Guide du module : Auditorium

Introduction

Dans ce deuxième projet, nous allons nous pencher sur la 2D et la gestion du son dans Unity.

Les éléments que nous allons étudier dans ce projet sont :

- **AudioClip** [Manual] [Scripting Reference]
- **AudioSource** [Manual] [Scripting Reference]
- **AreaEffector2D** [Manual] [Scripting Reference]
- **Rigidbody2D** [Manual] [Scripting Reference]
- **BoxCollider2D** [Manual] [Scripting Reference]
- **Raycasts** [Manual] [Scripting Reference]

Objectif

L'objectif de ce projet est de recréer quelques niveaux du jeu Auditorium créé par le studio **Cipher Prime** en 2009, voici la version jouable en ligne **https://cipherprime.com/games/auditorium/**.

Depuis janvier 2021, il n'est plus possible de jouer à la version flash en ligne. Il est possible tout de même d'y jouer grâce au logiciel Flashpoint téléchargeable ici : **https://bluemaxima.org/flashpoint/downloads/** . Attention, l'archive dézippée prend près de 2 Go de place sur le disque dur. Il est ensuite possible de rechercher Auditorium et d'y jouer.

Les éléments indispensables de l'exercice sont :

- Générer des particules avec une certaine force
- Créer les boîtes musicales qui réagissent au passage des particules
- Créer les zones permettant de modifier la trajectoire des particules (on les appellera *effector*)
- Pouvoir manipuler les effectors à la souris :
    - drag'n drop sur le centre pour les déplacer
    - drag'n drop sur le bord pour les redimensionner
- Coder la condition de victoire : toutes les boîtes musicales sont allumées en même temps pendant un certain temps.

Progression

1. Générer les particules
2. Créer les boîtes musicales
3. Créer les effectors
4. Coder l'interaction avec les effectors
5. Coder la condition de victoire

Depuis Unity Hub, commencer par créer un nouveau projet avec le template 2D. Puis ajouter les assets fournis sous forme de UnityPackage, celui-ci contient des sprites, des sons, et un petit script qui vous aidera à générer des cercles à l'écran.

Générer les particules

Le modèle de particule

Ici nous n'utilisons pas le Particle System de Unity parce que les particules générées par ce composant ne sont pas facilement manipulables par le code. Nous utiliserons des GameObjects et agirons sur leur Rigidbody.

- Créer le modèle de la particule que le générateur va instancier.
    - Utiliser un Quad pour la représenter graphiquement,
    - Supprimer son composant **MeshCollider**.
    - Ajouter un composant **BoxCollider2D** et un composant **Rigidbody2D**.
    - Utiliser un composant **TrailRenderer** pour dessiner sa traîne.

Le générateur de particules

- Créer un GameObject vide qui sera le générateur de particules. Lui assigner une icône pour pouvoir le placer facilement dans le niveau par la suite.

- Ajouter un nouveau script au générateur.
    - Le générateur doit instancier des particules en continu avec :
        - une cadence paramétrable,
        - une position aléatoire dans un cercle de rayon paramétrable
            - Utiliser la fonction **Random.insideUnitCircle**.

Les particules doivent avancer dès leur création dans une certaine direction avec une certaine vitesse de départ, et s'arrêter au bout d'un moment. Quelle technique utiliser pour la faire avancer ? Faire les modifications nécessaires.

- Faire en sorte de pouvoir paramétrer :
    - la direction du mouvement de la particule,
    - la vitesse de départ de la particule,
    - le ralentissement de la particule.

Destruction des particules

- Détruire les particules quand elles ne bougent plus :
    - Soit en utilisant la propriété Auto Destruct du composant TrailRenderer,
    - Soit en calculant la vitesse de la particule grâce à son Rigidbody2D (grâce à la propriété velocity), et en détruisant la particule quand cette vitesse est inférieure à un certain seuil.

Créer les boîtes musicales

Le modèle de boîte musicale

- Créer le prefab des boîtes musicales.
    - Lui ajouter un composant **AudioSource**.
    - Dans ce composant activer la lecture en boucle (**loop**).
    - Réfléchir à comment représenter graphiquement les barres de volumes, et la hiérarchie du GameObject.

- Créer dans la scène quelques instances de boîte musicales, et ajouter un des **AudioClip** fourni dans l'AudioSource.

Lire les pistes musicales

Nous allons synchroniser manuellement le lancement des pistes musicales.

- Dans le composant AudioSource, décocher le paramètre Play On Awake.

Il faut créer un GameObject nommé "AudioManager" et lui ajouter un script qui :

- récupère toutes les AudioSources présentes dans la scène dans sa méthode Start(),
- À la suite dans Start(), créer une boucle sur ces AudioSources pour les faire démarrer à la même frame avec **AudioSource.Play()**.

Représenter graphiquement le volume

Ajouter un script au modèle de boîte musicale qui va synchroniser le volume du composant AudioSource avec la représentation graphique des barres de volume.
Pour ce faire, il faut :

- Créer un Material "allumé" et un Material "éteint",
- Récupérer le composant **Renderer** de chaque barre de volume et le mettre dans un tableau,
- Déterminer combien de barres sont allumées et combien de barres sont éteintes en fonction du volume de l'AudioSource (**AudioSource.volume**),
- Pour chaque Renderer du tableau, changer son Material (**Renderer.material**) en fonction de ce nombre.

Faire interagir les particules avec les boîtes musicales

- Une particule qui entre dans la zone de détection de la boîte musicale augmente le volume de cette dernière d'un cran.
- Au bout d'un certain délai (paramétrable) sans particule entrante, le volume de la boîte musicale baisse de lui-même avec une certaine vitesse (elle aussi paramétrable).

Les effectors

En passant dans les **AreaEffector2D** les particules seront déviées dans une direction avec une certaine force.

Le modèle de l'effector

- Créer le modèle de l'effector. Lui ajouter un composant AreaEffector2D et un **CircleCollider2D**.

L'effector est représenté graphiquement par un cercle central indiquant la direction de la force appliquée, et par un cercle extérieur indiquant le rayon d'action et la magnitude de la force.

- Réfléchir à l'organisation hiérarchique du GameObject.
- Utiliser le composant fourni "CircleShape" pour dessiner les deux cercles.
- Ajouter le sprite fourni "icon_arrow" au cercle central.

Tester l'effector

- Ajouter une instance de l'effector dans la scène.
- Tester le comportement de l'effector en le plaçant sur le chemin des particules. Les particules doivent être déviées dans la direction de l'effector avec une certaine force.
- Tester les différents paramètres du composant **AreaEffector2D**.

L'interaction avec les effectors

Il faut dans un premier temps déterminer la position de la souris à l'écran. Il y a plusieurs méthodes pour cela.

La plus générale et celle qui offre le plus de souplesse est d'utiliser des **Raycasts** tirés depuis la caméra, à l'endroit où est le pointeur de la souris. Étudier l'exemple de la documentation **Raycasts**, ainsi que les fonctions **Physics2D.Raycast** et **Physics2D.GetRayIntersection**.

Changer le curseur

- Préparer les images des curseurs ("icon_move" et "icon_resize") dans les **imports settings** de chaque fichier image :
    - Mettre "Texture type" à "Cursor",
    - Mettre "Max Size" à 32.

- Utiliser **Physics2D.GetRayIntersection** pour détecter quand la souris est au-dessus d'un effector.
    - Utiliser la méthode **Cursor.SetCursor** pour changer le pointeur de la souris pour l'icône "icon_move" quand il est au-dessus du cercle ou de l'icône centrale.
        - Pour détecter si le pointeur est au-dessus du centre, on peut calculer la distance entre la position du pointeur et le centre de l'effector, et tester si cette distance est inférieure au rayon du cercle central.
        - Attention, le hot spot est au centre de l'image du curseur.
    - Sinon, changer le pointeur de la souris pour l'icône "icon_resize" quand il est au-dessus du cercle extérieur.

Déplacement des effectors

Le déplacement d'un effector se fait par drag'n drop depuis son cercle ou flèche centrale.

- Détecter quand le joueur clique avec le bouton gauche de la souris (**Input.GetMouseButtonDown**), et tester si un effector est sous le pointeur de la souris à ce moment-là.
    - Si oui, mettre une référence vers cet effector dans une variable globale. Cela sera l'effector actif.

- Détecter quand le joueur relâche le bouton de la souris (**Input.GetMouseButtonUp**), et dans ce cas, remettre l'effector actif à **null**.

- Détecter si le joueur a cliqué sur l'icône ou le cercle central.
    - Si tel est le cas, on sait qu'on doit déplacer l'effector actif.

- Pour déplacer l'effector actif, simplement mettre la position de son transform à la position de la souris.
    - Attention à la coordonnée **z** de la position de la souris, penser à la mettre au même **z** que celui de l'effector.

Redimensionnement des effectors

Le redimensionnement d'un effector se fait par drag'n drop depuis son cercle extérieur. Utiliser la même méthode que pour le déplacement.

- Si le joueur a cliqué sur l'effector, mais pas dans son cercle central, alors on sait qu'on doit redimensionner l'effector.

- Pour redimensionner l'effector actif, il faut modifier le rayon du cercle extérieur (utiliser la propriété Radius du composant CircleShape).
    - Il faut également penser convertir le nouveau rayon en une magnitude plus ou moins grande pour l'AreaEffector2D attaché à l'effector.

Condition de victoire

Le niveau est gagné quand toutes les boîtes musicales sont remplies à 100% pendant un certain temps.

- Modifier le script AudioManager pour qu'il teste régulièrement si les volumes respectifs des boîtes musicales restent à 100% pendant une durée paramétrable.

Pour aller plus loin...

- Raffiner le déplacement des effectors :
    - Empêcher d'aller en dehors des bords de l'écran
    - Prendre en compte le delta entre la position du clic de la souris et le centre de l'effector, pour éviter que l'effector "saute" directement à la position de la souris
- Raffiner le redimensionnement des effectors :
    - Ajouter un rayon maximum et minimum en fonction de la taille du cercle intérieur
    - Prendre en compte le delta de position comme pour le déplacement
- Plusieurs niveaux qui s'enchaînent
- Afficher un chronomètre avec la durée de résolution à la fin d'un niveau
- Ecran de titre, de fin
- Builder le jeu
- Ajouter du **Post Processing** pour faire briller les particules