

**Mémoire Projet Manic Shooter : Astral**

Par **Romain Schlotter**

**Mémoire présenté  
en vue de l’obtention d’un Bachelor Européen  
Jeux Vidéo et Serious Games**

**Problématique :  
Le genre manic shooter est très peu présent sur le marché international malgré la présence de ses codes dans plusieurs jeux actuels.**

Année Universitaire 2020/2021

N° de candidat : 106988

Table des matières

[Remerciements 6](#_Toc73895797)

[Introduction 7](#_Toc73895798)

[Présentation du projet 8](#_Toc73895799)

[Intention 8](#_Toc73895800)

[Problématique et question centrale de recherche 8](#_Toc73895801)

[Équipe du projet 8](#_Toc73895802)

[Besoins du projet 9](#_Toc73895803)

[Contraintes 9](#_Toc73895804)

[Prestations attendues pour la production 9](#_Toc73895805)

[Planification initiale du projet 10](#_Toc73895806)

[Analyse du genre 11](#_Toc73895807)

[Introduction 11](#_Toc73895808)

[Les éléments de bases 11](#_Toc73895809)

[Les contrôles 11](#_Toc73895810)

[Les trajectoires 12](#_Toc73895811)

[Les collisions 12](#_Toc73895812)

[Le système de score 13](#_Toc73895813)

[Les boss 13](#_Toc73895814)

[Objectifs 14](#_Toc73895815)

[Conceptualisation du projet 15](#_Toc73895816)

[Point sur l’histoire du jeu 15](#_Toc73895817)

[Concepts mis en avant 16](#_Toc73895818)

[Concept de retour dans le temps 16](#_Toc73895819)

[Retour dans le temps 16](#_Toc73895820)

[Les éléments 17](#_Toc73895821)

[La bombe de dernière chance 18](#_Toc73895822)

[Système de boss et de cantique 18](#_Toc73895823)

[Système de Score 19](#_Toc73895824)

[Point par objet 19](#_Toc73895825)

[Zone de Collecte 19](#_Toc73895826)

[Bonus de Cantique 19](#_Toc73895827)

[Destruction d’attaque 20](#_Toc73895828)

[Frottement 20](#_Toc73895829)

[Points par absorption 20](#_Toc73895830)

[Fin de niveau 20](#_Toc73895831)

[Fin du jeu 20](#_Toc73895832)

[Gestion des ressources 21](#_Toc73895833)

[Continue ? 21](#_Toc73895834)

[Niveaux de difficulté 21](#_Toc73895835)

[Facile 22](#_Toc73895836)

[Normal 22](#_Toc73895837)

[Difficile 22](#_Toc73895838)

[Divin 22](#_Toc73895839)

[Contrôles 23](#_Toc73895840)

[Légende 23](#_Toc73895841)

[Conception de niveau 24](#_Toc73895842)

[Thème du Niveau 24](#_Toc73895843)

[Idée Forte 24](#_Toc73895844)

[Points d’Orgues 24](#_Toc73895845)

[Contexte 25](#_Toc73895846)

[Impacte de la réussite sur l’histoire 25](#_Toc73895847)

[Dialogue d’introduction 25](#_Toc73895848)

[Liste des évènements 26](#_Toc73895849)

[Progression déroulée 26](#_Toc73895850)

[MoodBoard 27](#_Toc73895851)

[Plan du niveau 27](#_Toc73895852)

[Indications supplémentaires 27](#_Toc73895853)

[Réalisation du projet 28](#_Toc73895854)

[Choix des outils 28](#_Toc73895855)

[Problématiques 29](#_Toc73895856)

[Réalisation 29](#_Toc73895857)

[Les éléments magiques 29](#_Toc73895858)

[Le système de retour dans le temps 29](#_Toc73895859)

[Gestion de projet 29](#_Toc73895860)

[Conclusion 30](#_Toc73895861)

[Résultat obtenu 30](#_Toc73895862)

[Analyse critique 30](#_Toc73895863)

[Perspectives 30](#_Toc73895864)

[Apports personnels 30](#_Toc73895865)

[Annexes 31](#_Toc73895866)

[Placements des touches sur différentes plateformes 31](#_Toc73895867)

[Clavier 31](#_Toc73895868)

[PS5 31](#_Toc73895869)

[XBOX 32](#_Toc73895870)

[Switch 32](#_Toc73895871)

[MoodBoard du premier niveau 33](#_Toc73895872)

[Plan de conception du premier niveau 34](#_Toc73895873)

[Bibliographie 35](#_Toc73895874)

[Lexique 36](#_Toc73895875)

# Remerciements

# Introduction

L’un des premiers shoot’em up est Space Invaders en 1978, précédé par Spacewar! en 1962, ils sont suivis par une séries de jeux d’arcades entre les années 80 et 2000, entre 2001 et 2003 le légendaire Ikaruga prend la relève, puis le genre du shooter 2D en vue du dessus va se faire de moins en moins présent sur le marché.  
Aujourd’hui des jeux comme Nier : Automata en 2017 ou Returnal en fin Avril 2021 reprennent des codes du shoot’em up mais surtout du manic shooter.

Ces dernières sorties de jeu et la réédition d’Ikaruga en 2018 sur Nintendo Switch montrent que ce genre de jeu n’est pas totalement tombé en désuétude.

C’est pourquoi dans le cadre de mon projet Bachelor je tiens à tenter ma chance en répondant à cette question : « Comment remettre au goût du jour le genre manic shooter vieillissant ? ».

Afin de répondre à cette question j’ai réalisé un manic shooter au cours de cette année scolaire en reprenant les codes du genre et en y ajoutant des solutions actuelles, qui n’étaient pas disponible à l’époque des bornes d’arcades, pour le moderniser.

Nous commencerons par la mise en place du projet de son idée d’origine jusqu’au plan de réalisation. Puis nous nous attaquerons aux différents écueils et succès inattendus qui ont eu lieu durant la réalisation de ce jeu. Et enfin nous étudierons le résultat et le comparerons avec ce à quoi nous nous attendions en début de projet.

# Présentation du projet

## Intention

Les codes du manic shooter sont toujours présents dans les jeux vidéo actuels, mais le genre se fait de plus en plus rare sur le marché. En produisant un jeu du genre avec les moyens actuels et en essayant de le sublimer, nous essaierons de comprendre pourquoi ces jeux se font plus rares tout en cherchant un moyen d’actualiser le genre sans le dénaturer.

## Problématique et question centrale de recherche

La problématique à laquelle nous allons tenter de répondre :   
Le genre manic shooter est très peu présent sur le marché international malgré la présence de ses codes dans plusieurs jeux actuels.

La question principale du mémoire est la suivante :  
Comment remettre au goût du jour le genre manic shooter vieillissant ?

Pour répondre à cette question nous allons procéder par étape :   
-Étudier les anciens jeux du genre qui ont réussi à faire leur place sur le marché.  
-Voir quels codes du manic shooter parviennent encore à se glisser dans les jeux actuels.  
-Conceptualiser un jeu à partir de ce que nous avons appris de tout cela.  
-Réaliser ce jeu avec nos moyens actuels.  
-Étudier le résultat et conclure sur notre question initiale.

## Équipe du projet

Romain Schlotter, 22 ans, étudiant en Bachelor à l’école Ludus Académie.

## Besoins du projet

Le jeu sera développé en utilisant le moteur Unity sous sa version LTS (Long Time Support) la plus récente. Le code source sera hébergé sur GitHub. Le jeu sera réalisé en français et une traduction anglaise est prévue.

Les illustrations seront en 2D, pour mettre en valeur le principe du rideau de balles du manic shooter, l’univers du jeu se voudra plutôt sombre pour mettre en avant les tirs et motifs à l’écran qui seront la source de lumière principale du jeu.

## Contraintes

Le jeu sera développé sur Unity pour pouvoir être porté sur le maximum de plateformes à savoir : PC, Mac, Switch, PlayStation et Xbox.  
Dans le cas d’une publication, l’intégralité du jeu doit être libre de droit ou possédé par le développeur.  
La réalisation de ce jeu se fait sans budget initial et par une seule personne dans le cadre de cette recherche.

## Prestations attendues pour la production

Nous avons pour but de rendre un jeu comprenant :

* Un menu.
* Un tableau des scores.
* Une scène des options pour changer la langue et les contrôles.
* Un écran de lancement du jeu.
* Une scène pour le cœur du jeu.
* Une portabilité possible sur un maximum de plateformes

## Planification initiale du projet

**Début du projet :** 16 décembre 2020.

* **12/2020 :** Début du projet après sa validation.
* **01/2021 :** Validation des concepts et éléments formels du jeu.
* **02/2021 :** Prototype jouable et vérification des contrôles sur les diverses plateformes.
* **03/2021 :** Préparation de la maintenabilité du code de la première version jouable.
* **04/2021 :** Finalisation du premier niveau jouable en version gold.
* **05/2021 :** Itération pour la création des niveaux suivants.
* **06/2021 :** Finalisation du projet, de la documentation et rendu.

**Fin de projet :** Juin 2021

# Analyse du genre

## Introduction

Pour réaliser un manic shooter qui va reprendre les codes des précédents jeux du genre, il faut tout d’abord déterminer comment ce genre de jeu est fait.  
Nous allons voir ça en définissant les éléments de bases du jeu, puis en définissant les contrôles du joueur. Après nous allons voir les motifs et trajectoires qui apparaissent ainsi que les collisions des différents éléments. Enfin nous allons nous pencher sur le système de score et la partie la plus importante de ces jeux qui est les boss.

### Les éléments de bases

Les éléments de base des shmups sont : le personnage joueur, les ennemis apparaissant successivement à l’écran, les tirs du joueur, ceux des ennemis, les bonus et malus que le joueur peut récupérer et les boss qui possèdent des motifs de tirs plus complexes que les ennemis de base.  
Sur l’interface on peut retrouver, la vie du joueur, sa puissance de feu actuelle, ses bombes, son score actuel, le meilleur score enregistré, l’état actuel de la partie et d’autres informations relatives au jeu et à l’état de la partie comme la position du boss en abscisse en bas de l’écran pour aider le joueur à se repérer.

### Les contrôles

Le personnage du joueur peut en général bouger dans les 4 directions, auxquels se rajoutent les diagonales de manière libre sans effet de gravité ou d’accélération.  
Le personnage passe donc instantanément du déplacement à vitesse maximale à l’arrêt et inversement.  
Il existe aussi souvent un mode concentré où le personnage se déplace lentement pour effectuer des mouvements plus précis.

Les tirs du joueur peuvent être continus ou non, nécessiter d’appuyer sur un bouton ou non.  
Notamment pour les « shmup » sur mobile et tablette, la disparition de l’action de tir qui devient automatique permet d’améliorer la maniabilité d’un jeu qui demande à être précis sur un support qui ne permet que difficilement cette précision.

Après les tirs il y a les options de changement de mode de tir qui sont soit automatique quand on ramasse un bonus ou un malus, soit peuvent s’effectuer sur une action du joueur.

Enfin il y a les bombes qui permettent en général de nettoyer l’écran et de sortir de situations complexes, mais tout le temps en quantité limitée.

### Les trajectoires

En supposant que les trajectoires complexes sont soit basées sur des points de passages soit des composés de trajectoires simples, on peut identifier trois types de trajectoires de base pour les tirs, motifs et déplacements du joueur et des ennemis :  
-La trajectoire en ligne droite où l’objet possède une direction, un point de départ et une vitesse qu’il gardera tout au long de cette trajectoire.  
-La trajectoire circulaire où l’objet décrit une courbe autour d’un mobile en fonction de sa vitesse.  
-La trajectoire ciblée qui est une reprise des premières trajectoires, mais en fonction de la position d’une cible, joueur ou ennemi.

Avec ces trajectoires de bases, ainsi qu’une gestion de points de passages, on peut dessiner n’importe quelle figure à l’écran pour pimenter le défi du joueur qui est de survivre sous cette pluie de projectiles.

### Les collisions

Dans ces jeux où les éléments actifs peuvent atteindre plusieurs centaines à l’écran, il est important pour le joueur de les différencier en un regard, et de connaitre les interactions lors de la collision de ces éléments.

En règle générale ils sont les suivants :  
-Le joueur pouvant récupérer les bonus, se blesser sur les attaques ennemies et les ennemis et générer des attaques.  
-Les ennemis pouvant se blesser sur les attaques du joueur, blesser le joueur, générer des attaques et des bonus.  
-Les attaques ennemis pouvant blesser le joueur.  
-Les attaques du joueur pouvant blesser les ennemis.  
-Les bonus pouvant être récupérés par le joueur.

### Le système de score

Dans ce genre de jeu qui s’est développé sous le format arcade, il est important de tenir un compte du score du joueur et de valoriser certaines manières de jouer plus risquées pour garder une certaine re jouabilité et permettre au joueur de sentir une amélioration dans ses compétences en jeu.  
Les actions récompensées sont en général les suivantes :  
-Vaincre des ennemis.  
-Esquiver les attaques ennemies.  
-Jouer à un niveau de difficulté plus élevé.  
-Récupérer les bonus.

Ces actions peuvent être bien plus développées selon le jeu et ce que l’on cherche à récompenser.

### Les boss

Le principe de boss que l’on trouve dans ces jeux est l’un des plus simples.  
Au milieu des niveaux ou à la fin, ils sont là pour ponctuer l’expérience du joueur en reprenant les codes et obstacles qu’il a pu découvrir dans le niveau et en lui présentant à un niveau supérieur.  
La musique change, l’interface s’adapte et tout est fait pour mettre à l’épreuve le joueur.  
Enfin quand il est vaincu les récompenses arrivent, le niveau suivant se débloque, etc…  
L’important est que cette expérience soit mémorable et tranche avec le reste du jeu.

## Objectifs

Les genres « shmup » et « manic shooter » sont relativement simples dans leur construction, on contrôle un personnage qui va avancer dans le jeu et vaincre les ennemis sur son chemin jusqu’à atteindre un boss, puis le vaincre et passer au niveau suivant.

Le but de ce projet est donc de reprendre ces codes simples, d’y ajouter un concept qui me semble pertinent et de produire un jeu qui se veut contemporain.  
Durant cette production il faudra réévaluer et tester ce que je peux produire à mon échelle, et ce qui peut être produit au niveau d’une production actuelle. Puis voir s’il est pertinent de pousser le projet davantage.  
Si c’est le cas, c’est qu’il est possible de remettre au gout du jour le genre manic shooter avec de nouveaux concepts.  
Sinon c’est que ce genre de jeu devient aujourd’hui une base pour d’autres jeux mais ne se suffit plus à lui-même pour se faire un nom sur le marché.

Pour ce faire je vais tout d’abord détailler le concept que j’ai établi pout ce projet, ensuite je vais présenter sa réalisation, puis conclure et répondre à notre question centrale de recherche qui est : « Comment remettre au goût du jour le genre manic shooter vieillissant ? ».

# Conceptualisation du projet

## Point sur l’histoire du jeu

Astral : La chute des Dieux

J’ai conçu l’histoire de ce jeu à partir d’un jeu de rôle que j’ai animé en 2015 sur forum, en voici un synopsis :

Sur le continent monde de Treskri vivent magiciens, monstres, dragons et dieux.  
Nous nous situons moins d’une année après la trêve d’une guerre qui mena à la déchéance de presque tous les dieux qui tentèrent d’arrêter les mortels dans leurs querelles intestines.  
Cette défaite est preuve de faiblesse des divinités et un ancien contrat avec les mortels les force à laisser tomber leurs attributions et accepter leur fin.  
La plupart abandonnèrent leur statut et purent vivre au sol avec les mortels.  
Mais certains d’entre eux résistèrent incapable d’accepter ce sort.

C’est alors qu’intervient Meraili, une archimage dont les capacités ne sont plus à prouver.  
Dans l’histoire d’origine elle part déchoir les dieux renégats avec une armée de mercenaires, dans ce jeu, étant le personnage du joueur, elle ira seule et affrontera en séquence chacun des renégats en retraçant le chemin sur l’ancien champ de bataille qui a été pris dans le jeu de rôle.

Les renégats qu’elle affrontera donc sont, dans l’ordre :  
-Haurach, déesse du destin.  
-Vutha, déesse de la nuit et de l’obscurité.  
-Svant, dieu du ciel et des saisons.  
-Marfedelom, dieu de la mort.  
-Ixen, dieu du feu.  
-Ithquent, dieu représentant la divinité.

Au début de chaque niveau, ainsi qu’avant et après chaque boss de fin de niveau aura lieu un dialogue court de mise en situation.  
On y découvrira également le personnage de l’Auteur qui servira à présenter le décor au joueur sans jamais interagir avec les personnages diégétiques.

## Concepts mis en avant

Pour m’approprier le genre du « manic shooter » et l’adapter à l’histoire que je compte narrer dans ce projet, j’ai réalisé plusieurs concepts :

## Concept de retour dans le temps

En tant qu’idée phare pour mettre en avant les technologies actuelles, concepts difficiles à mettre en place à l’époque des bornes d’arcade et convenir à l’histoire écrite, voici le « Retour dans le temps ».

## Retour dans le temps

Pour valoriser une manière de faire du score, aider les joueurs en difficulté à passer une zone difficile et rendre ce jeu unique, le joueur pourra faire un retour dans le temps de cinq secondes en arrière lorsqu’il active une bombe de l’élément arcane, ou une bombe de dernière chance.  
Les concepts d’élément et de bombe de dernière chance seront développés juste après.  
Ce retour dans le temps est quasi-total, tout revient comme il y a cinq secondes à l’exception de trois choses, la position du joueur, son score et son stock de bombes qui lui diminue.  
Puis lorsqu’il atteint le moment où il était, une bombe s’active et nettoie l’écran des attaques ennemies.  
Cela lui permet de passer outre le moment où il était en difficulté et de cumuler à nouveau du score dans le passage de cinq secondes qu’il a refait.  
Ce concept est important car il va revenir à chaque moment critique où le joueur active une bombe, et tous les éléments du jeu doivent pouvoir revenir cinq secondes en arrière sans chargement pour ne pas briser le « flow » du jeu.  
Il a également pour but de différencier le projet par rapport aux précédents jeux du genre et de créer son identité sur le marché.

## Les éléments

Dans l’univers d’Astral les magiciens usent de différents éléments magiques avec des affinités les uns pour les autres.  
Afin de reprendre l’idée de divers tirs possibles pour le personnage joueur et de la réappliquer avec une idée sympathique d’affinité reprise d’Ikaruga, voici le concept des éléments.

Dans ce jeu il y a cinq éléments, trois principaux qui sont : le feu, le bois et la terre, et deux neutres qui sont l’arcane et le divin.

Chaque personnages, joueur ou ennemi, absorbe les attaques du même type que l’élément qu’il arbore actuellement.  
Le joueur peut arborer les éléments : arcane, feu, bois et terre.  
Les ennemis les éléments : divin, feu, bois et terre.

Les attaques de l’élément feu infligent plus de dégâts sur les ennemis de l’élément bois, celles de bois infligent plus de bois sur ceux l’élément terre, et celles de l’élément terre infligent plus sur ceux de l’élément feu.

Avec cette roue élémentaire et la capacité de changer d’élément à tout moment, il sera demandé au joueur de porter le bon élément au bon moment pour absorber les attaques ennemies ou leur infliger des dégâts augmentés.

Il est bon de savoir qu’à l’exception des boss, tous les personnages invoquent des attaques du type qu’ils portent actuellement.

Le joueur aura un motif de tir différent selon l’élément qu’il arbore, sa bombe également sera relative à l’élément actuelle.

## La bombe de dernière chance

Le système de bombe ne différent aucunement aux autres jeux du genre, je vais l’éluder.  
Le joueur a un stock de bombes qu’il peut activer et lorsqu’il le fait il réaliser une attaque large touchant une grosse partie de l’écran, le nettoyant au passage des attaques ennemies.

Le principe de bombe de dernière chance est un peu différent.  
Lorsque le personnage joueur se fait toucher par une attaque, ce qui aurait pour conséquence de lui faire perdre une vie, ce principe lui offre une fenêtre courte, d’environ un quart de seconde, pour appuyer sur le bouton d’activation d’une bombe.  
La mort du personnage joueur est ainsi annulée et la bombe d’arcane qui effectue un retour dans le temps est activé.  
En contrepartie, deux points de bombes sont consommés dans le procédé, si il en reste moins au joueur, mais toujours plus que zéro, tout est consommé.  
Si le joueur n’a plus de bombes il n’a pas la possibilité d’activer cette bombe de dernière chance.

Ce concept a pour but de permettre au joueur de corriger une erreur de placement et de récompenser son réflexe d’appuyer sur une bombe pour éviter de se prendre un coup.  
Cette bombe permet également de faire apparaitre le principe de retour dans le temps même si le joueur ne revient pas sur l’élément arcane pour l’activer.

## Système de boss et de cantique

Sur le même principe que les jeux du genre, le jeu change drastiquement quand un boss arrive. Une barre de vie apparait en haut de l’écran et le boss lance des attaques impressionnantes qui mettent le joueur à mal.  
Lorsque la vie du boss atteint zéro, il passe à sa phase suivante, ses motifs changent, et il faut à nouveau réduire à néant sa barre de vie jusqu’à ce qu’il soit définitivement vaincu.

Le système de cantique, à l’instar du système de « spellcards » des jeux Touhou Project, est une phase particulière du boss où il va effectuer un motif particulier d’attaque unique, et qui offre un bonus de score s’il est passé sans bombe et sans perdre de vie par le joueur.

Système de Score

### Point par objet

C’est le score actuel du joueur affiché sur l’interface, il augmente en récupérant les objets qui tombe dans l’ère de jeu.  
Vaincre les ennemis ne donne pas directement de point, les « chaines » d’Ikaruga ou Dompachi ne sont donc pas possibles, mais ils laissent tomber des objets donnant du score et de la puissance de feu.  
Gagner des points dans un niveau permet également de récupérer bombes et vies à certains paliers.

### Zone de Collecte

Le quart haut de l’écran agit comme un point de collecte de tous les objets de l’écran si le joueur va se concentrer là-bas, cela ramène tous les objets au personnage joueur.  
Le haut de l’écran est un lieu dangereux, les ennemis s’y rassemble pour se disperser en bas, c’est également là que le boss se déplace quand il est là et que les attaquent sont les plus denses.  
Y aller et appuyer sur la touche qui ralentit le personnage est donc récompensé par la collecte de tous les objets à l’écran.

### Bonus de Cantique

Quand un boss effectue un « cantique » l’une de ses attaques spéciales, elle a une durée.  
Si le boss perd suffisamment de vie pendant le cantique il prend fin et le joueur récupère un bonus de points équivalent à la durée restante du cantique.  
Perdre une vie ou utiliser une bombe durant un cantique annule la possibilité de recevoir le bonus à la fin de ce dernier.

### Destruction d’attaque

Lorsqu’un cantique est terminé ou que le joueur lance une bombe, toutes les attaques ennemies à l’écran sont détruites, pour chacune d’elle le joueur gagne des points.

### Frottement

Quand le joueur est sous un élément sensible à une attaque qui le frôle, il gagne des points.  
Cela est fait pour inciter les joueurs qui veulent faire du score à se rapprocher du feu sans le toucher.

### Points par absorption

Lorsque le joueur absorbe des attaques d’un élément il gagnera des points à la fin de la chaine d’absorption selon le nombre de points absorbés.

### Fin de niveau

À la fin d’un niveau, la difficulté actuelle du jeu ainsi que le bonus de complétion du niveau s’ajoutent au score total de la partie en cours.  
Au début du niveau suivant le score du niveau repasse à zéro, notamment pour les bonus de bombes et de vies, et le score total est écrit au-dessus.

### Fin du jeu

Le nombre de vies et de bombes restantes vont servir de multiplicateurs au score actuel pour récompenser les joueurs qui sont arrivés jusqu’au bout du jeu.  
Perdre durant le jeu donne un score qui n’obtiendra pas ce multiplicateur.

### Gestion des ressources

Tout au long de la partie, quand le joueur va chercher à faire le meilleur score possible, la gestion des vies et des bombes sera différentes que s’il cherchait simplement à terminer le jeu.  
Par exemple perdre une vie ne réduit pas le score actuel durant le niveau et cela permet de recharger quelques bombes qui peuvent permettre de détruire plus d’attaques à l’écran. Dans cette idée laisser certains ennemis vivre plus longtemps pour qu’ils puissent attaquer avant de lancer une bombe peut également être une bonne idée.  
Le frottement est une autre prise de risque souhaitée du joueur.  
Et perdre une vie ou utiliser une bombe durant un cantique est aussi quelque chose à éviter quand on joue pour le score mais de normal quand on souhaite juste finir le jeu.

### Continue ?

Lorsque le joueur a perdu toutes ses vies il peut utiliser un « continue » pour continuer là où il en est, cependant son score de niveau et son score total retourneront à zéro.  
Le nombre de « continue » disponibles est fixe durant toute la partie et relatif au niveau de difficulté.

## Niveaux de difficulté

Pour que le jeu soit accessible à tous et reste intéressant pour les vétérans de ce genre de jeu, différents niveaux de difficultés seront proposés.  
En plus de changer le nombre d’ennemis et d’attaques qui apparaissent à l’écran, soit la difficulté pure, différentes variables sont amenées à changer :

### Facile

Multiplicateur de Score : x1.0  
Présence du niveau extra : Non  
Départ avec 5 vies et 3 bombes.  
Bombes réinitialisées à 3 après la perte d’une vie.  
Nombre de « Continue » : 5

### Normal

Multiplicateur de Score : x1.2  
Présence du niveau extra : Non  
Départ avec 5 vies et 3 bombes.  
Bombes réinitialisées à 3 après la perte d’une vie.  
Nombre de « Continue » : 4

### Difficile

Multiplicateur de Score : x1.5  
Présence du niveau extra : Oui  
Départ avec 4 vies et 3 bombes.  
Bombes réinitialisées à 3 après la perte d’une vie.  
Nombre de « Continue » : 3

### Divin

Multiplicateur de Score : x2.0  
Présence du niveau extra : Oui  
Départ avec 4 vies et 2 bombes.  
Bombes réinitialisées à 2 après la perte d’une vie.  
Nombre de « Continue » : 0

## Contrôles

Le jeu est prévu pour être joué sur le maximum de plateformes possibles, pour ce faire il faut prévoir les contrôles sur ces-dites plateformes.

Note : Le téléphone portable qui était envisagé en tout début de projet a été retiré des plateformes cibles à cause du manque de précision que l’on peut avoir sur cette plateforme et du manque d’exemples contraires.  
Le jeu « DodonPachi Unlimited » qui aurait pu être un bon exemple, sur Android et iOS, a été retiré des plateformes car il n’a pas fonctionné comme souhaité.

Voici donc le placement des touches par défaut prévu pour Astral :

### Légende

1 : Déplacement vers le haut.

2 : Déplacement vers le bas.

3 : Déplacement vers la droite.

4 : Déplacement vers la gauche.

5 : Passer en Concentration.

6 : Tirer.

7 : Bombe.

8 : Passer à l’élément suivant.

9 : Passer à l’élément précédent.

10 : Revenir à l’élément Arcane.

Note : Ordre des éléments Feu 🡪 Bois 🡪 Terre.

[Schéma des manettes en annexe.](#_Placements_des_touches)

## Conception de niveau

Durant la réalisation de ce projet, je vais réaliser le premier niveau prévu, voici le détail de son concept.

### Thème du Niveau

Meraili entre dans une forêt dense pour accomplir sa mission de déchoir les dieux renégats.  
Le chapitre commence, notre héroïne entre en scène et ses adversaires ne sont normalement pas encore au courant qu’elle arrive, mais la forêt n’est pas très hospitalière envers les étrangers.

Éléments importants : Premier niveau, forêt dense, élément Bois très présent.

### Idée Forte

Quasi-omniprésence de l’élément bois à l’exception du boss de milieu de niveau qui n’est pas un simple élémentaire de bois et le boss de fin qui sera le premier Dieu à affronter, qui vont surprendre les joueurs essayant déjà de jouer sur les affinités élémentaires.

### Points d’Orgues

L’apparition du premier semi-boss Caesin une elfe de la forêt.  
**La confrontation avec le premier Boss Haurach (lu « Orak ») la Déesse du Destin.**

### Contexte

Une légende annonçait la déchéance des Dieux quand les mortels les surpasseront, les évènements passés ont mené à la disparition et l’acceptation de leur déchéance de la plupart des Dieux.  
Seul Ithquent un dieu majeur représentant l’idée même de la divinité s’est opposé à cette fin.  
Une poignée de dieux mineurs l’ont rejoint dans sa rébellion et ensemble nul ne s’est ce qu’ils préparent.  
Bekilip, la déesse de la Destruction réduit à néant un village dans lequel se trouvait de passage Meraili suite à une prédiction d’Haurach la déesse du destin.  
Cet acte mena au début de la quête de Meraili d’étouffer dans l’œuf les risques que représentent ces dieux renégats et affaiblis par la foi vacillante des mortels.  
Pour atteindre le repère des renégats, le chemin le plus sûr se trouve par l’orée de la forêt des elfes.  
C’est donc ici que la quête de notre héroïne commence.

### Impacte de la réussite sur l’histoire

Meraili parvient à déchoir sa première cible et à traverser la forêt en direction du Champ de Bataille qui a vu d’innombrables combats entre les Royaumes et l’Empire.  
Et ainsi se rapprocher de l’instigateur de cette rébellion.

### Dialogue d’introduction

« Ainsi la grande Archimage suivant une quête de feu la couronne des trois Royaumes s’engage dans l’ancienne forêt des elfes espérant atteindre le repère des dangereux renégats prêts à laisser leurs ouailles mourir pour retrouver un semblant de leur puissance d’antan. »

### Liste des évènements

-Le monologue d’introduction :  
L’Auteur, un personnage externe à l’intrigue et à ce qu’il s’y passe qui ne fait que décrire la situation, cela permet aussi de lancer le jeu après une action du joueur.  
-L’apparition du Boss de Milieu de Niveau :  
Tout en continuant d’avancer Meraili va se faire attaquer par des flèches qui viennent des côtés de l’écran, elles seront d’élément bois, puis une elfe apparaitra pour lancer quelques grosses attaques d’élément terre avant de repartir une fois vaincue.  
Son apparition est brève et est censée surprendre le joueur.  
-Premier Boss :  
Un dialogue va interrompre la progression du joueur et le terrain aura changé.  
Haurach se dresse face à Meraili, la déesse muette ne pipe mot face à la magicienne qui essaie de régler ça par la parole. Jusqu’à voir le bâton que porte l’oracle divin.  
Les mots que la déesse n’a jamais pu prononcer ne seront pas utile lors du crépuscule de son existence divine.  
S’en suit le combat contre le premier boss, conclu par la victoire du joueur et quelques mots de l’Auteur.

### Progression déroulée

-Meraili arrive dans la forêt sur une introduction de l’Auteur.  
-Elle y trouve des élémentaires de bois qui l’attaquent sur sa route.  
-S’ensuit une attaque préventive d’une elfe de la forêt qui cherche à la chasser.  
-Puis son chemin continue jusqu’à une clairière où tout se calme et l’image d’Haurach se dessine.  
-Après un entretien infructueux le combat commence contre la Déesse du destin.  
-Le niveau se conclu sur le dernier chant d’une déesse muette et notre héroïne continue sa route.

### MoodBoard

[MoodBoard du premier niveau présent en annexe.](#_MoodBoard_du_premier)

### Plan du niveau

[Plan du premier niveau présent en annexe.](#_Plan_de_conception)

### Indications supplémentaires

Ennemi Inférieur : Ne peut pas lancer d’attaque dans ce niveau.

Ennemi de base : Peut lancer les attaques une par une dans ce niveau.

Ennemi supérieur : Peut lancer des attaques un peu plus complexes et a plus de vie.

Il s’agit du premier niveau dans le mode de difficulté le plus simple.

Quant à la durée : Le niveau dure environ 1 minute 40 secondes et le boss de fin dure de 1 minute à 1 minute et 20 secondes.  
Le niveau complet dure donc environ 3 minutes.

# Réalisation du projet

## Choix des outils

Pour réaliser ce projet il faut un moteur de jeu, parmi les différents moteurs existants j’ai décidé de partir sur le moteur Unity.  
J’ai choisis ce moteur car il est en accès libre pour les projets étudiants tels que celui-ci et également parce que j’ai déjà travaillé avec ce moteur durant mes années scolaire précédentes.  
Unity permet également de compiler un projet unique et de le distribuer en tant qu’exécutable sur une multitude de plateformes :  
Windows, Mac, Linux, Android, iOS, Nintendo Switch, PlayStation, XBOX, et d’autres…  
Cela permet de répondre à une des problématiques du projet qui est la facilité de diffusion de celui-ci.

Pour la réalisation des assets graphiques j’ai utilisé Sketchbook, simplement parce que je possède le logiciel et que j’ai l’habitude de travailler avec.  
Il ne s’agit probablement pas de l’outil le plus adapté pour réaliser des produits graphiques pour un jeu, mais n’étant que novice dans le domaine il n’est pas pertinent de pousser davantage dans ce sens.

Pour la réalisation des assets audios j’ai utilisé FL Studio pour sa version étudiante, étant également novice dans le domaine, j’utilise un logiciel qui m’a été conseillé par un camarade de classe.

## Problématiques

La problématique du projet est la suivante :

**« Le genre manic shooter est très peu présent sur le marché international malgré la présence de ses codes dans plusieurs jeux actuels. »**

Ce qui nous amène à la question :

« Comment remettre au goût du jour le genre manic shooter vieillissant ? »

Pour répondre à cette question nous allons réaliser le projet conceptualisé plus tôt et selon cette réponse nous serons à-même d’expliquer la problématique soulevée.

## Réalisation

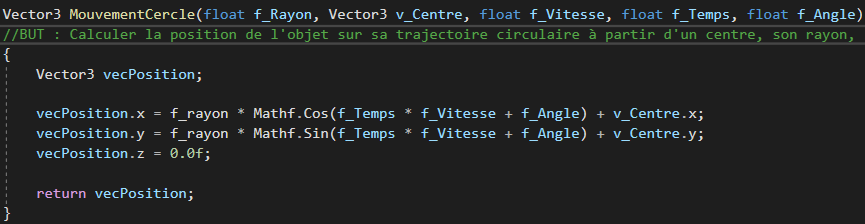
### La classe Minuteur

Dans ce projet on souhaite avoir un contrôle sur le temps qui s’écoule et pouvoir le remonter pour notre concept principal.  
Il faut donc créer un script en « singleton » que je vais attacher à l’objet GameManager dans Unity, et qui va garder une trace du temps actuel.  
Cette classe héritant de « MonoBehaviour » possède deux variables membres :  
-Un booléen stockant l’état marche/arrêt du minuteur.  
-Un réel « double » qui stock le temps mis à jour dans la fonction surchargée « Update » en s’incrémentant de la variable système Time.deltaTime tant que le booléen précédent est vrai.  
Enfin elle possède des mutateurs et assesseurs publiques pour que tous les scripts du projet puissent accéder au temps ou le changer le cas échéant.  
Note importante : Cette classe est en « singleton », cela signifie qu’il n’y a qu’une seule instance de cette classe à la fois, afin de prévenir des erreurs pouvant provenir de l’existence de deux minuteurs simultanés.

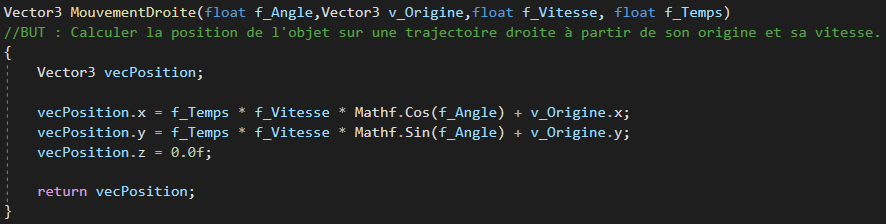
### La classe Trajectoire

De nombreux éléments vont se déplacer indépendamment les uns des autres, pour les gérer il leur faut un script de déplacement.  
Pour simplifier le code des différentes trajectoires qui auront lieu dans le jeu, j’ai créé une classe mère abstraite Trajectoire qui va posséder les variables et fonctions de base dont auront besoin les trajectoires qui seront créées.  
Nous avons donc une classe abstraite nommée Trajectoire héritant de « MonoBehaviour ».  
Cette classe possède une référence à l’instance du Minuteur créé plutôt et placé sur l’objet GameManager.  
Elle possède également quatre autres variables membres :   
-Un réel de type « float » pour stocker la vitesse de l’objet.  
-Un réel de type « double » pour stocker l’instant de naissance de l’objet.  
-Un vecteur 3D pour stocker la position de départ de l’objet.  
-Un réel de type « float » pour stocker le temps personnel actuel de l’objet.  
Ainsi que deux fonctions virtuelles protégées :  
-Une fonction « Start() » qui initialise les variables de la trajectoire.  
-Une fonction « Update() » qui met à jour le temps de l’objet à chaque « frame ».

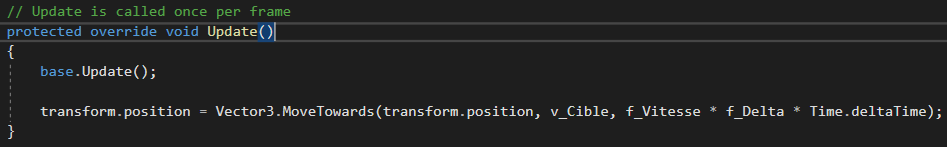
À partir de cette classe mère, trois classes ont été créées pour réaliser divers trajectoires :

  
La classe TrajectoireCirculaire implémentant la fonction MouvementCercle comme suit :

Cette fonction permet de calculer la position de l’objet à partir de son temps, sa vitesse, son angle, son centre et son rayon sur une trajectoire circulaire.

La classe TrajectoireDroite implémentant la fonction MouvementDroite comme suit :

Cette fonction permet de calculer la position de l’objet à partir de son temps, sa vitesse et son origine sur une trajectoire droite.

La classe TrajectoireChercheCible surchargeant uniquement la fonction Update() comme suit :

Ici nous utilisons la fonction « MoveTowards() » d’Unity déplaçant notre objet vers sa cible, la variable f\_Delta que l’on découvre ici est un multiplicateur pour la vitesse de l’écoulement du temps du Minuteur, quand le temps s’écoule normalement elle vaut 1, quand il est en pause elle vaut 0 et les autres cas de vitesse d’écoulement du temps reste possible au besoin.

Avec ceci on a les trajectoires de bases que peut supporter un objet.

### La classe Motif

### La classe Ennemi

### La classe Chemin

### La classe Joueur

### Les vagues

### Le script Niveau

### L’interface utilisateur

### Les dialogues

## Gestion de projet

### Outils de gestion de projet

Les outils de gestion de projet que j’utilise sont github pour le versioning et Ludus-net pour la planification précise tâche par tâche.  
Github : <https://github.com/Irthir/Projet_Bachelor>   
Ludus-Net : <https://ludus-net.fr/>   
En décembre 2020 j’ai rédigé le cahier des charges du projet, sa note de cadrage et réalisé un tableau d’antériorité ainsi que son graphe MPM, la planification qui suit est le résultat de ceci.

### Planification révisée du 10-05-2021

**Début du projet :** 16 décembre 2020.

* **12/2020 :** Début du projet après sa validation.
* **01/2021 :** Validation des concepts et éléments formels du jeu.
* **02/2021 :** Prototypage et vérification des contrôles sur les diverses plateformes.
* **03/2021 :** Recherche et développement sur les éléments clefs du projet.
* **04/2021 :** Préparation de la maintenabilité du code de la première version jouable.
* **05/2021 :** Mise en place du premier niveau jouable en version gold.
* **06/2021 :** Finalisation du projet, de la documentation et rendu.

**Présentation et rendu du projet :** 23 juin 2021.

* **07/2021 :** Itération pour la création des niveaux suivants.
* **08/2021 :** Itération pour la création des différentes difficultés.
* **09/2021 :** Démarchage d’artistes pour la reprise des assets du jeu.
* **10/2021 :** Supervision de la création de ces assets.
* **11/2021 :** Réalisation des tests de qualité du jeu.
* **12/2021 :** Corrections à la suite des tests.
* **01/2021 :** Préparation de la première sortie du jeu.

**Première sortie officielle du projet :** Janvier 2022.

* **02/2021 :** Veille et correction du jeu à la suite de la sortie.
* **03/2021 :** Étude de faisabilité quant aux portages et aux traductions potentielles.
* **04/2021 :** Réalisation des traductions retenues.
* **05/2021 :** Réalisation des portages retenus.

**Fin du projet :** Juin 2022.

# Conclusion

## Résultat obtenu

## Analyse critique

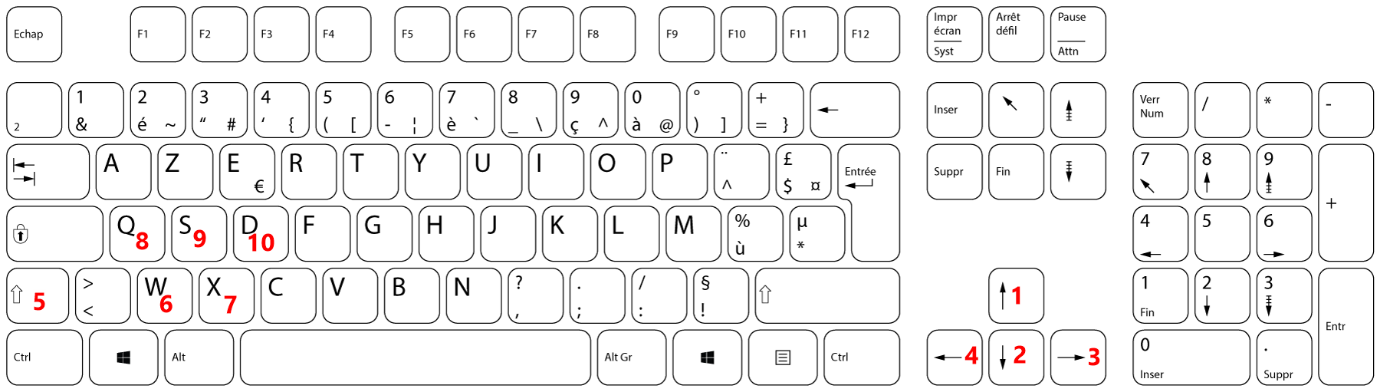
## Perspectives

## Apports personnels

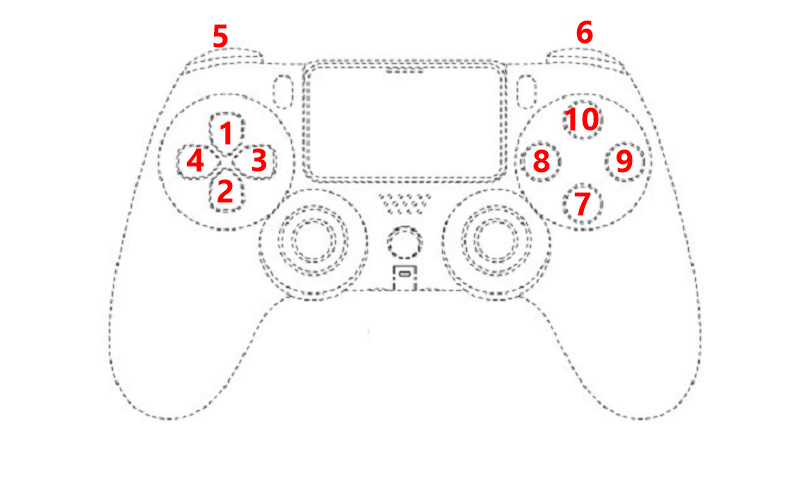
# Annexes

## Placements des touches sur différentes plateformes

### Clavier



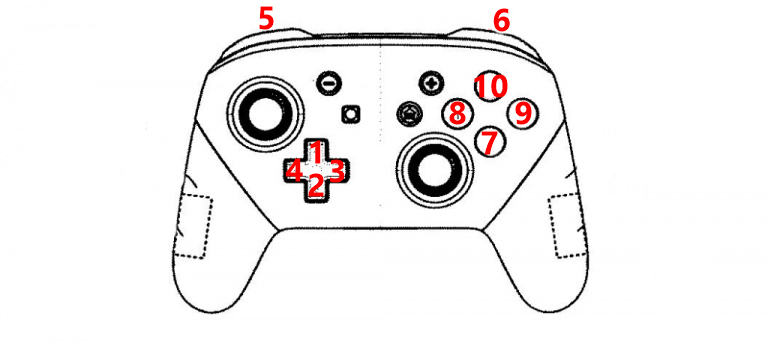
### PS5



### XBOX

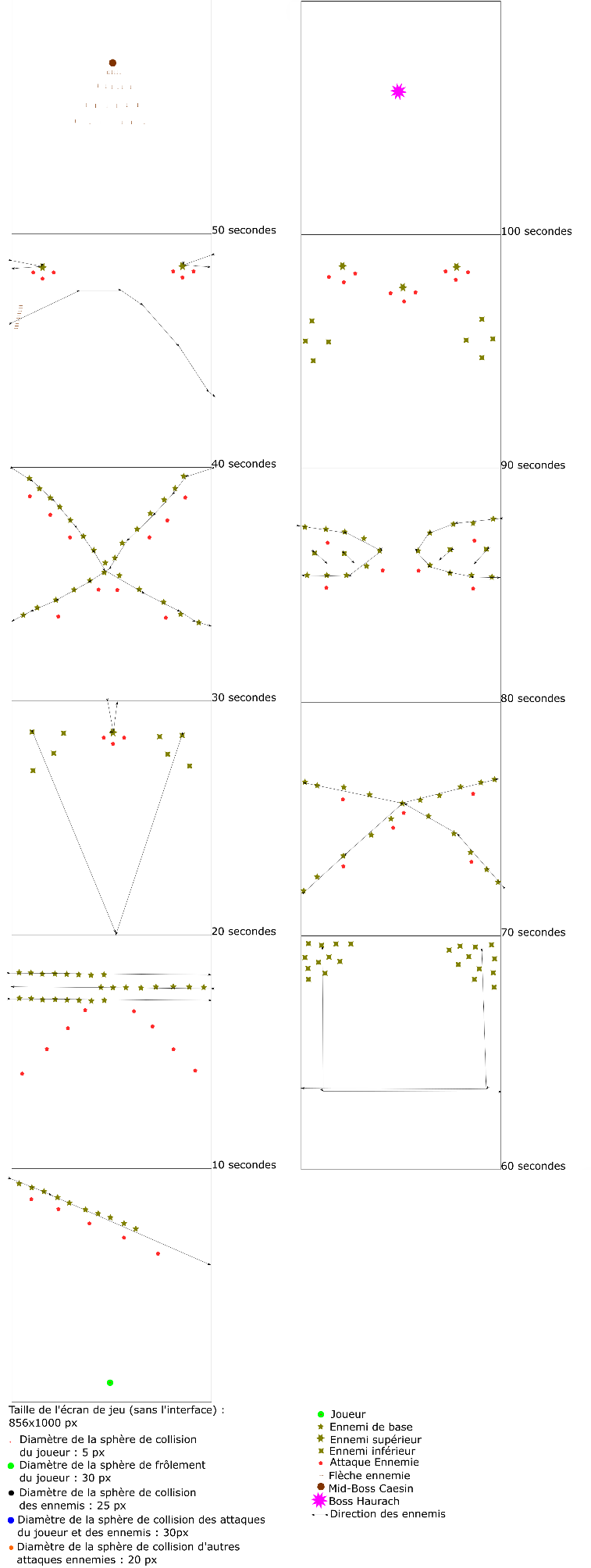


### Switch



## MoodBoard du premier niveau

## Plan de conception du premier niveau



# Bibliographie

Shoot’em Up. Wikipédia. <https://fr.wikipedia.org/wiki/Shoot_%27em_up> dernière modification datant du 01/02/2021.

Du shoot’em up : Considérations techniques.  
Dernière modification datant du 01/09/2018 par Simon Albou

<https://medium.com/@simonalbou/du-shootem-up-consid%C3%A9rations-techniques-1-3-f080e76c5c71>  
<https://medium.com/@simonalbou/du-shootem-up-consid%C3%A9rations-techniques-2-3-fa429d3f4fd6>  
<https://medium.com/@simonalbou/lusage-de-compute-shaders-sur-unity-un-gain-de-performance-violent-54c1b0f72698>   
<https://medium.com/@simonalbou/du-shootem-up-consid%C3%A9rations-techniques-3-3-a7c018612ab7>

Différences de motifs : « Macrododging/Micrododgind »  
Dernière modification datant du 24/03/2020 par Andrew Fan

<https://sparen.github.io/ph3tutorials/ddsga4.html>

Courbes mathématiques :  
Dernière modification datant du 01/12/2020 par Robert Ferréol  
<https://mathcurve.com/courbes2d/courbes2d.shtml>

# Lexique

Shoot’em up : Provenant de l’anglais « Shoot them up » est un genre de jeu vidéo où le personnage joueur doit détruire de nombreux ennemis avec une arme de tir tout en survivant et en avançant dans les niveaux.

Manic Shooter : Également connus sous les noms de « Bullet Hell » ou « Danmaku », soit « Enfer de balles » ou « Rideau de balles », c’est un sous-genre du shoot’em up où l’écran est couvert des tirs ennemis et où la marge de manœuvre du joueur dépend entièrement des motifs de tirs à l’écran.