BLUNIER Hugo

Catch the Fruits - Note d'intention

Dans le Git

Le Repository GitHub est divisé en trois parties :

- Cette note d'intention, séparée des autres dossiers ;
- Le projet Unity, dans le dossier « BLUNIER_Hugo_Catch_The_Fruits » ;
- La build fonctionnelle du projet, dans le dossier « CatchTheFruitsBUILD ».

Le jeu

Le jeu se joue avec les touches Q et D du clavier (A et D sur un clavier QWERTY). La touche E permet également de dash.

L'interaction avec les boutons du menu se fait avec la souris.

Le joueur contrôle la barre violette située en bas de l'écran. Celle-ci peut attraper les trois types d'objets du jeu :

- Pommes : Celles-ci rapportent un point si elles sont ramassées, et font perdre une vie si elles sont manquées.
- Bombes : Celles-ci font perdre une vie au joueur si elles sont ramassées, et n'ont aucun effet si elles sont manquées.
- Poires : Celles-ci rapportent trois points et régénèrent un point de vie (sans excéder le maximum) si elles sont ramassées, et n'ont aucun effet si elles sont manquées.

Ces éléments de jeu apparaissent selon une valeur aléatoire tirée à chaque spawn, de 1 à 6. Si 1 est tiré, une bombe apparaît ; si 6 est tiré, une poire apparaît. Dans tous les autres cas, une pomme normale est instanciée.
Le dash du joueur est une accélération de vitesse durant 2 secondes, après lesquelles l'input doit être pressé de nouveau. Je n'ai volontairement pas mis de cooldown à ce dash, optant à la place pour un boost de vitesse moins puissant pour un équilibre entre un jeu de précision (esquiver les pièges qui apparaissent s'ils bloquent le chemin vers des fruits) et de vitesse (pour attraper des fruits à portée lointaine). Toutefois, le dash ne peut être initié si le boost de vitesse est déjà en cours, pour éviter son spam.

Le joueur possède 5 points de vie. À mesure que la partie avance, le spawn des éléments de jeu accélère (partant d'une base de 2 secondes) de 0.25 secondes par trois spawns, jusqu'à un minimum de 0.75 secondes de délai par spawn, rendant le jeu suffisamment rapide pour

être challengeant. Toutes les valeurs citées sont modifiables dans l'éditeur, avec des headers pour faciliter leur visibilité.

La vitesse du joueur et toutes les valeurs relatives au dash sont modifiables via le PlayerCharacter. Des UnityEvents assurent également la perte et le gain de vie des pièges et fruits bonus.
Toutes les valeurs relatives au spawn (délai de spawn, zone de spawn, probabilités d'apparition d'un piège et d'un fruit bonus) sont modifiables dans le GameManager.
Les points de vie du joueur (valeur actuelle et maximum) ainsi que leur perte via le manquement d'un fruit sont modifiables via l'ElementDestroyer.

L'UI est fonctionnelle et met à jour la santé et le score du joueur. À 0 points de vie, la partie s'arrête et le joueur retourne sur le menu principal.

Ce qu'il manque au jeu

→ Limiter le spawn des éléments de jeu pour éviter qu'ils ne soient trop loin les uns des autres. C'est une difficulté majeure que j'ai eu en programmant ce jeu, je n'ai pas réussi à atteindre un résultat satisfaisant, et la feature n'est pas utilisable.

→ Corriger un bug sur le déplacement du joueur. Lorsque le joueur récupère un élément de jeu, parfois, si l'élément est récupéré sur les côtés de la barre, le mouvement du joueur va légèrement se bloquer sur la direction d'origine, l'empêchant d'être immobile. Parfois, le bug se résout tout seul une fois un autre élément récupéré ; certainement un problème de collision, mais que je n'ai pas réussi à résoudre.

→ Utiliser l'InputPlayerActions : C'est certainement la partie de programmation qui me pose le plus problème, car je n'arrive tout simplement pas à le faire fonctionner (même en revoyant les cours, des exemples d'autres projets ou des tutoriels extérieurs, c'est quelque chose que je n'ai jamais réussi à faire fonctionner). Je compte reprendre ce point pendant l'été car ça me pose beaucoup problème, j'ai utilisé l'alternative du GetKey comme sur mes projets précédents, mais ce n'est pas satisfaisant.