M2 Ingénieurie du jeu Vidéo
Gamagora - ICOM 2024/2025

Module Bases des moteurs de jeux (Unreal Engine)

Travaux Pratiques 4

Jeu 2D Side Scroller

(illustrations réalisées sous UE5.4.4)

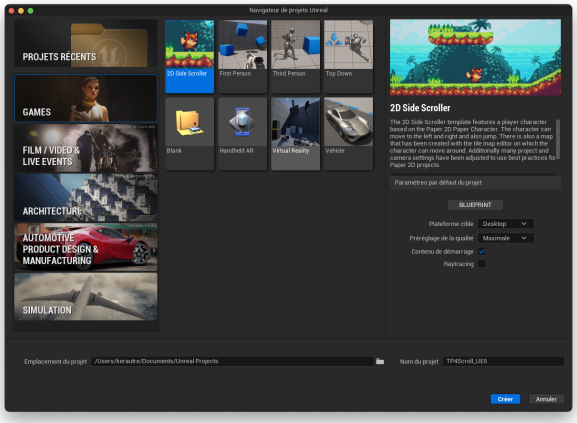
Objectif: Dans cette séance, l'idée est de reprendre les notions vues dans les TP précédents mais dans un contexte différent en utilisant un modèle de jeu de type *2D side scroller*.

Partie I: Installation du modèle de jeu

Pour réaliser ce TP, il est nécessaire dans un premier temps d'installer le modèle de jeu `TP_2DSideScrollerBP.zip` disponible sur moodle et aussi distribué sur *GitHub*[1]. L'installation s'effectue en décompressant l'archive récupérée dans le répertoire `templates` d'UE55, c'est à dire dans le chemin ressemblant au suivant :

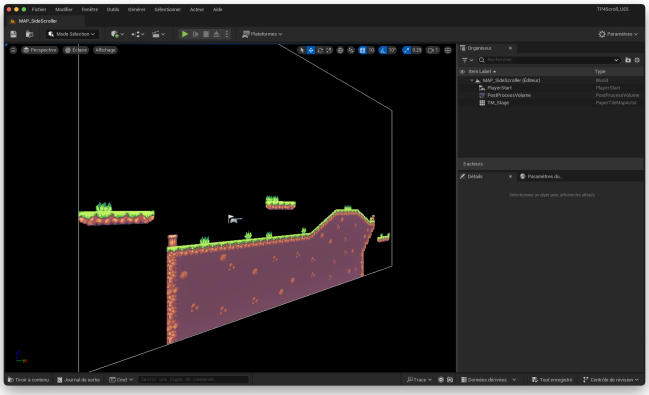
`C:\ProgramFiles\EpicGames\UE_[version]\Templates`

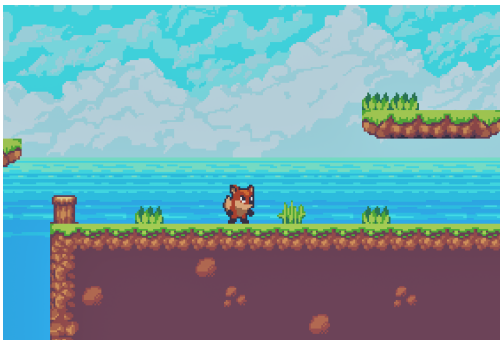
1.1 Ce répertoire contient les différents modèles de jeu et si votre installation a bien fonctionné, lorsque vous relancer *Unreal Engine* il devrait désormais vous proposer un nouveau type de jeu *2D Side Scroller* comme illustré ci-dessous.



1. `https://github.com/CobraCodeDev/TP_2DSideScrollerBP`

1.2 Si tout fonctionne bien, vous devriez avoir le jeu suivant lorsque vous lancez le jeu.

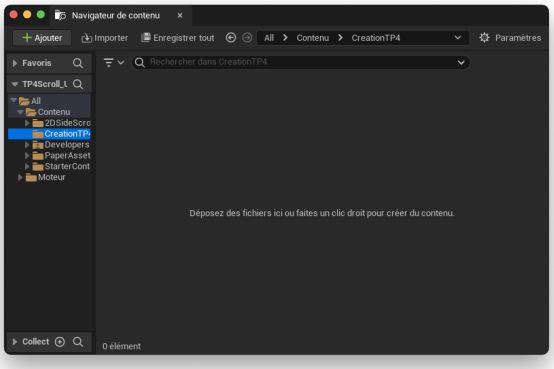




Partie II: Ajout d'élément dans le jeu

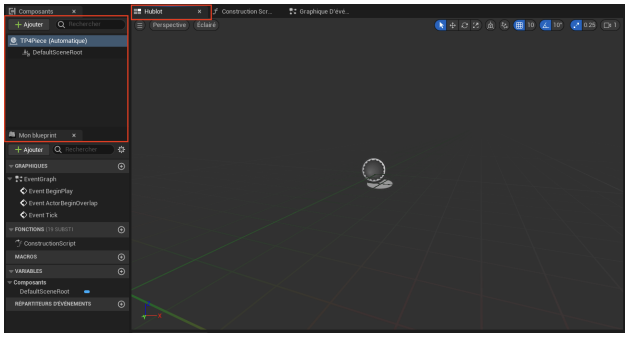
Dans cette partie, il s'agit de créer une classe *Blueprint* qui servira à représenter une pièce dans le jeu. L'avantage de créer une nouvelle classe et qu'elle va permettre d'être re utilisée partout dans le niveau.

2.1 Afin de retrouver rapidement vos réalisations, il est demandé de créer un nouveau répertoire nommé *CreationTP4* qui contiendra vos réalisations comme sur l'image suivante :



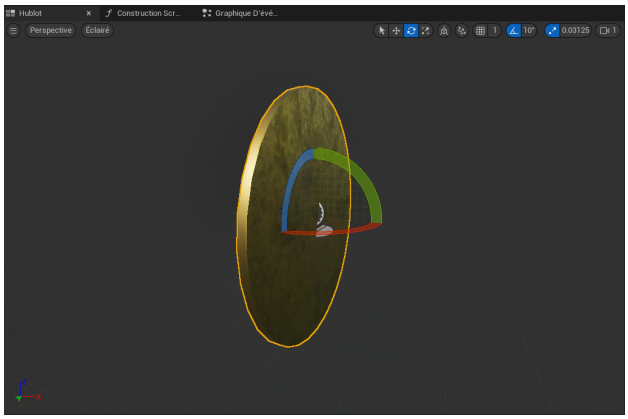
2.2 Rajoutez une nouvelle classe *Blueprint* (nommée `TP4Piece`) de type `Actor` dans le répertoire précédant qui représentera une pièce que l'on pourra ensuite placer plus tard dans le jeu à différents endroits. **Indication** : une nouvelle classe s'obtient à partir d'un clic droit depuis le navigateur de contenu.

Une fois la classe crée, il s'agira alors de commencer la conception par la modélisation 3D de sa représentation. Pour cela il s'agit de rajouter des éléments dans la zone du hublot :

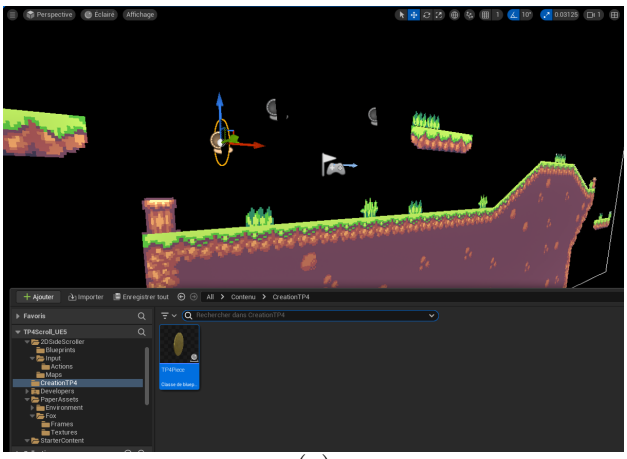


2.3 Dans la zone des composants, rajoutez un cylindre et ajustez le verticalement et ajoutez une rotation (voir éléments ci-dessous).

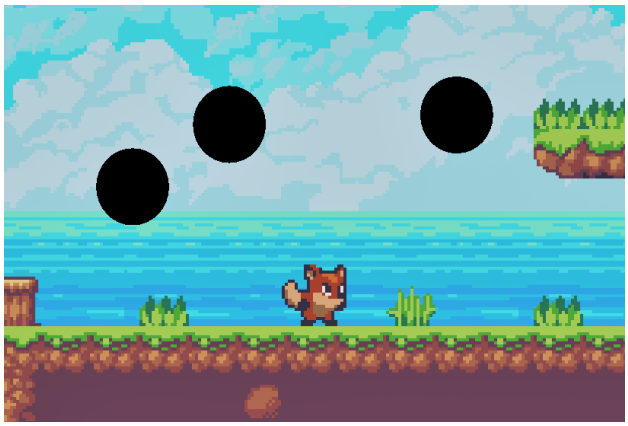
2.4 Toujours dans l'éditeur de la classe *Blueprint*TP4Piece, rajoutez un *StarterContent* afin de modéliser la pièce avec le matériel or comme sur l'image suivante :



2.5 Ajoutez quelques pièces dans la scène comme sur l'image suivante

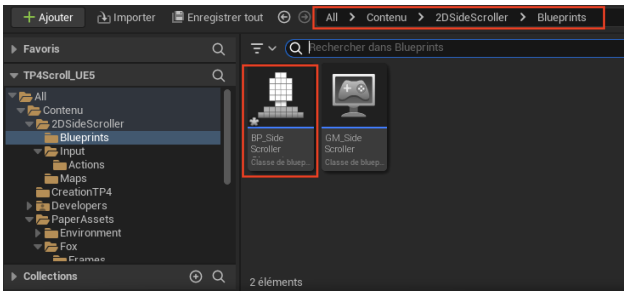


(a)



(b)

2.6 Comme vous pouvez le voir dans la simulation du jeu (ou l'image (b) ci-dessus), les pièces apparaissent sombres. Pour que cela ne soit pas le cas vous pouvez rajouter une lumière dans la classe *Blueprint* représentant votre joueur. Cette classe est située dans le répertoire représenté sur l'image (a) ci-dessous. Après avoir ajouté la lumière et ajusté les paramètres vous obtiendrez une nouvelle visualisation comme sur l'image (b) ci-dessous.



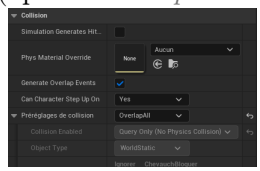
(a)

(b)

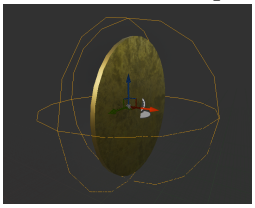
Partie III: Acquisition des pièces par le joueur

Maintenant il s'agit de gérer l'acquisition des pièces par le joueur. Pour cela trois étapes sont à suivre.

— Etape 1 : faire en sorte que le joueur puisse passer à travers la pièce.
→ dans le champ *collision*, il y a un preset permettant de faire en sorte que tous les éléments qui se superposeraient au cylindre (option *overlap all* illustré ci-dessous).



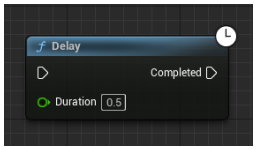
— Etape 2 : détecter quand le joueur entre en contact avec la pièce. Pour une plus grand jouabilité, rajoutez une `SphereCollision` sur votre pièce.



— Etape 3 : faire disparaitre la pièce en utilisant la fonction `destroy`.

3.1 Suivre les étapes précédentes pour faire en que le ramassage de pièce soit effectif.

3.2 Ajouter un bruit de pièce qui sera produit lorsque le joueur passe sur la pièce. Le fichier est disponible sur *moodle*, et il ensuite possible de l'intégrer dans votre projet à partir d'un simple glissé-déposé dans les assets, puis en glissant déposer le son dans l'*event graph*. **Attention :** comme la pièce est détruite juste après vous devrez rajouter un petit délais entre la lecture de la piste audio et la destruction de la pièce (voir ci dessous).



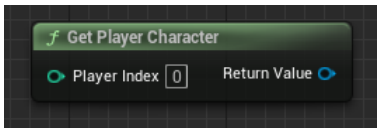
Partie IV: Gestion de base de la partie

Dans cette partie l'objectif sera de :
— Détecter lorsque le joueur tombe dans le vide.
— Replacer le joueur sur la position initiale.
— Etendre les dimensions du plateau et adapter les décors.

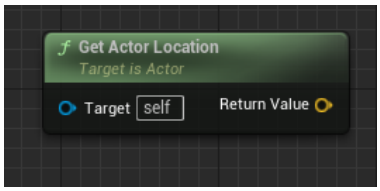
Pour détecter quand le joueur sors de la zone de jeu, une stratégie simple consiste à détection à chaque rafraichissement (`Event Tick`) les coordonnées du joueur et de vérifier si ce dernier n'est pas en dessous de la valeur seuil choisie manuellement. Il s'agit d'abord d'implémenter cette stratégie pour ensuite proposer une solution plus robuste en cas par exemple d'ajustement de la scène.

4.1 En déplaçant un élément de la scène, repérez la coordonnée en Z du décors qui permettra de détecter quand le joueur sors du cadre du jeu.

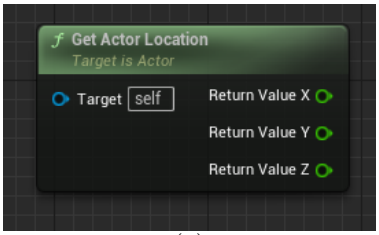
4.2 Sachant que l'objet controlé par le joueur peut être obtenu par la fonction `GetPlayerCharacter` (image (a) ci-dessous), dans le ***Level Blueprint***, rajoutez des instructions de façon à détecter quand la position du joueur passe sous un le suil de la question précédente. **Indication :** Il est possible de transformer un vecteur (valeur de retour illustré sur l'image (b)) en plusieurs composantes (image (c)) à partir d'un clic droit.



(a)

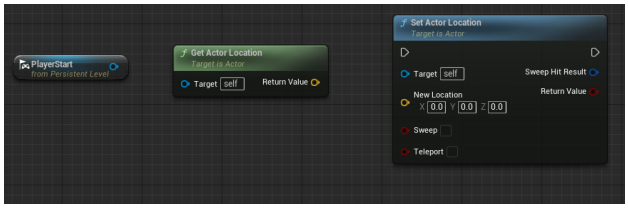


(b)



(c)

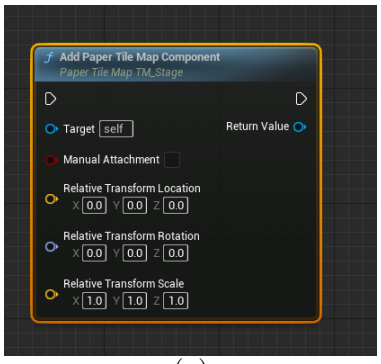
4.3 Une fois que vous avez testé le passage du joueur en dehors du jeux, il s'agit de replacer le joueur à sa position initiale. Pour cette étape, vous pourrez utiliser les blocs suivants :



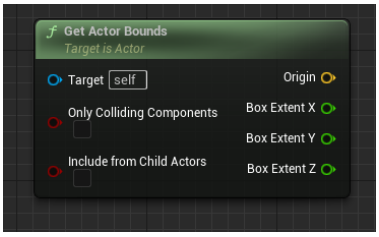
4.4 Enfin, pour rentre la détection robuste à des ajustements du design sans ajuster manuellement les seuils, il est possible d'utiliser un acteur vierge et de récupérer ses coordonnées. Rajoutez ces éléments dans le jeu et exploitez les coordonnées automatiquement dans le ***Level Blueprint***.

Partie V: Pour ceux qui vont vite...

Pour ceux qui vont vite, il s'agit de générer un plateau de jeu infini. La stratégie pourra suivre les étapes suivantes :
— Détecter quand le joueur arrive à un certain endroit prés de la fin du monde.
— Générer un nouveau décors (voir image ci-dessous).
— Ajuster les points de références pour générer à l'infinie des plateau automatiquement (utiliser les dimensions du plateau avec le bloc (b)).


Add Paper Tile Map Component
Paper Tile Map TM_Stage
Target self
Return Value
Manual Attachment
Relative Transform Location
X 0.0 Y 0.0 Z 0.0
Relative Transform Rotation
X 0.0 Y 0.0 Z 0.0
Relative Transform Scale
X 1.0 Y 1.0 Z 1.0

(a)


Get Actor Bounds
Target is Actor
Target self
Origin
Only Colliding Components
Box Extent X
Box Extent Y
Include from Child Actors
Box Extent Z

(b)