**420-203-RE – A20**

**Travail pratique numéro 2**

# TP2

# Objectifs du travail

* Utiliser une interface graphique en Java avec Slick2D
* Classe abstraite et/ou interface
* Réaliser un programme interactif
* MVC – Patron observateur
* Peut être fait en équipes de 2 personnes de calibre comparable

## Dates de remise

**Le 11 décembre 2020 : Remise finale du projet sur Léa**

* + Le projet complet de votre programme
  + Une autoévaluation de votre programme dans un fichier remis avec votre projet (ce qui fonctionne très bien, moins bien, pas du tout et pourquoi). Ce doit être un texte complet, même s’il est court.
  + Les jeux d’essais utilisés pour tester votre programme, sous forme de tableau, dans un fichier remis avec votre projet.

*NB. Le travail complet doit être remis sur Léa pour* ***un seul*** *des deux membres de l’équipe.*

# Les colonies terriennes

La Terre n’existe plus. Le méchant Donald l’a mangée. Une poignée de colons d’élite qui venait d’établir une colonie sur Mars. Les super colons ont besoin de matériaux bruts. Ils convertissent leur vaisseau minier du charbon (propre) à l’électricité à l’aide d’une carcasse de voiture électrique rouge lancée en orbite par l’entreprise SpaceXY quelques années auparavant, et convertissent les robots culinaires en broyeurs de minerai. Résolus, ils lancent leur vaisseau minier vers la ceinture d’astéroïdes en espérant y trouver des métaux et de jolies roches pour faire des sculptures, des comptoirs de cuisine, et des presse-papiers.

# Image illustrative de l’article Ceinture d'astéroïdes

Figure 1 Ceinture d'astéroïdes. https://fr.wikipedia.org/wiki/Ceinture\_d%27ast%C3%A9ro%C3%AFdes#/media/Fichier:Asteroid\_Belt-fr.svg

Vous êtes le chef de ces colons de compétition. Et celui qui a tiré la courte paille. Vous pilotez le SS Temp (vous avez trouvé le nom vous-même). Bonne chance.

# Spécifications

## Le SS Temp

Le vaisseau fait 128x128 pixels. Vous déterminez l’apparence exacte du vaisseau, du tir de canon et du conteneur de minerai.

### Déplacements

Vous contrôlez le vaisseau en utilisant les touches WASD.

### Attaques

Le canon de votre vaisseau tire avec la touche Espace. Il est fixe et tire droit devant avec une portée équivalente à la moitié de la hauteur de l’écran.

Le canon est capable de fracturer un astéroïde en deux.

### Collection du minerai

Les broyeurs installés à l’avant du vaisseau peuvent récolter un astéroïde dont la taille est inférieure à celle du vaisseau.

Si le vaisseau essaie de broyer un astéroïde de taille égale ou supérieure à la sienne, il est endommagé. Après trois collisions, le vaisseau est perdu dans l’espace.

Le vaisseau a une capacité de stockage égale à l’aire du vaisseau (128²). Chaque astéroïde récolté a un volume de minerai utilisable égal à la moitié de son aire. Le reste est rejeté durant le traitement initial.

### Envoi du minerai

Lorsque le vaisseau est plein, le vaisseau ne peut plus récolter de minerai. Les broyeurs s’arrêtent et le vaisseau doit envoyer sa cargaison en direction de Mars. Pour lancer la cargaison, le pilote utilise la touche E.

## Astéroïdes

Les astéroïdes se déplacent verticalement du haut vers le bas de l’écran, à des vitesses variables. Beaucoup d’astéroïdes se déplacent également latéralement.

Un astéroïde fait 256x256 pixels. Un tir le brise en deux morceaux principaux de 128x128 pixels, et ainsi de suite.

Les images doivent être chargées à partir d’au moins un spritesheet. Les astéroïdes doivent avoir des aspects variés.

Les astéroïdes n’entrent jamais vraiment en collision. Ils se superposent sur l’axe des Z.

### Monde (écran de jeu)

Devra contenir :

* État des dégâts subis
* L’état de la cargaison
* La cargaison totale envoyée vers Mars
* Des étoiles

Le jeu (monde) est toujours en déplacement vers le bas, pour donner l’impression que le vaisseau avance.

### Détection des collisions

Soyez logique dans la détection de vos collisions, en suivant les conventions de ce type de jeu :

* Un de vos tirs de canon doit fragmenter un astéroïde, mais ce même projectile ne doit pas être en mesure d’endommager le vaisseau.
* Il faut utiliser les interfaces et/ou les classes abstraites dans la détection des collisions entre objets, et le déplacement des objets, pour rendre le programme plus facilement maintenable et évolutif.

## Gestion de la mémoire

Toute entité qui n’est plus utile doit être gérée en mémoire adéquatement… on ne veut pas que le jeu ralentisse au fil du temps.

# Fin de la partie

* Lorsque vous n’avez plus de points de vie (vous avez perdu)
* Autrement la partie est infinie, le but est d’obtenir le plus de points avant de mourir.
* Vous devez permettre de recommencer une nouvelle partie à la fin de la partie en cours (et uniquement à la fin de la partie en cours).

# Autres spécifications

* MVC + patron observateur uniquement pour :
  + le cargo
  + les dégâts
  + le minerai renvoyé sur Mars
* Vous devez afficher en tout temps les informations pertinentes au jeu.
* Aucun menu requis.

### De plus :

* Vous serez noté sur les classes créées, leurs hiérarchies, la portée des variables et méthodes etc. Pensez-y plus tôt plutôt que plus tard.
* Respecter les normes de programmation, et les standards de programmation orienté objet   
  (ex : l'encapsulation des données, la surcharge des constructeurs, etc).
* Vous serez évalué sur l’esthétique et la convivialité de l’application.

# Critères d’évaluation

Votre programme doit respecter les normes de programmation.

Vous devez réaliser ce travail en équipe de 2, sauf exception.

Soumettre un projet NetBeans

|  |  |
| --- | --- |
| Critère | Nombre de  points accordés |
| Document d’accompagnement du programmeJeux d’essaiAuto-évaluation | /2 |
| Esthétique et convivialité  * Groupement logique de composants * Homogénéité entre fenêtres * Sélection judicieuse des composants utilisés * Originalité/Complexité/esthétique des composants graphiques | /3 |
| Programmation  * Clarté, simplicité (utilisation judicieuse des instructions de Java) * Respect des normes de programmation orientée objet (Classes, hiérarchie, encapsulation etc.) * MVC + Patron observateur * Javadoc | /4 |
| Fonctionnement du programme  * Résultats obtenus * Respect des consignes | /11 |
| Total | /20 |

