:

Vous devrez optimiser en n'affichant pas les asteroides qui ne sont plus visibles ou qui sont au dela d'une certaine distance (clipping). Vous pouvez aussi optimiser l'algorithme de detection de collisions.
Au dela des regles de base vous avez toute liberte et n'hesitez pas etre creatif pour etendre les fonctionnalites.

2 Le projet
Vous devez utiliser l'API graphique donnée TP et respecter les contraintes ci-dessous :
C.1 Le projet doit être pertinemment découpé en paquetages avec des types privés et procédures et fonctions de taille raisonnable.
C.2 Le vaisseau doit etre implementé à l'aide d'un objet protégé
C.3 Une tache distinct de celle d'affichage se charge de mettre à jour la position du vaisseau en fonction du temps
C.4 Il doit utiliser les exceptions pour signaler la fin du jeu.
C.5 Les paramètres du jeu (par exemple, nombre d'asteroides, vitesse sur l'axe Z, etc.) doivent être facile à changer dans le code et regroupés dans un package.

Bonus : Vous pouvez essayer de prouver a l'aide de SPARK que votre algorithme (optimisé ou non) de detection de collisions est equivalent à la spécification suivante : La distance entre le vaisseau et chacun des asteroides est superieur a la somme du rayon de l'asteroide et du vaisseau.

L' évaluation du projet tiendra compte du nombre de fonctionnalités implémentées, du respect des contraintes de codage ci-dessus, de la robustesse du code, du style de programmation employé (code respectant les contraintes ci-dessus, clair et lisible, bien indenté, avec des commentaires pertinents).

La qualité graphique n'est pas l'objectif ici meme si la forme aide toujours à promouvoir le fond.

3 L'organisation
Le projet doit être effectue en binôme.
Le rendu du projet est constitue du code du projet et d'un rapport.
Le rapport doit être rendu sous format PDF ou HTML.
— Il doit être court (max. 4 pages) et sans code source.
— Il doit expliquer l'architecture de votre projet.
— Il doit expliquer comment utiliser votre programme, y compris comment l'installer si jamais l'utilisateur doit faire quelque chose.
— Vous devez lister les fonctionnalités disponibles et leur état de fonctionnement. Quels bugs sont-ils encore présents ? Où sont ils localisés ? Qu'essayeriez vous de faire pour les corriger si vous en aviez le temps ?
— Le rapport doit indiquer comment vous avez testé votre programme. Un minimum est d'écrire 2-3 tests fonctionnels et quelques contrats (n'oubliez pas d'activer les assertions dans ce cas -gnata)
— Vous pouvez détailler une fonctionnalité dont vous êtes particulièrement fier ou ajouter toute remarque qui vous semble utile.

Dates et éch éances Le projet final est à rendre sur le site Didel du cours au plus tard le 16 février 2016 à 9h59. Il n'y a pas de soutenance de projet.