

Epic Wheel Hero

Documentation technique

DIVTEC | Gallucci Alex & Steven Halaghia

Table des matières

[Introduction 4](#_Toc52355235)

[Matériel / logiciel 4](#_Toc52355236)

[Matériels nécessaires 5](#_Toc52355237)

[Logiciels nécessaires 5](#_Toc52355238)

[Fichiers nécessaires 5](#_Toc52355239)

[Première Scène : Menu 4](#_Toc52355240)

[Création de la scène : 5](#_Toc52355241)

[Description du Script : 6](#_Toc52355242)

[Deuxième Scène : Jeu 8](#_Toc52355243)

[Les notes 9](#_Toc52355244)

[Disposition 9](#_Toc52355245)

[Créations des scripts 9](#_Toc52355246)

[MouvementRoueG & MouvementRoueD 9](#_Toc52355247)

[Description 9](#_Toc52355248)

[Mis en fonctions des notes 8](#_Toc52355249)

[Les roues 8](#_Toc52355250)

[Le viseur 8](#_Toc52355251)

[Thèmes 10](#_Toc52355252)

[Les vies 11](#_Toc52355253)

[Le score 11](#_Toc52355254)

[Explications 11](#_Toc52355255)

[Implémentation 12](#_Toc52355256)

[Troisième Scène : Classement 13](#_Toc52355257)

[Création de la scène : 14](#_Toc52355258)

[Initialisation du classement : 14](#_Toc52355259)

[Description du Script : 15](#_Toc52355260)

[Suppression d’une entrée dans le classement 15](#_Toc52355261)

# Introduction

Avant de procéder à la lecture de cette documentation technique, assurez-vous d’avoir lu le « Guide d’utilisation » de ce même jeu.

# Matériel / logiciel

## Matériels nécessaires

|  |  |  |
| --- | --- | --- |
| 2x | Guitare Hero pour Xbox One | Une image contenant guitare, assis, table  Description générée automatiquement |
| 1x | Moniteur / Ecran pour l’affichage | Une image contenant moniteur, équipement électronique, télévision, plat  Description générée automatiquement |
| 2x | Adaptateur USB pour Guitare Hero | Une image contenant téléphone mobile  Description générée automatiquement |
| 1x | [OPTIONEL]  Enceinte / Haut-parleur | Une image contenant équipement électronique, moniteur, assis, ordinateur  Description générée automatiquement |
| 1X | PC / laptop  Windows | Une image contenant équipement électronique, moniteur, assis, ordinateur  Description générée automatiquement |

## Logiciels nécessaires

|  |  |  |
| --- | --- | --- |
| 1X | Epic Wheel Hero | https://github.com/divtec-cejef/PROJET-PO-2020-GALLALE-HALASTE |
| 1X | JoyToKey | https://joytokey.net/en/download |

## Fichiers nécessaires

|  |  |  |
| --- | --- | --- |
| 1x | EpicWheelHero.cfg |  |

# Première Scène : Menu



Le menu est la première scène qui se charge quand on lance le jeu.

Il a pour but de :

* Lancer la partie
* Régler la difficulté de la partie.
* Accéder rapidement au classement.

## Création de la scène :

Pour la composition de la scène « Menu » il a fallu créer et disposé tous ces objets :

Une image contenant table

Description générée automatiquement

Attribution des textes et des musiques au script :

Une image contenant table

Description générée automatiquement

## Description du Script :

Le script permet de faire naviguer le joueur 1 dans le menu.

Il navigue entre les différents éléments du menu à l’aide du bouton orange (équivalent à la touche « T »), il déclenche ensuite un switch qui prend en valeur une variable « compteur » initialiser a 1 au début du script, ce qui permet de passer au premier « case 1» :



Dans le « case 1 » on trouve des lignes de code qui permette de changer le style du texte afficher à l’écran en plusieurs étapes :

* Ajoute un effet souligné :



* Si les autres textes sont en souligné alors on enlève ces effets :

Remarque : le code est réadapté pour les autres textes



* Faire apparaitre l’icône « roue verte » :



* Faire disparaitre l’icône « roue orange » et « roue rouge » :



* Jouer la musique « themeFarWest » :



* Ensuite change la valeur de la variable « compteur » :



Ce qui donnera ce résultat :



Remarque : le code (case 2, case 3, case 4) est réutilisé et réadapté

Le joueur valide son choix avec le bouton vert (équivalent à la touche « R »), il déclenche ensuite un switch qui prend en valeur la variable « compteur » (toujours équivalent à l’index de l’élément sélectionnée dans le menu) :



Dans le « case x » on trouve des lignes de code qui permette d’exécuter la scène « Jeu » dans la difficulté choisi ou exécuter la scène classement :

(Exemple pour difficulté « moyen »)



Un dernier « IF » a été créer afin de quitter totalement le jeu en appuyant sur la touche 3 :



# Deuxième Scène : Jeu

## Disposition des notes

Les notes se déplacent de case en case dans le sens des aiguilles d’une montre. Ils apparaissent à la case 5, se déplacent de cases en case jusqu’à revenir à la même case 5 ou ils seront détruits.

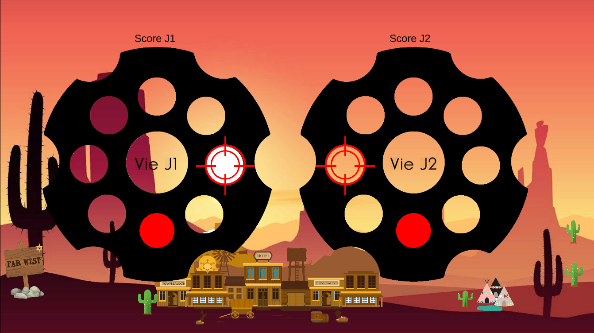
## Créations des scripts

Pour gérer les notes de chaque roue, 2 fichiers scripts seront créés :

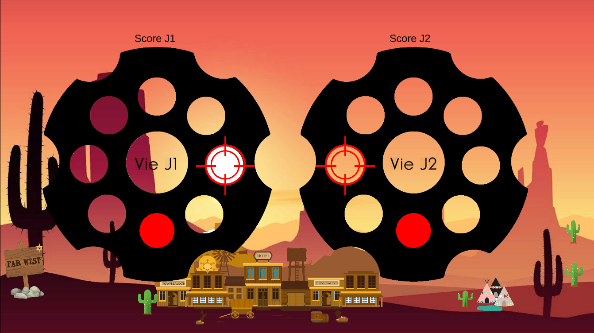
mouvementRoueG.cs et mouvementRoueD.cs.

Une fois les 2 scripts crées,

Le fichier mouvementRoueG.cs est lié à la note de la roue Gauche.



Le fichier mouvementRoueD.cs est lié à la note de la roue Droite.



## MouvementRoueG & MouvementRoueD

### Description

Les principaux scripts du jeu. Ils gèrent presque toutes les fonctionnalités de la partie.

### Mis en fonctions des notes

Les notes sont tout d’abord **déclarées** dans le fichier mouvementRoueG.cs et mouvementRoueD.cs.

L’interface Unity affiche désormais une multitude de nouvelles options. Il faut renseigner quelles images correspond à quelle note.

Une image contenant table

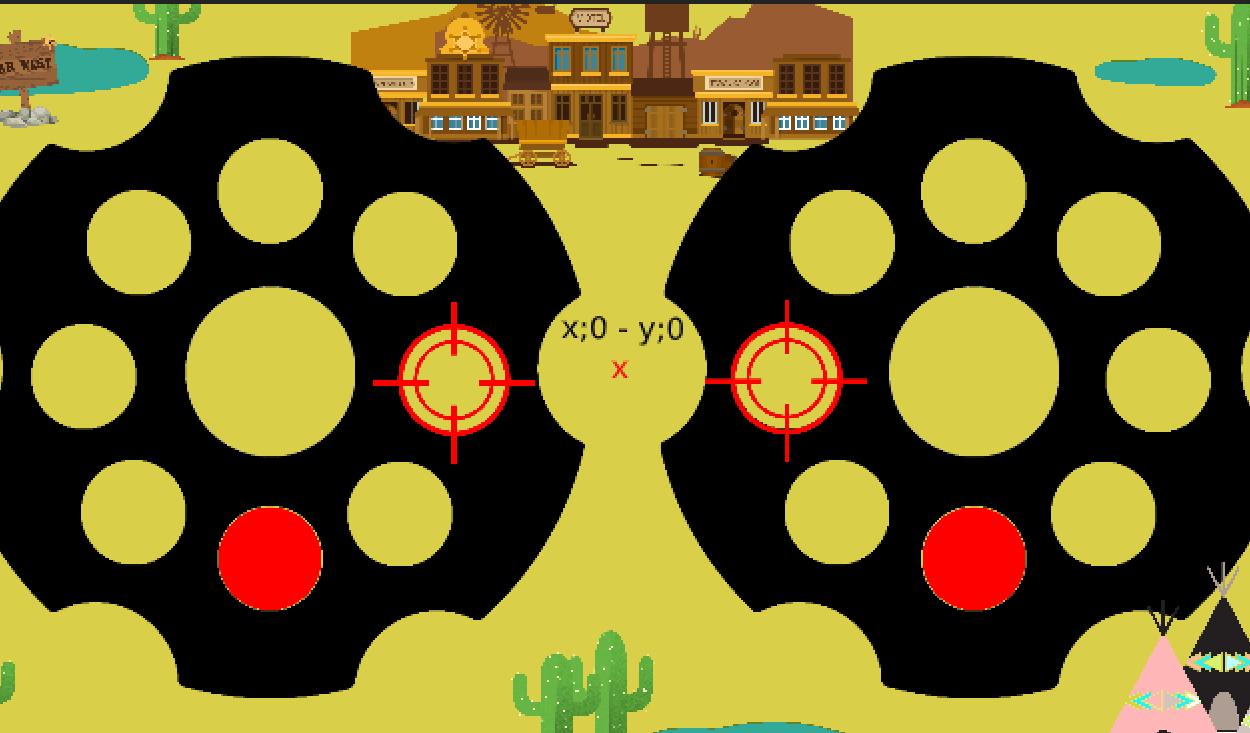
Description générée automatiquement

Pour que les notes puissent se déplacer, elles ont besoins de positions cibles. Pour cela nous allons créer dans chaque cellule de la roue, un objet qui sera rendu invisible.

### Les roues

Les roues représentent des chargeurs de révolver. Elles sont parfaitement fixes, seuls les notes à l’intérieur se déplacent de cases en cases.

Celles-ci sont positionnées de sorte que les points à l’exacte milieu de celles-ci soient x=0, y=0.



Les points entre les roues sont X;0 , Y;0

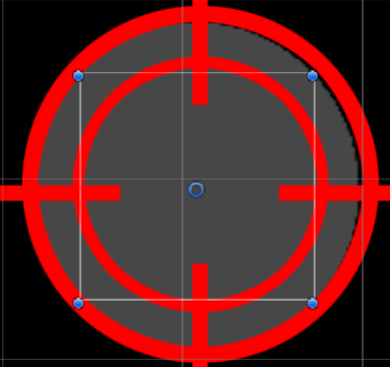
La roue de gauche se positionne aux coordonnées X;6.2 , Y;0.

La roue de droite se positionne aux coordonnées X;-6.2 , Y;0

### Le viseur

Afin de savoir si la note est parfaitement dans le viseur, nous utilisons le système suivant :

4 images invisibles sont disposées à l’intérieur du viseur.



Quand le joueur appui sur une note les évènements suivant se passent :

|  |  |  |
| --- | --- | --- |
| Illustration | Condition(s) | Conséquence(s) |
|  | Les 4 points touchent la note. | La note peut être touchée. Le joueur gagne le score en conséquence.  Le combo s’incrémente en conséquence. |
|  | Moins que 4 points touchent la notes (0-3) | La note ne peut pas être touchée. Rien ne se passe |

Pour chacun des 4 points, un script est assigné. Celui-ci nous retournera si le point est en contact avec un objet (une note dans notre cas).

2 fonctions nous suffirons à l’intérieur de ces script ;



« OnTriggerEnter2D » va être déclenché quand le point va entrer en contact avec un autre objet (pour nous la note).

« OnTriggerExit2D » va être déclenché dès que le point ne sera plus en contact avec un autre objet.

Les script sont nommés de la manière suivante :

triggerP1G, triggerP2G, triggerP3G, triggerP4G pour la roue de gauche.

triggerP1D, triggerP2D, triggerP3D, triggerP4D pour la roue de droite.

Les points sont disposés de la manière suivante :



Point 3

Point 4

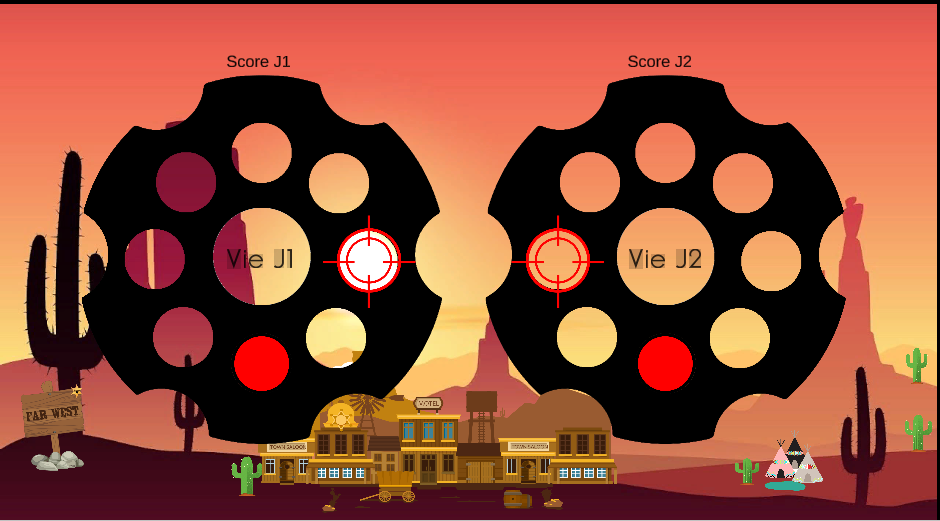
Point 2

Point 1

**La note ne peut être touchée que si les 4 points sont en contact avec celle-ci.**

Notez qu’il suffit d’éloigner les 4 points pour augmenter la difficulté, la note restant moins de temps en contact avec ceux-ci.

### Les thèmes

Far West

Le premier thème du jeu, celui qui s’affiche quand on lance la partie, est un thème Far West

Celui-ci est composé de :

* 1x image d’arrière-plan
* 4x cactus
* 1x panneau en bois
* 1x ville far west
* 1x trio de tepees
* 1x lac

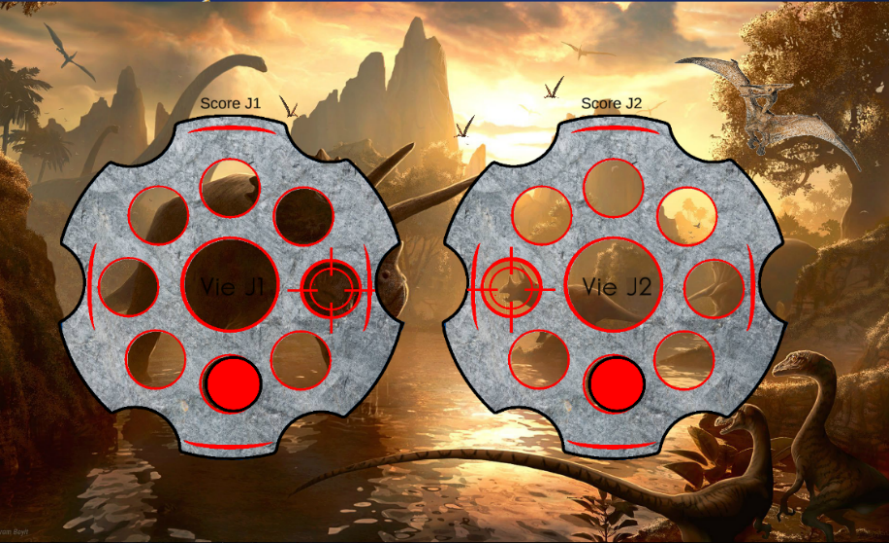


Futuristique

Le second thème, est un thème futuristique

Celui-ci est composé de :

* 1x image d’arrière-plan
* 1x vaisseau spatial
* 1x hélicoptère

Préhistoire

Le dernier thème est un thème préhistoire

Celui-ci est composé de :

* 1x image d’arrière-plan
* 1x grand ptérodactyle
* 4x petits ptérodactyle





### Les vies

Quand la partie commence, chaque joueur possède 3 vies.

Le joueur perd une vie si :

* Il appuie sur la note avant qu’elle n’arrive dans le viseur.
* Il appuie sur la note après qu’elle soit arrivée dans le viseur.
* Il appuie sur une note spéciale « perte d’une vie ». 

La variable « vieValue » contient le nombre de vie sous forme d’entier.

A chaque fois que l’on manipulera les vies, on affectera cette variable.



Notons que « \uE06E » est la représentation du charactère Unicode représentant un cœur. (♡)

### Le score

#### Explications

Un score est constamment affiché à l’écran pour chaque joueur. Pour chaque note « classiques » touchées, le score augmente de 10 points.

Quand le joueur touche plusieurs touchées d’affilée, il déclenche **un combo**.

Celui-ci fonctionne de la manière suivante :

|  |  |
| --- | --- |
| Nombre de notes touchées d’affilée | Score gagné |
| 1 | 10 |
| 2 | 20 |
| 3 | 30 |
| 4 | 40 |
| 5 et + | 50 |

Il n’y a pas de limite maximale au score.

#### Implémentation

Un scripte nommé « scoreScript G.cs » est assigné à la roue de gauche.

Un scripte nommé « scoreScript D.cs » est assigné à la roue de droite.

Les 2 classes sont composées du même code :



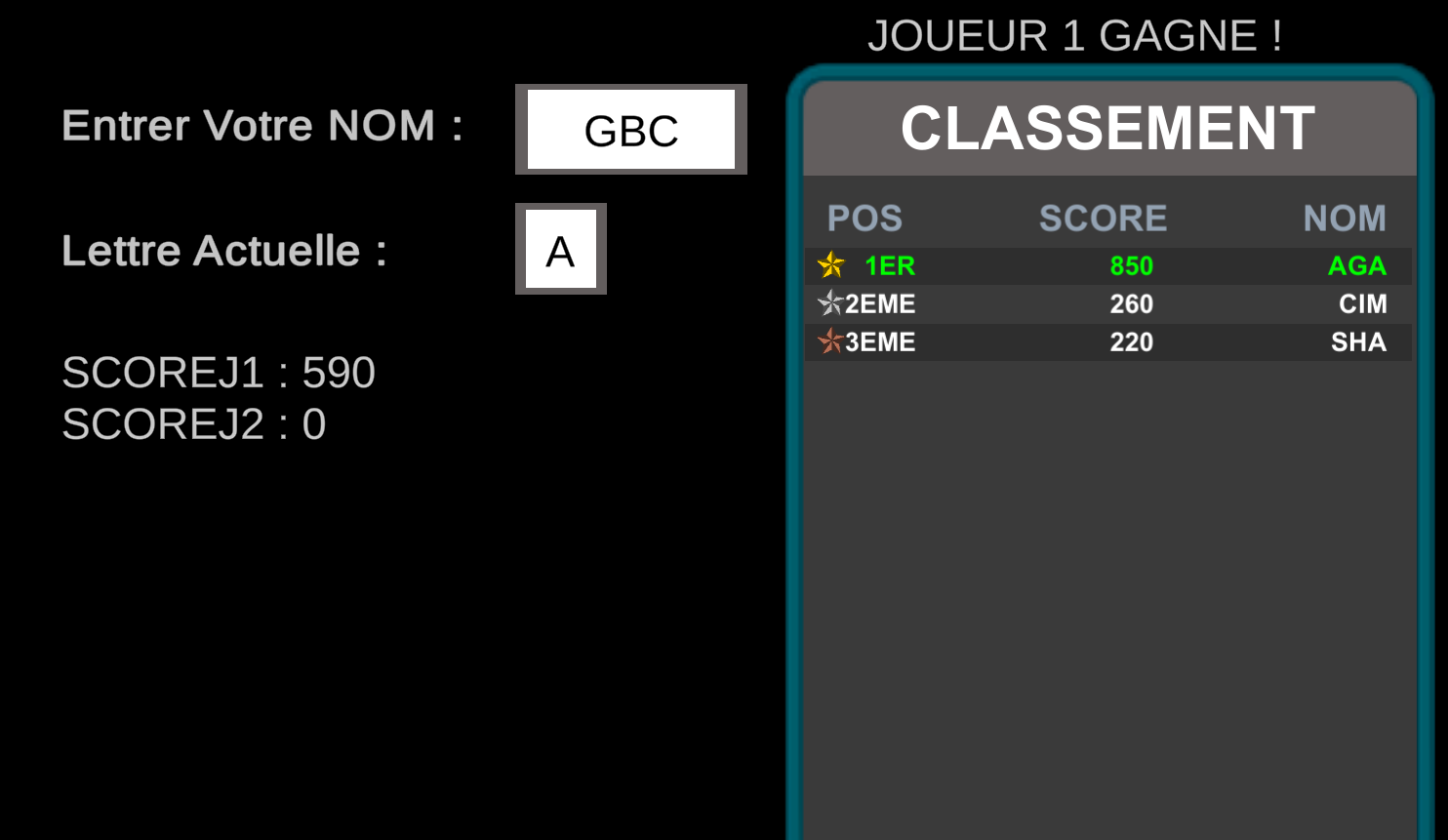
La variable scoreValue contient le score sous forme d’entier.

L’objet score est le score sous forme d’objet textuel, ici de type TextMeshProGUI.

Désormais, a chaque fois que l’on manipulera le score, on utilisera la variable score de ce script -> scoreScriptG.scoreValue = x

Il en va de même pour la roue de droite avec scoreScriptD.scoreValue = x

# Troisième Scène : Classement



Le classement a pour but d’afficher les précédents meilleurs scores.

Quand la partie est terminée, le joueur qui possède le plus de points la remporte et peu inscrire un pseudo de trois lettres.

## Création de la scène :

Pour la composition de la scène « Menu » il a fallu créer et disposé tous ces objets :

Une image contenant table

Description générée automatiquement

Attribution des textes au script :

Une image contenant table

Description générée automatiquement

## Description du Script :

Le script pour la scène classement est composé en deux script :

Le premier est open source, le script gère la partie affichage du classement et sauvegarde du classement.

Lien : https://unitycodemonkey.com/video.php?v=iAbaqGYdnyI

Le deuxième nous l’avons créé afin de pouvoir créer et insérer des pseudos dans le classement.

A la fin d’une partie et pouvoir revenir à la scène « menu » ou quitter totalement le jeu.

Le joueur possédant le score le plus élevé peut inscrire 3 initiales dans le classement, cela passant par plusieurs étapes :

* si une égalité a lieu on choisi au hasard le gagnant :



* si le score du joueur 2 est plus grand que celui du joueur 1 on affiche le gagnant, les scores obtenu durant la partie, la lettre sélectionné du joueur et son pseudo:

Remarque : dans cet exemple le joueur 2 gagne la partie



* si le bouton orange (la touche g) est appuyer on peut sélectionner la lettre voulue :



* si le bouton vert (la touche f) est appuyer on confirme la lettre sélectionner et la liste des lettres se remet au début de l’alphabet:



* si le nom est plus petit que trois lettre alors on peut continuer d’en ajouter :



* si le joueur se trompe il peut effacer son nom en pressent sur la lettre k :



* Si la touche 1 est appuyer et le nom possède 3 lettre alors il est envoyé dans le classement :



* Si la touche 2 est appuyer et on réinitialise toute les variable de la scène « Jeu » et on lance la scène «Menu » :



* Si la touche 3 est appuyé alors on quitte totalement le jeu :



## Suppression d’une entrée dans le classement

Pour supprimer une entrée en particulier, on supprime la ligne dans [la clé du registre](#_Initialisation_du_classement).

Pour réinitialiser le classement, on vide la clé du registre qui les stocks.

## Initialisation du classement :

Les données sont stockées dans la base du registre.

HKEY\_CURRENT\_USER\Software\Unity\UnityEditor\The Epic Wheel Company\Epic Wheel Hero\highscoreTable\_h…

Une image contenant table

Description générée automatiquement