MASTER 1 DEDI
2020-2021

DOSSIER EXPLICATIF

DE "LOST SOULS"

FRANCINE ALTMEIER & QUENTIN PESTRE

I) DESCRIPTION DU CONCEPT

- **L'HISTOIRE**

Découvrez « Lost Souls » le nouveau jeux 2D réalisé en pixel Art. Vous allez rentrer dans le corps d'un fantôme d'une petite fille perdue, qui est à la recherche de son corps physique. Cependant, attention, les monstres cruels de Satan vont essayer de vous empêcher d'atteindre votre objectif. Vous aurez la possibilité de les éliminer et ainsi leur donner une bonne leçon, en leur sautant sur la tête.

Mais heureusement, vous n'êtes pas seul dans votre mission. Les anges célestes vont essayer de vous aider. Ces esprits angéliques, ne peuvent pas se montrer dans cet univers si obscur, cruel et dangereux. Cependant, ils seront présents, et vous enverront sur votre chemin des cœurs divins, qui vous permettront de vous soigner. De plus, ils vous protégeront quelques secondes lorsque vous avez reçu des dégâts. Ils ont aussi mis au point, une base de réapparition, à mi-parcours qui vous permettra de ne pas avoir à recommencer tout votre chemin quand vous l'aurez atteint et si vous tombez dans les bas-fonds infernaux. Ils ne peuvent cependant pas faire de miracle et il faudra seulement compter sur vous-même pour atteindre votre objectif, et ainsi revenir à la vie.

Afin de récupérer votre corps physique il va falloir récupérer les âmes bleues, au dernier luminaire, les anges célestes vont recevoir ces âmes et vous pourrez ainsi grâce à leur aide retrouver votre corps.

- **LES PERSONNAGES**

Le personnage principal :
un fantôme de petite fille qui souhaite retrouver son corps physique.

Les ennemis : moustiques, chauves-souris, larves, squelettes et mante-religieuses.

LE BUT DU JEU

Vous devez amener le personnage à la fin du parcours un dernier luminaire sera présent, une fois franchit cela indique la fin du jeu. Des ennemis vous infligeront des dégâts si vous ne faites pas attention le long de votre parcours. Si vous n'avez plus de vie vous perdez. Le parcours dispose de difficultés notamment avec des plateformes et des zones de vide. Si vous tombez vous devez recommencer votre cheminement.

LES POINTS FORTS DE NOTRE PROJET

- Système de mouvement (droite, gauche)
- Système de saut
- Système de « checkpoint »
- Système de jauge de vie
- Système de perte de points de vie
- Système de récupération de vie
- Système de « Respawn » lorsque le personnage tombe
- Système de fondu lors de la réapparition du personnage
- Système de collectes d'éléments (flammes bleus)
- Un système d'invincibilité et de clignotement du personnage lorsqu'il subit des dégâts
- 4 menus différents : menu principal, menu pause, menu game over, menu de fin de jeu.
- Un univers graphique élaboré basé sur du pixel Art
- Une ligne d'arrivée avec un luminaire en fin de parcours
- Des ennemis avec différents aspects graphique

II) NOTICE D'UTILISATION

NOTAMMENT LES TOUCHES CLAVIERS DU CLAVIER À UTILISER DANS LE CADRE DU BUILD WINDOWS

Le personnage dirigé par le joueur de « Lost Souls » à la possibilité de se déplacer seulement de façon linéaire. Ce déplacement restrictif est dû au fait que notre jeu soit réalisé en 2D. Les déplacements se font à l'aide du bouton rond entouré de gris et d'un cercle blanc, en bas à gauche de l'écran d'affichage du jeu. Ce visuel représente l'élément appelé « joystick » dans les « game play ». Ce joystick permet un déplacement de gauche à droite et vers la hauteur, du personnage.

- **Pour se déplacer vers la gauche,** le rond interne du joystick de couleur blanche doit être tiré glissé vers la zone grise. Il est aussi possible de le déplacer au-delà du cercle blanc en direction de la gauche pour que le personnage se déplace avec une vitesse maximale. Plus le rond interne est tiré plus le joueur à une vitesse importante.

- **Pour se déplacer vers la droite,** le principe est le même. Il est nécessaire de déplacer le rond blanc du joystick vers la droite, plus il est étiré plus le personnage à une vitesse importante.

- **Le déplacement vers le haut,** correspond à faire sauter le personnage dans le jeu. Il est nécessaire pour réaliser cette action de diriger le joystick vers le haut. Une nuance par rapport aux autres types de déplacement est cependant à noter. En effet, pour un meilleur confort de jeu, il n'a pas une proportion de saut. Que vous tiriez au maximum le joystick vers le haut ou que vous le montiez d'une façon minime, le personnage sautera à la même hauteur.

- En pressant le bouton de la lettre « h » sur le clavier, cela va volontairement procurer des dégâts sur le personnage. Nous avons beaucoup utilisé cette fonctionnalité lors de la création du jeu, pour faire des tests et sera peut-être enlevée lors de l'export final.

- La barre espace permet de faire sauter le personnage mais est beaucoup moins pratique que l'utilisation du joystick.

En ce qui concerne les déplacements des ennemies il se dirige exclusivement de gauche à droite entre deux bornes définies. Ces bornes sont visibles dans la « scène » de création du jeu grâce à l'utilisation d'icon par ce qu'en effet ce que nous avons appelé « Waypoint » sont en réalité des « GameObject Empty » et ne sont à l'origine pas visible. L'avantage des icons est qu'ils ne sont pas visibles par le joueur, mais permettent une meilleure compréhension visuelle lors de la création du projet.

D'AUTRES FONCTIONNALITÉS SONT PRÉSENTES DANS LE JEU :

En pressant **le bouton « Echap »** en haut à gauche sur votre clavier, cependant cette action n'est pas possible pour la version mobile. L'action de **mise en pause du jeu** vient d'être effectuée, le joueur comprend de suite l'action qu'il vient de réaliser parce que la mention textuelle « Jeu en pause » s'affiche immédiatement au milieu de l'écran. De plus, en dessous de cette inscription nous voyons un menu apparaître. Le menu dispose de trois boutons :

- Un bouton permettant de **« reprendre la partie »** en cours de jeu, il est aussi possible d'accéder aux paramètres.
- Un bouton **« paramètres »** ouvre une fenêtre de type Pop-Up. Cette fenêtre donne la possibilité :

◦ De quitter le mode « plein écran »,
◦ De régler le volume sonore indépendamment des sons et des musiques. En mettant ce curseur au minimum le volume est inaudible.

➤ Pour quitter le mode paramètre une croix de fermeture de la fenêtre pop-up est présent en haut à droite.

- Un bouton permettant d'accéder au **« menu principal »**. Le joueur se retrouve alors sur le menu que l'on retrouve en ouvrant le jeu.

Le menu principal que l'on retrouve en lançant le jeu ou encore en utilisant les autres menu, est quant à lui annoté du nom du jeu, en grand, au milieu de l'écran. Nous pouvons apercevoir l'inscription « Lost Souls ». Et de nouveau trois boutons apparaissent :

- Le bouton **« commencer le jeu »,** qui permet de lancer une partie.
- Le bouton **« paramètres »** est de nouveau présent ici avec les mêmes fonctionnalités qu'expliquer ci-dessus
- Un bouton **« quitter le jeu »,** permettant d'interrompre les programmes en cours d'exécution.

Un menu spécifique s'ouvre à la mort du personnage. Ce menu est lui aussi composé de trois boutons :

- Un bouton **« retenter »** qui permet de rejouer une partie
- Un bouton redirigeant vers le **« menu principal »**
- Un bouton permettant de **« quitter »** le jeu

Et enfin nous retrouvons le dernier menu qui s'affiche lorsque le joueur franchit la ligne d'arrivée. En franchissant cette ligne, un son pour féliciter le joueur retentit ainsi que ce menu qui s'affiche indiquant "Bravo !! Tu as réussi à retrouver ton corps" qui montre au joueur qu'il a réussi à apporter les âmes récupérées sur son chemin afin de les donner aux anges célestes, qui lui permettent ainsi de retrouver le corps physique de la petite fille que incarne le joueur. On y voit également le personnage en couleur pour montrer qu'il n'est plus un fantôme et qu'il est à nouveau vivant. Ce menu est composé de trois boutons :

- Un bouton « rejouer » qui permet de rejouer une partie si le joueur en a envie.
- Un bouton redirigeant vers le « menu principal »
- Un bouton permettant de « quitter » le jeu

III) PRODUCTION DU GROUPE

INDICATIONS SUR CE QUI A ÉTÉ PRODUIT PAR LE GROUPE ET CE QUI A ÉTÉ DIRECTEMENT INTÉGRÉ À PARTIR « D'ASSETS »

Les designs, musique, éléments graphiques ont été soigneusement sélectionné pour rentrer pleinement dans l'univers de « Lost Souls ». Nous les avons trouvées sur des sites mettant à disposition des images en pixel Art, libre de droit. De plus, nous avons sélectionné une typographie en pixel, sur le site « Sharp Retro », pour une immersion encore plus globale.

INSPIRATIONS

Nous nous sommes inspirés pour réaliser ce jeu de notre enfance. Avec « Super Mario » pixélisé joué sur la Séga. Mais sans oublier des jeux beaucoup plus récents de type « Hollow Knight » et « Gris ».

DESIGN D'INTERFACE

Le design d'interface a été réalisé à l'aide d'éléments graphique importés, dans une optique d'uniformisation et afin de créer un concept immersif fort. La façon de jouer a aussi été étudiée pour que le joueur comprenne facilement comment se déplacer et interagir avec le personnage. Nous avons aussi mis à disposition 3 menus différents pour faciliter l'utilisation du jeu.

DESIGN SONORE

Le design sonore est un élément très important, il a beaucoup été travaillé dans notre projet, afin de vous permettre une immersion globale. En effet, vous retrouverez une musique d'ambiance pour le menu principal du nom de « Where to now » et une musique d'ambiance pour la partie du jeu du nom de « Life Wave ». On y trouve également plusieurs sons d'interaction, comme quand le personnage récupère une âme, de la vie, ou encore quand il subit des dégâts et réussi à éliminer un ennemi. Lorsque le personnage traverse le dernier checkpoint qui est la ligne finale du jeu, on peut également entendre un son pour féliciter le joueur.

DESIGN D'EXPÉRIENCE

Au cœur de notre formation le design d'expérience passant par le design d'interface et le design sonore permet une immersion forte du joueur. Cette immersion est accentuée par l'histoire, la narration et l'univers graphique du jeu. Tous ces éléments permettent une utilisation facilement compréhensible du « Game Play », tout en procurent des émotions et des sensations.

PROGRAMMATION

Nous avons été assez loin dans la programmation pour le projet, cela est passé par plusieurs éléments de programmation en C# de type :

- Variable
- Fonctions / passages de paramètres
- Variables privées / publiques
- Fonctions standards simples
- Boucles
- Conditions
- Coroutines
- Fonctions standards plus complexes

SOURCES

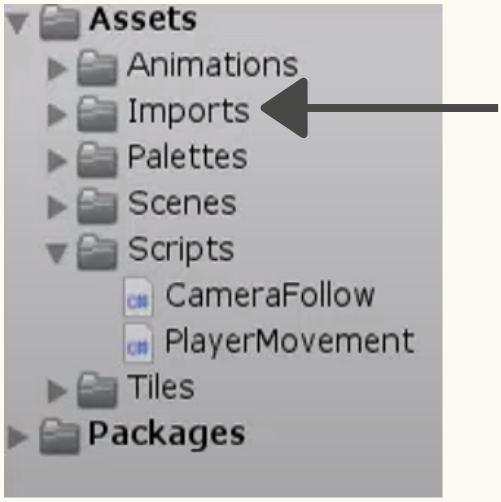
Nous avons apprécié réaliser ce projet avec l'aide des tutoriels de la chaîne YouTube « TUTO UNITY FR » mais aussi des tutoriels « Brackley », qui nous ont beaucoup aidé pour quelques petites subtilités.

Les seuls « assets » à proprement dit que nous avons intégrés ainsi que les seuls fichiers de codes d'autres sources que nous avons utilisées sont ceux pour le bon fonctionnement du joystick.

En espérant que vous prendrez autant de plaisir à y jouer que nous à le concevoir, bonne découverte dans notre nouvel univers.

LOST SOULS

FICHES PÉDAGOGIQUE :
COMMENT CRÉER UN ENNEMI 2D ?



1. IMPORTER LE VISUEL DE L'ENNEMI

Glisser votre image dans un dossier « Imports » préalablement créé dans votre projet afin de bien ranger votre travail, puis glisser le visuel sur la « scene ». Renommer ensuite votre visuel en « Graphics ».

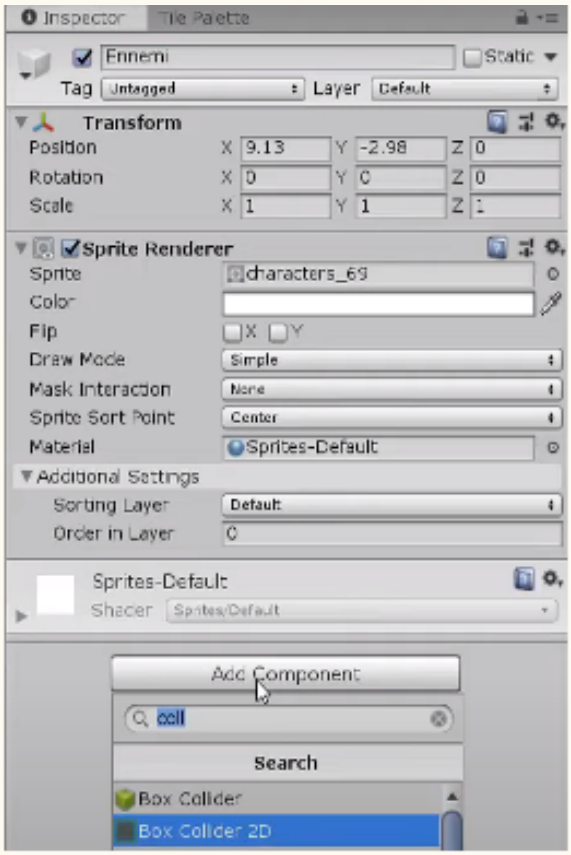
➤ Vous pouvez régler la taille de votre ennemi avec « scale », si besoin.

2. METTRE UN « BOX COLIDER »

Cliquer sur « Add Component », puis « box Colider 2D » par exemple.

Appuyer sur le logo suivant : 

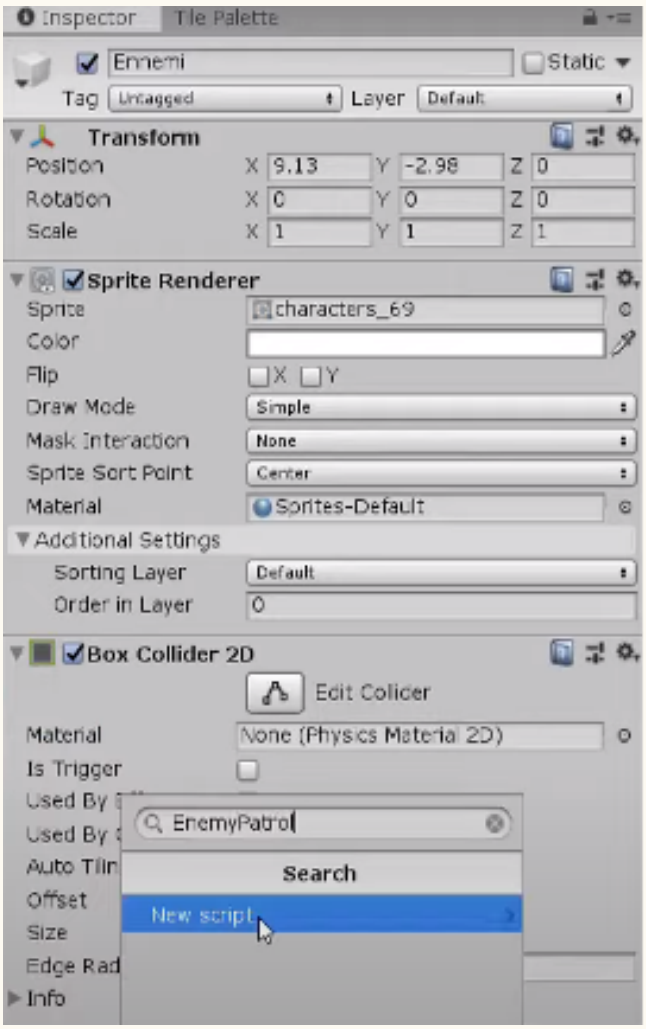
➤ Vous pouvez maintenant régler la taille du « box colider ». Il vous suffira de bien entourer l'ennemi. Vous venez de créer la zone qui pourra être alloué ultérieurement aux dégâts.



2. MAINTENANT NOUS DEVONS CRÉER UN SCRIPT POUR FAIRE DÉPLACER L'ENNEMI.

Pour cela cliquer sur le bouton « Add Compenent », nous allons créer un script que nous avons décidé d'appeler « EnemyPatrol ».

Puis ouvrez le script préalablement rangé dans le dossier « scripts » du projet.



4. PROGRAMMATION DU SCRIPT

Le système de déplacement se fera avec un système de « Waypoints ». Deux points entre lesquels l'ennemi va se déplacer.

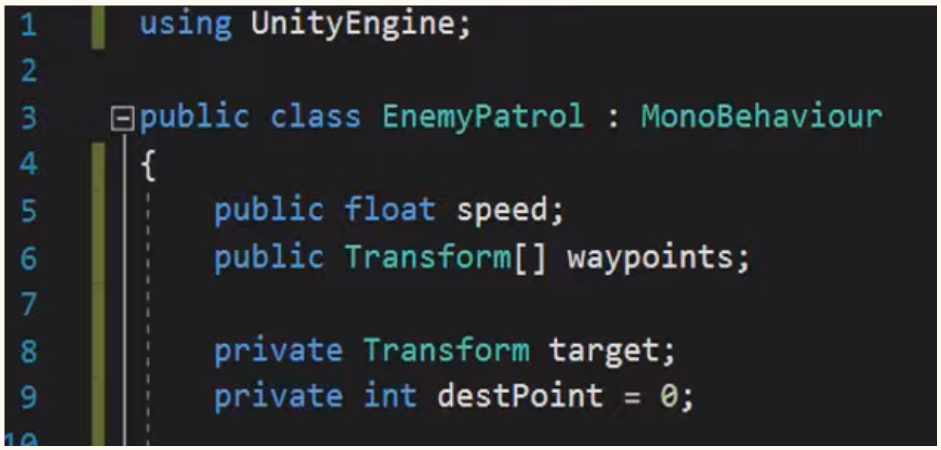
Nous allons faire une variable « float speed » qui définira la vitesse de déplacement de l'ennemi.

Nous allons créer une variable « transform waypoints » qui permettra de définir les « waypoints » vers lesquelles l'ennemi va se déplacer.

Une variable privée de type « transform target » qui définit la cible ou l'ennemi va se déplacer

Une variable privée de type « transform despoint » qui définit le point de destination nous le renseignons par défaut égale 0.

Voici le code :



```
using UnityEngine;

public class EnemyPatrol : MonoBehaviour
{
    public float speed;
    public Transform[] waypoints;

    private Transform target;
    private int destPoint = 0;
```

Le déplacement de l'ennemi se fait dans la fonction « Update ».

Pour cela on a mis un « vector 3 avec « dir » qui définit la direction suivante « target.position - transform.position ; »

Pour le déplacement nous allons utiliser « transforme.Translate » puis nous donnons la direction du translate dans les parenthèses : « transforme.Translate(dir.nomalized) » que l'on va multiplier par la vitesse « (* speed) » puis multiplier par « (Time.deltaTime) » pour faire un déplacement au fil du temps.

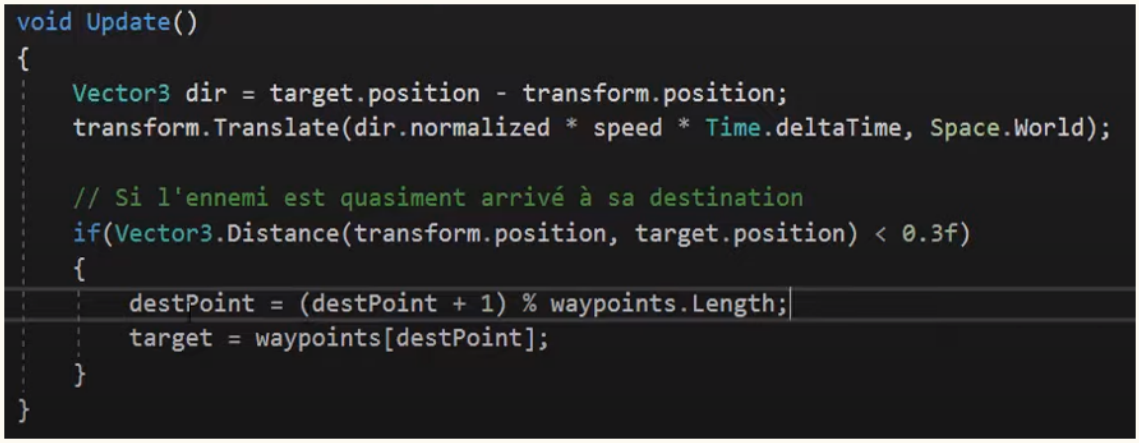
Pour finir, nous précisons « (Space.World) » pour que le personnage se déplace par rapport au monde.

Nous avons ensuite ajouté une condition pour que l'ennemi aille d'un point vers un autre et ne fasse pas juste un aller simple.

Pour rajouter une condition nous utilisons « if ». Si notre ennemi et le « Waypoint » vers lequel il se dirige à une distance inférieure à 0,3, il va alors se diriger vers le « waypoint » suivant.

On calcule la distance entre transform.position de notre ennemi et target.position pour avoir la distance entre les deux « Waypoints ». Du coup, si la distance entre l'ennemi et le Waypoint est inférieure à 0,3 alors l'ennemi fait demi-tour.

Voici le code :



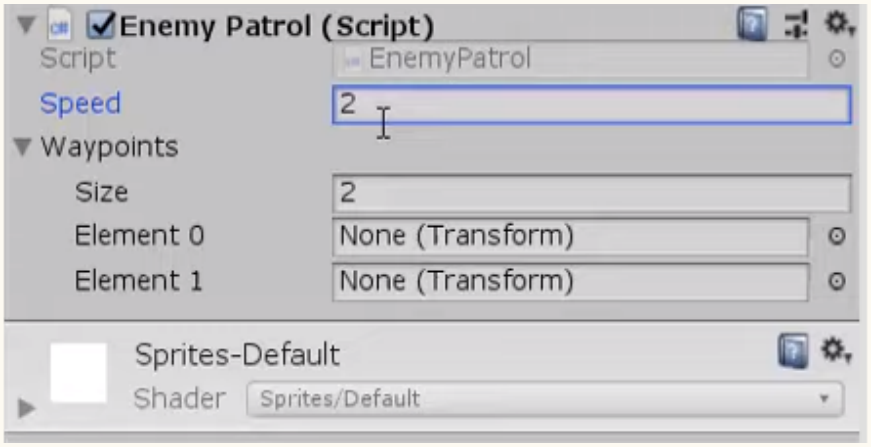
```
void Update()
{
    Vector3 dir = target.position - transform.position;
    transform.Translate(dir.normalized * speed * Time.deltaTime, Space.World);

    // Si l'ennemi est quasiment arrivé à sa destination
    if(Vector3.Distance(transform.position, target.position) < 0.3f)
    {
        destPoint = (destPoint + 1) % waypoints.Length;
        target = waypoints[destPoint];
    }
}
```

Dans la fonction « Start » on va initialiser « target » pour qu'il soit égal à « waypoints » et de valeur par défaut 0.

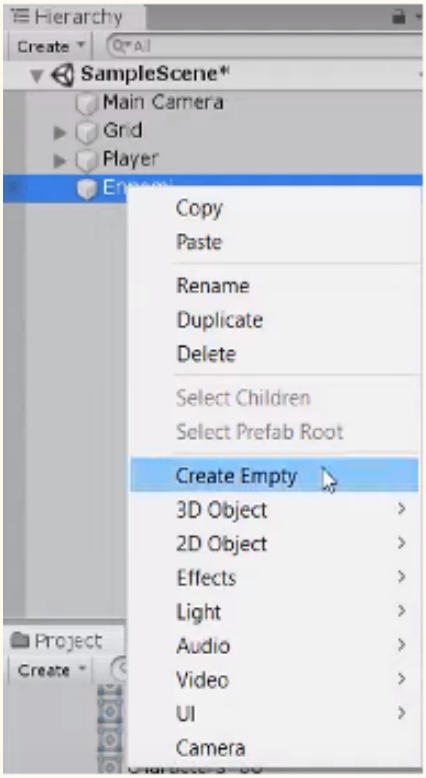
```
// Start is called before the first frame update
void Start()
{
    target = waypoints[0];
}
```

5. DÉFINIR DANS « L'INSPECTOR » LE NOMBRE DE « WAYPOINTS »

Dans « Size » et la vitesse de l'ennemi dans « speed ». Ici nous avons renseigné « 2 » dans les deux champs.

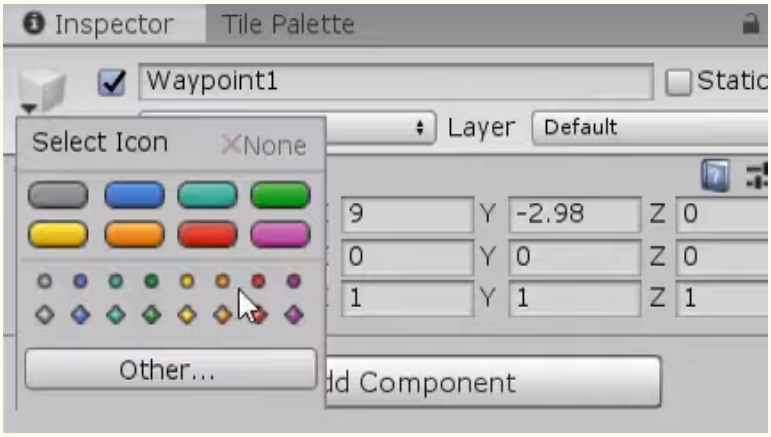


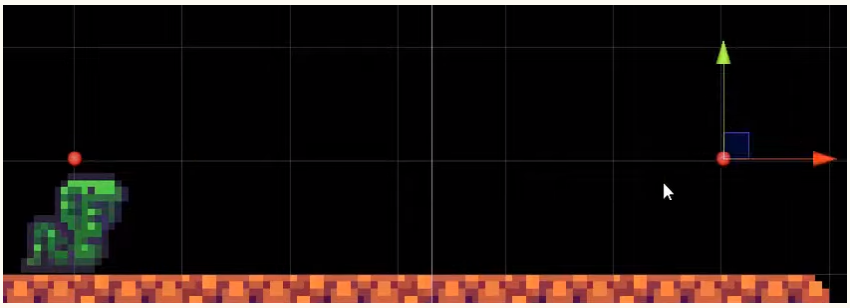
6. CRÉER LES « WAYPOINTS »



Pour cela commencer par créer un « Empty ».

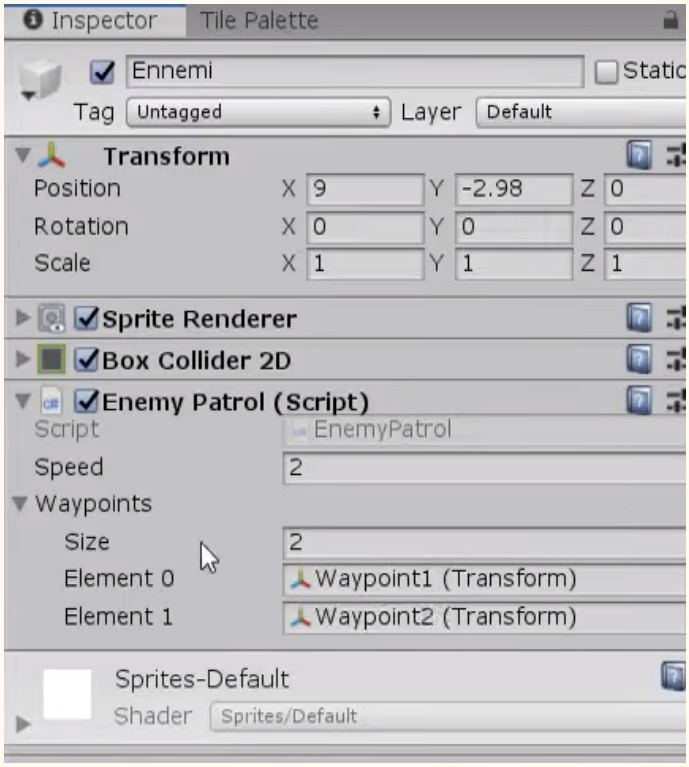
Pour qu'il reste visible on va lui rajouter un Icon, pour cela cliqué dans « l'inspector » sur le cube en haut à gauche, puis sélectionné une icon.





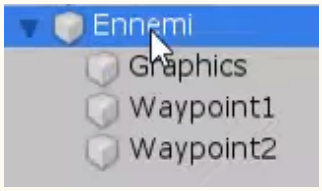
Maintenant vous pouvez dupliquer le « Waypoint » pour qu'il y en ai deux. Renommez les par exemple en « Waypoint1 » et « Waypoint2 ».

Maintenant renseigner « Waypoint1 » dans « élément 0 » de la liste du script « Enemy Patrol » et « Waypoint2 » dans « élément 1 » comme ci-joint.



7. HIÉRARCHISER LES FICHIERS

Maintenant renseigner « Waypoint1 » dans « élément 0 » de la liste du script « Enemy Patrol » et « Waypoint2 » dans « élément 1 » comme ci-joint.

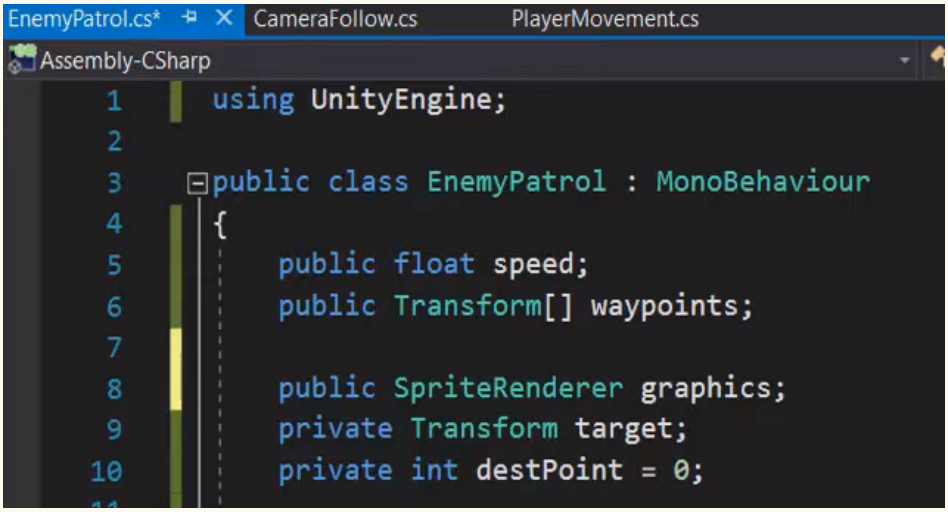


8. FACULTATIF : VOUS POUVEZ ANIMER VOTRE ENNEMI

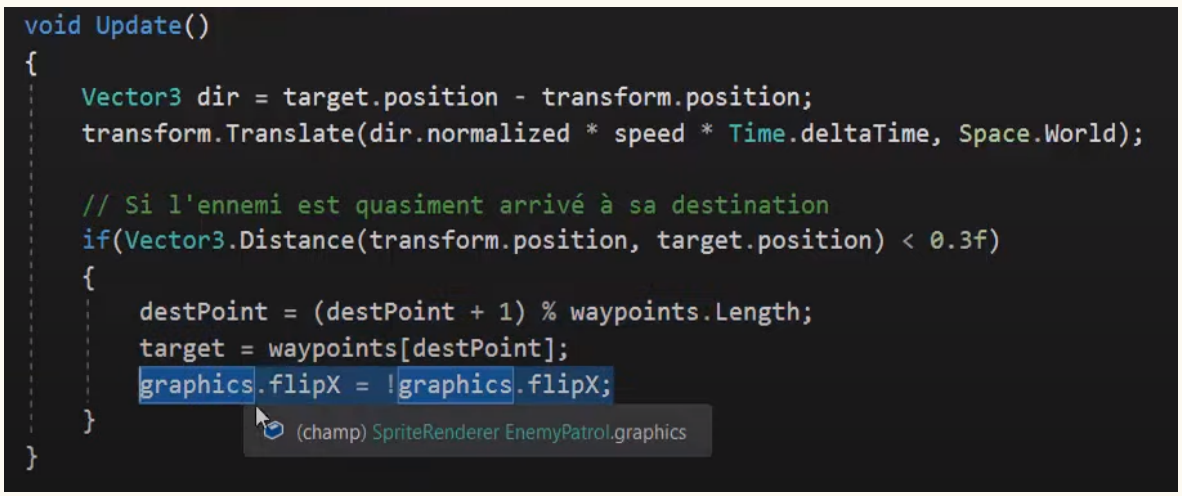
Pour cela, utilisé la fenêtre d'animation dans « Windows », « Animation », « Animation », cela permet de faire marcher votre ennemi.

9. FAIRE TOURNER L'ENNEMI QUAND IL FAIT DEMI-TOUR

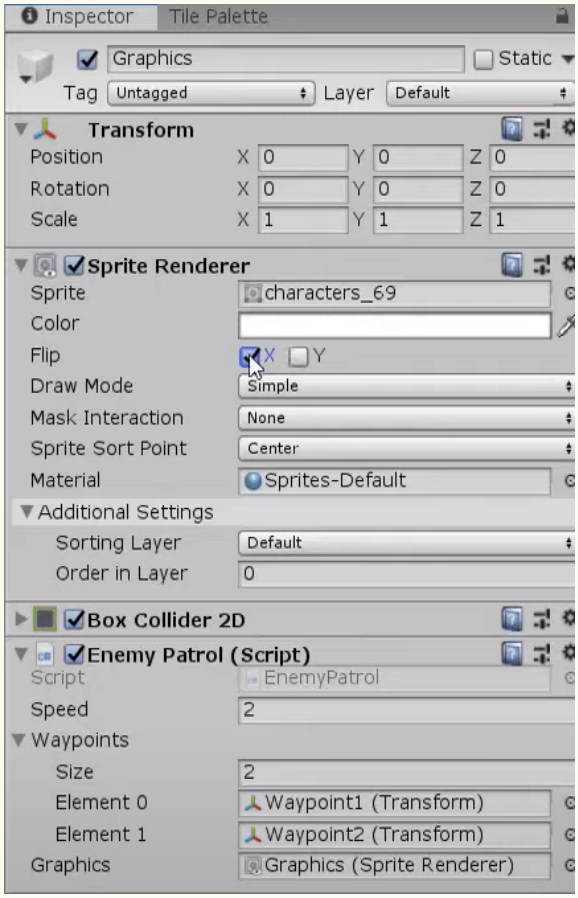
Rajouter la variable : « public SpritRenderer graphics ; » dans le script « EnemyPatrol »



Puis « graphics.flipX =!graphics.flipX ; » dans la fonction Update

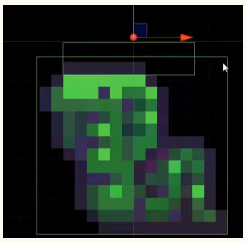


Dans « l'inspector » un champ de liste « Graphics » est apparut dans le fichier « EnemyPatrol », glisser le « Sprite Renderer » à l'intérieur. Puis coché la case « Flip », « X ».



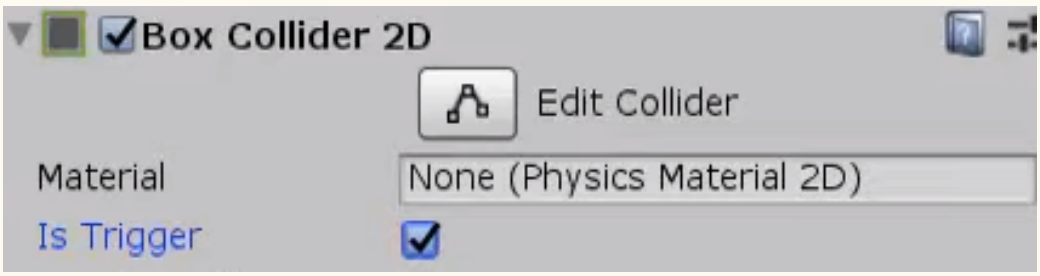
10. ÉLIMINER L'ENNEMI EN LUI SAUTANT SUR LA TÊTE

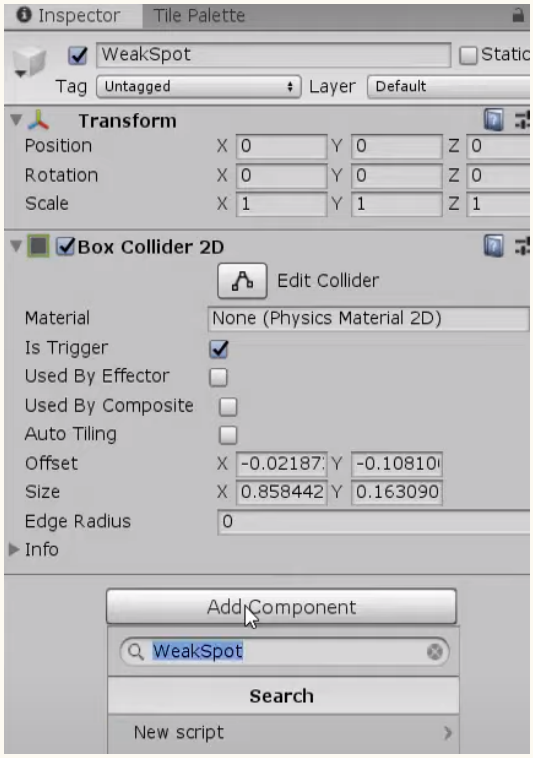
Pour cela créer un « Empty », nous l'avons nommé « WeakSpot ».



Puis créer un « box colider 2D» sur ce « Game Object » « WeakSpot ». Placer ce « box colider 2D» sur la tête de l'ennemi de façon que quand le joueur saute dessus il soit en contact avec. comme ci-joint

Cocher maintenant la case « Is Trigger » dans le « box colider » pour que le personnage soit détecté quand il entre dans cette zone et qu'il puisse la traverser.





Créer maintenant un nouveau script sur le « Game Object » « WeakSpot », en appuyant sur « add component », nous avons appelé le script « WeakSpot ».

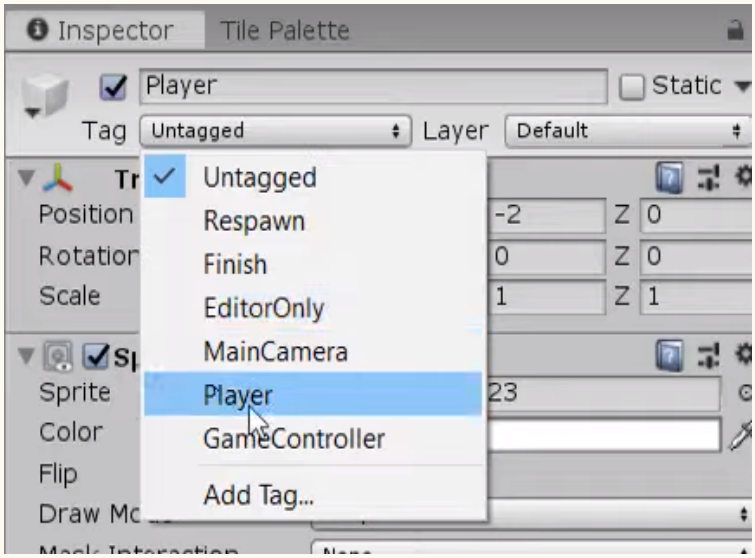
Nous allons maintenant créer une « void OnEnterTrigger2D » avec entre parenthèses collision « (colision2D collision) ». Ce code permet de lancer la variable suivante, à chaque fois qu'il y a une collision, c'est à dire qu'un élément va entrer en contact avec cette zone.

Nous voulons détruire l'ennemi, si le joueur et seulement le joueur entre dans le « WeakSpot ». Nous devons donc définir que la collision soit prise en compte seulement lorsque c'est le personnage qui entre dans cette zone. Pour cela on va faire une vérification avec une condition « if ».

Voici le code :

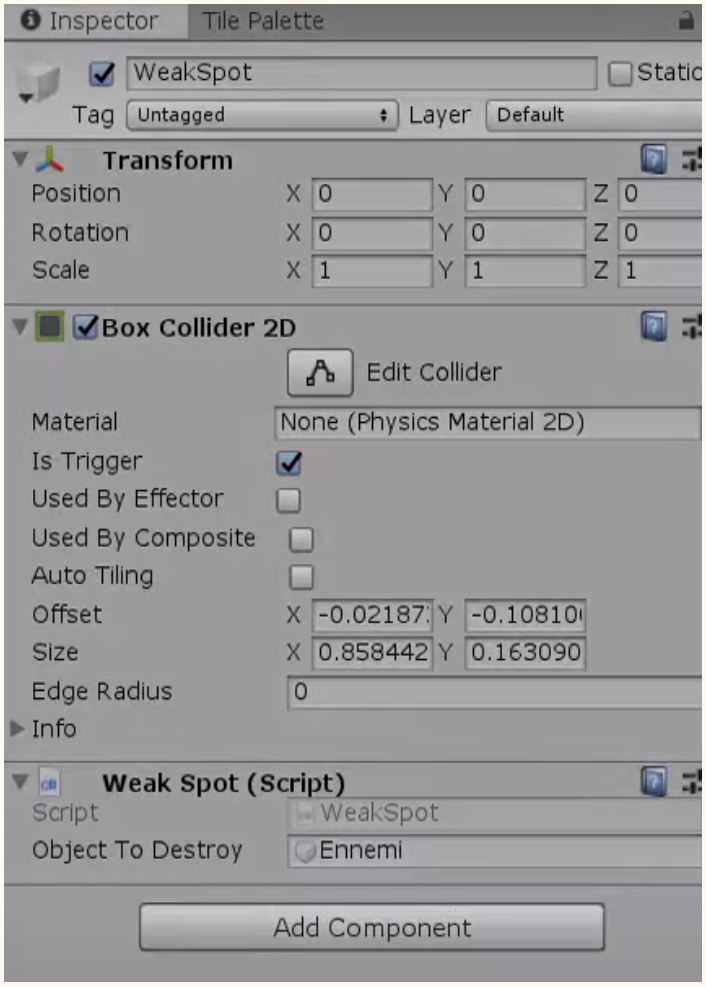
```
using UnityEngine;

public class WeakSpot : MonoBehaviour
{
    public GameObject objectToDestroy;

    private void OnTriggerEnter2D(Collider2D collision)
    {
        if (collision.CompareTag("Player"))
        {
            Destroy(objectToDestroy);
        }
    }
}
```

Avec cette fonction nous demandons de vérifier si l'objet qui entre en collision, a le tag « Player ». Pour cela il est nécessaire de définir sur le joueur dans la partie « Inspector », puis « Tag » l'attribut « Player », comme ci-dessous.



Puis pour que cela fonctionne il suffit de mettre le « Game Object » « WeakSpot » en enfant du « Game Object » « Graphics », comme ci-dessous.

Pour finir, il faut aussi glisser le « Game Object » « Ennemi » dans la liste de « l'inspector » du « WeakSpot », à l'intérieur de « Object To Destry », comme ci dessous.



LOST SOULS

BRAVO ! VOUS AVEZ CRÉÉ UN ENNEMI ET VOUS POUVEZ L'ÉLIMINER.