M2 Ingénieurie du jeu Vidéo
Gamagora - ICOM 2024/2025

Module Bases des moteurs de jeux (Unreal Engine)

Travaux Pratiques 5

Jeu 2D Side Scroller (suite)

(illustrations réalisées sous UE5.3.1)

Objectif: Dans la continuité de la séance précédente, il s'agit de poursuivre la réalisation du mini jeu de type *Side Scroller*. Un premier objectif sera de gérer le score à travers le nombre de pièces récupérées par le joueur puis de gérer le nombre de vies.

Partie I: Gestion du score du joueur

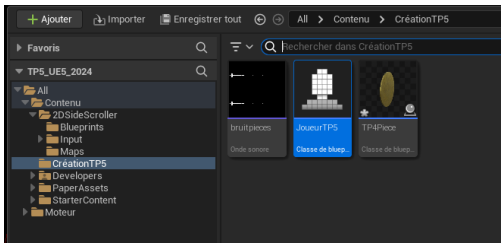
Dans cette partie, il s'agit de pouvoir récupérer et afficher le score du joueur simplement en mode texte à l'écran (en utilisant un `print()`. Dans la partie suivante nous verrons comment rajouter un widget dans l'interface. Le résultat de cette partie devra ressembler à l'affichage suivant :

Stratégie : pour la gestion du score nous allons utiliser la stratégie simplifiée suivante :

1. Le joueur possédera une variable lui permettant de retenir le nombre de pièce qu'il a ramassé.
2. Le joueur possédera aussi une fonction lui permettant d'augmenter son score lorsqu'il ramasse une pièce.
3. A partir de la détection de collision d'une pièce, il sera possible de récupérer le joueur et il suffira d'appeler sa fonction d'augmentation du score.

1.1 Comme pour la séance précédente, organisez vos créations en utilisant un nouveau répertoire *CreationTP5* qui contiendra aussi les créations de la séance précédente.

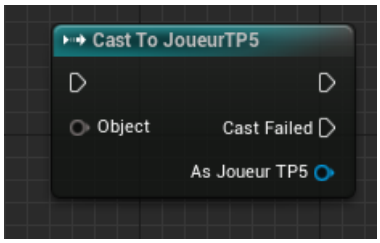
1.2 Recherchez la classe du joueur `BP_SideScrollerCharacter` et la déplacer dans le nouveau répertoire précédant en renommant la classe `JoueurTP5`. Vous devez obtenir obtenir une organisation ressemblant à l'image ci-dessous.



1.3 Afin de réaliser l'étape 1 de la stratégie précédente, rajoutez à votre joueur une variable `monScore` de type entier qui aura une valeur par défaut égale à 0.

1.4 Pour la deuxième étape, toujours dans la classe associée à votre joueur, rajoutez une fonction `gagnePiece` qui devra mettre à jour le score du joueur. Il s'agira d'incrémenter le score d'une unité.

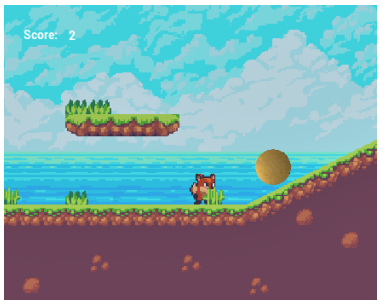
1.5 La dernière étape (3) va consister à appeler la fonction précédente à partir de la pièce.
Indication : l'event `begin overlap` permet de récupérer l'acteur qui génère l'événement d'*overlap* (*Other Actor*). Pour pouvoir appeler la méthode, il est nécessaire d'appliquer un cast pour effectuer des actions uniquement quand l'objet à l'origine de la superposition est bien du type du joueur (bloc illustré ci-dessous).



1.6 Afin de tester le bon fonctionnent de l'incrément du score, dans la fonction `gagnePiece` de la classe *Blueprint* `JoueurTP5` afficher le score courant du joueur. Vous devriez obtenir un résultat comparable à la première image illustrée page 1.

Partie II: Affichage du score basé widget

Dans cette partie, il s'agit d'améliorer l'affichage en utilisant cette fois un widget permettant un affichage persistant comme illustré ci-dessous.



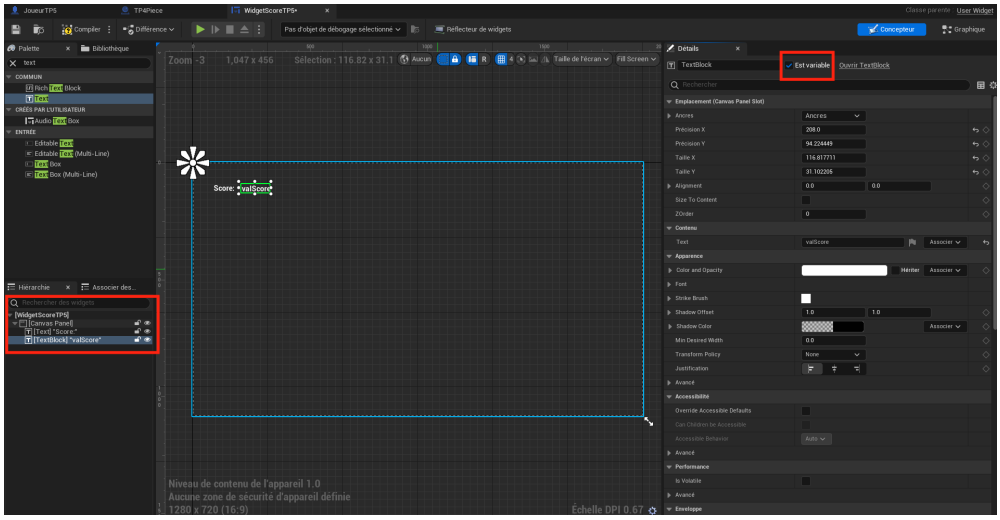
Afin de simplifier au maximum, le widget sera initialisé et géré dans la classe du joueur. Ce n'est pas une organisation pratiquée habituellement, mais elle a l'avantage de permettre de comprendre la génération de widget et de simplifier la mise à jour.

Stratégie pour l'affichage du score basé widget :

1. Création d'une nouvelle classe de type `User Widget` qui contiendra le texte à afficher.
2. Au lancement du jeu, la classe *Blueprint* du joueur va construire le widget.
3. Affichage à l'écran du widget (toujours par le joueur).
4. Toujours dans la classe du joueur, une référence sur le widget doit être enregistrée dans une variable afin de pouvoir mettre à jour les informations du score ensuite.
5. Enfin le joueur mettra à jour une partie du widget grace à la référence de l'étape 3.

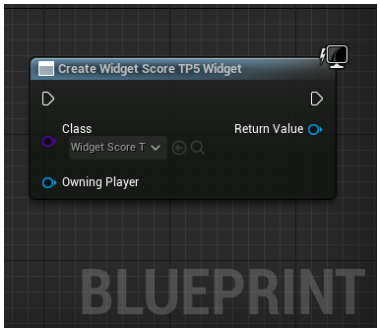
2.1 Pour réaliser la première étape, dans le *content browser*, ajoutez une nouvelle classe de type `UserWidget` que vous nommerez `WidgetScoreTP5`.

2.2 Ouvrez l'objet précédant, et rajoutez les composants de façon à obtenir la structure comme sur l'image suivante où les zones en rouge sont des indications sur la constitution de la scène.



Afin de pouvoir afficher le widget à l'écran, il est nécessaire qu'il soit construit (étape 1). Comme mentionné précédemment, la classe du joueur va être chargée de cette construction lors du lancement du jeu (même si ce n'est pas une pratique habituelle).

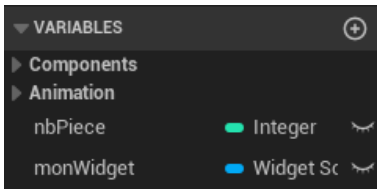
2.3 Dans la classe du joueur, à la suite des instructions déjà existantes de l'event `Begin Play`, appeler la fonction de création de widget illustré ci-dessous (étape 2).



2.4 Pour que l'affichage soit effectué sur l'écran du jeu (étape 3), il reste à appeler la fonction permettant l'ajout à l'écran : `addToViewPort`.

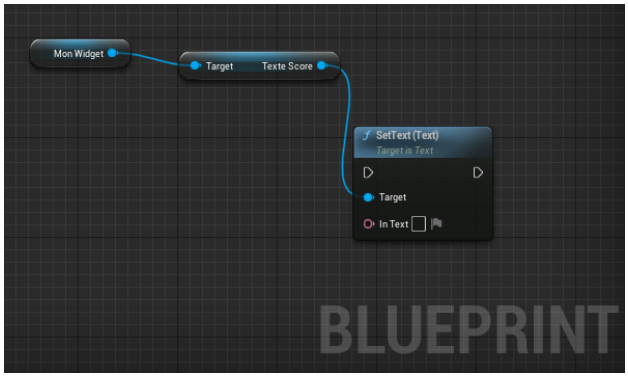
Si tout se passe bien, vous devriez voir apparaitre le score de manière statique à l'écran. Pour la suite, afin de permettre la mise à jour du widget il est nécessaire de sauvegarder, dans une variable du joueur, une référence sur le widget (étape 4).

2.5 Toujours dans la classe du joueur, ajoutez une variable de type `WidgetScoreTP5` que vous nommerez `monWidgetScore`.



2.6 Enregistrez dans cette variable `monWidgetScore`, le widget créé dans les questions précédentes dans la suite `Begin Play` du joueur.

2.7 Enfin pour réaliser la dernière étape, vous pourrez désormais mettre à jour le score depuis la fonction `gagnePiece` en utilisant les instructions suivantes :



Partie III: Complément au jeu

Pour compléter le jeu vous aurez à ajouter les éléments suivants :

— Ajout d'un énnemi dans la scène (sur la base du joueur).
— Pièce bonus permettant d'augmenter la taille du joueur.
— Ajout d'un nombre de vies en haut à droite de l'écran.