Travail de cession

Jeu vidéo

Jean-Yves Denoncourt

BlockBall

Table des matières

[1) Documentation 3](#_Toc57717022)

[1-1) Cours 3](#_Toc57717023)

[1-2) Site Web 3](#_Toc57717024)

[2) Modules utilise 3](#_Toc57717025)

[🡺 Collision 3](#_Toc57717026)

[🡺 Distribution normale et Gaussien 3](#_Toc57717027)

[🡺 Contrôles 3](#_Toc57717028)

[🡺 Forces 3](#_Toc57717029)

[🡺 Actions sonores 4](#_Toc57717030)

[🡺 Système de menu 4](#_Toc57717031)

[🡺 Parallaxe 4](#_Toc57717032)

[🡺 Particules 4](#_Toc57717033)

[🡺 State Machine 4](#_Toc57717034)

[🡺 Informations additionnelles 4](#_Toc57717035)

[3) Cheminement 5](#_Toc57717036)

[Suivi #1) Base du jeux (Player et Ball et collision de base, menu) 5](#_Toc57717037)

[Suivi #2) Création de la forme du jeux (blocks et collision avancer) 5](#_Toc57717038)

[Suivi #3) Ajoute de diverse partie pour agrémenter le jeu 5](#_Toc57717039)

[Suivi #4) Continuité dans l’ajout 5](#_Toc57717040)

[Suivi #5) Demande de Nicolas 5](#_Toc57717041)

# 1) Documentation

## 1-1) Cours

**Cours et note de cours en Jeu vidéo.**

Prof : Nicolas Bourre Support : Cours magistrat, PDF et vidéo  
Description :  
 Enseigner les bases de la programmation en jeu vidéo.

## 1-2) Site Web

<https://www.processing.org/>

Explication et descriptif du logiciel Processing. Aide pour trouve les façons de faire, comment bien programmer et rechercher sur le langage (java).

<https://happycoding.io/tutorials/processing/>

Explication pour faire toutes sortes de programmation en Processing (collision, Inputs …)

<http://www.jeffreythompson.org/collision-detection/circle-rect.php>

Explication pour la collision

# 2) Modules utilise

## 🡺 Collision

🡪Entre la balle et les mur cotes et dessus (rebondir)

🡪Entre la balle et le mur du bas (disparait)

🡪Entre la balle et les blocks qui on > 0 leur vie (rebondir)

🡪Entre la balle et la palette du player (rebondir)

🡪Entre la palette et les 2 murs de cote

## 🡺 Distribution normale et Gaussien

🡪Distribution normale (random) pour les blocks et leur nombre de vie (Alive).

🡪Distribution normale (random) pour les couleurs des feux artifices et leurs localisations.

🡪Distribution Gaussien pour la répartition des particules des feux d’artifices et des flammèches.

🡪Distribution normale (random) pour le nombre de vie des blocks (entre 0 et 4 vies).

## 🡺 Contrôles

🡪Il y a les contrôles dans le menu (« enter » pour commencer et « s » aller au setting)

🡪Il y a les contrôles pour le jeu (« souris » pour bouger la palette et « space » pour lancer une balle)

## 🡺 Forces

🡪Force appliquer sur la balle a son lancement et aux collisions

🡪Force appliquer sur les particules de feux d’artifices et de flammèches.

## 🡺 Actions sonores

🡪Bruits de forêts (6 min) qui est une musique d’ambiance tout le long du jeu en boucle.

🡪Bruit de collision (1 sec) qui est active sur collision entre balle et palette / blocks.

## 🡺 Système de menu

🡪Menu de départ qui est lancer en partant le programme ou après une des 2 fins. Il mène au jeu si « enter » est presser ou au menu Setting si « s » est presser.

🡪Menu de fin « game over » qui est lance quand toutes les balles du player sont passer et qu’il reste des blocks. Il mène au menu de départ quand « entrer » est presser.

🡪Menu de fin gagnant qui est lance quand le player a encore des balles et qu’il a détruit tous les blocks. Il mène au menu de départ quand « entrer » est presser.

🡪 Menu Setting qui est lance par le menu de départ en pressant « s ». Il mène au menu de départ quand « entrer » est presser. Ce menu est vide mais il sert de preuve de concept de navigation entre les menus.

## 🡺 Parallaxe

🡪4 images d’arrière-plan sur un thème de foret qui défile selon le déplacement de la palette.

## 🡺 Particules

🡪Particules qui sont émises selon une distribution Gaussienne (10 unités) lors d’une collision entre la balle et la palette. Le but est d’imiter des flammèches, ils sont donc rouges.

🡪Particules qui sont émises selon une distribution Gaussienne (20 unités) lors d’une collision entre la balle et un block seulement quand celui-ci perd sa dernière vie. Le but est d’imiter une explosion du block, ils sont donc la couleur du block.

🡪Particules qui sont émises selon une distribution Gaussienne (20 unités) lors d’une fin gagnante, et ce de façon aléatoire dans l’écran au 2 secondes. Le but est d’imiter une explosion de feux d’artifices, il sont donc de couleur aléatoire selon une distribution normale.

## 🡺 State Machine

Une machine a etat qui se divise en 5 états (c’est un enum avec un switch)

🡪menuStart :

C’est quand on est dans le menu de départ.

🡪menuSetting :

C’est quand on est dans le menu de setting.

🡪inGameOn :

C’est quand on est en jeu et que la balle est en mouvement.

🡪inGameOff :

C’est quand on est en jeu mais que la balle est en attente d’être lance, bref en début de jeu et quand on pert une vie.

🡪endGame :

C’est quand on atteint la fin gagnante, c’est-à-dire que le player a détruit tous les blocks et qu’il lui reste des vies.

🡪gameOver :

C’est quand on atteint la fin perdante, c’est-à-dire que le player n’a pas détruit tous les blocks et qu’il ne lui reste plus vies.

## 🡺 Informations additionnelles

🡪La couleur des blocks

Les blocks ont une vie entre 0 et 4, donc a 0 ils seront déjà explosés. La couleur du block sera en fonction du nombre de vie. C’est la même couleur pour tous et sa part du pale (moins de vie) au fonce (plus de vie). La couleur change quand il perd une vie.

🡪La vie du block

Les blocks ont une vie entre 0 et 4 et ce nombre est afficher dans le block par un carré rouge par vie. Le nombre de carré n’est pas le nombre de vie exactement, c’est plutôt que chaque carré non explosé a une vie et c’est vie supplémentaire sont afficher par un carré par vie. A chaque collision le block perd une vie et de ce fait un carré.

# 3) Cheminement

## Suivi #1) Base du jeux (Player et Ball et collision de base, menu)

* Ok--Implémenter la base de la programmation pour que je puisse ajouter mes classes et implémenter le jeu.
* Ok--Créer la classe Player qui est la barre en bas qui vient faire rebondir la balle. (Position et move)
* Ok--Créer la classe Ball qui est la balle dans le jeu qui rebondira sur les blocks, les cotes et le Player. (Position départ et speed)
* OK--Menu de départ avec affichage des contrôles

## Suivi #2) Création de la forme du jeux (blocks et collision avancer)

* Ok--Creation des blocs random selon grille et de facon aleatoir (pas super esthetique) et en attendant implementation des levels
* Ok--Creation du menu de fin GameOver
* Ok--Creation des collision mur x3 – blocks - player
* Ok--Creation du menu base de fin You win
* Ok--Debug pour rendre le jeu fonctionnel de base
* Ok--Refactor du code en StateMachine avec ENUM et switch
* Ok--Implentation du Parrallax (forest)

## Suivi #3) Ajoute de diverse partie pour agrémenter le jeu

* Ok--Parallaxe fonctionnel avec la souris
* Ok--Sound fonctionnel sur music theme et collision

## Suivi #4) Continuité dans l’ajout

* OK--Ajustement du parallax
* OK--Ajout des particules sur la barre du Player quand collision avec balle
* OK--Ajout des particules sur les blocks quand collision avec balle
* OK--Ajout au nombre de vie d’un block
* OK--Ajout des particules feux d’artifice quand niveau terminer (aleatoire)
* OK--Ajout du score à chaque collision avec bloc(20pts) et destruction de block (100pts)
* OK--Implémenter le menu de fin
* OK--Lancer la fin quand il n’y a plus de block

## Suivi #5) Demande de Nicolas

* OK--Changement sur les particules
  + Les couleurs sur les blocs soit les couleurs des particules et qu’il y a des particules sur les blocks seulement quand ceux-ci explosent.
* OK--Changement les vies des blocks
  + Les vies des blocks soit afficher en cube au lieu d’en nombre.
* OK--Changement sur les couleurs des blocks
  + Couleur passe de random a une teinte verte de plus en plus foncer selon nombre de vie.