

DOCUMENTATION – PROTO PLATFORMER

Description :

Ce projet nommé « **PROTO Platformer** », et un prototype de projet Unreal Engine 5.4.3, permettant la facilitation de la création d'un platformer en TPS.

Mécanique Implémentées :

- Système de jump configurable
- Système de dash
- Système de crouch/slide
- Système de score
- Système de gestion de la vie
- Système de timer
- Objets collectables (point pour le score, dégâts et vie)
- Ai basique se déplacent uniquement
- Pause Manager
- Audio Manager
- Save Manager
- End Game

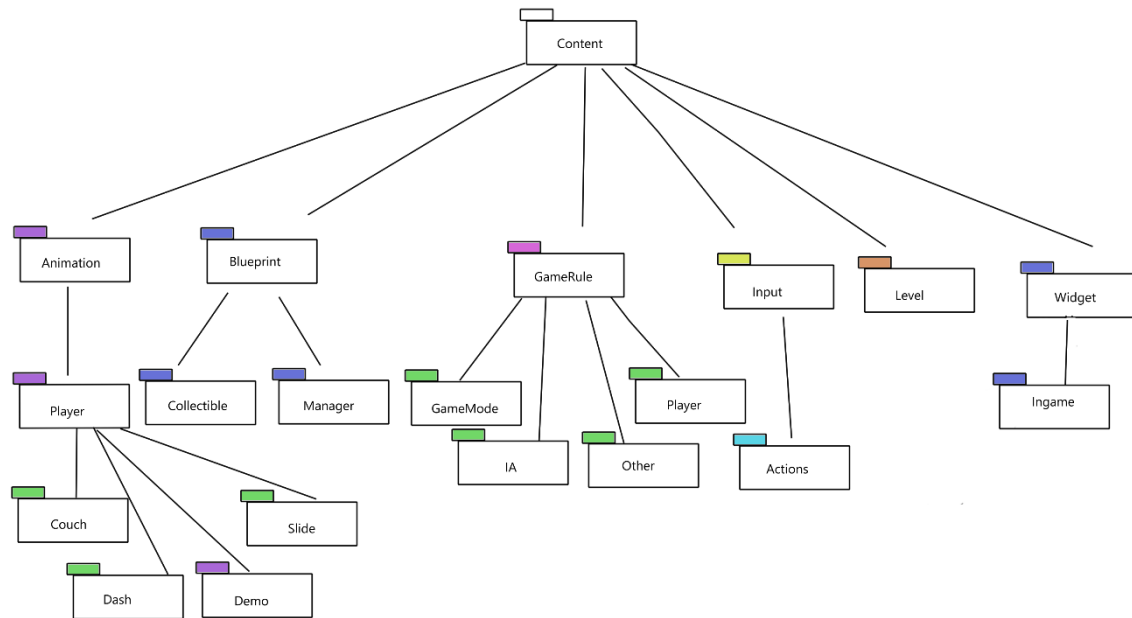
Sommaire :

- Architecture ----- p2
 - o Projet ----- p2
 - o Plugins ----- p3
- Fonctionnement ----- p4
 - o Projet ----- p4
 - o Joueur ----- p6
 - o Game Mode ----- p8
 - o Objets Collectables ----- p11
 - o IA ----- p13
- Conclusion ----- p17

Architecture :

- Projet :

Diagramme architecture projet.



- Plugins :

Diagramme architecture plugins - IASystem

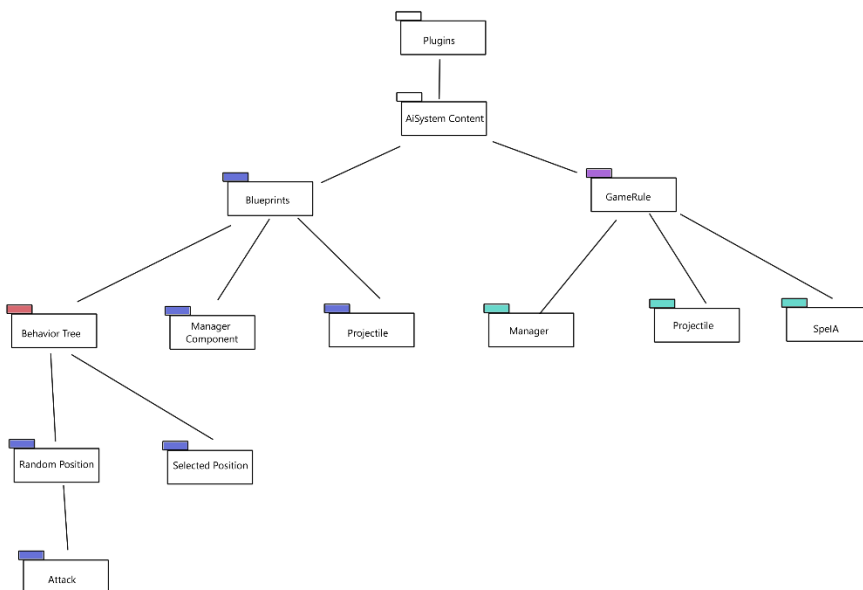


Diagramme architecture plugins – DashSysytem

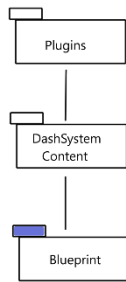


Diagramme architecture plugins – HealthSystem

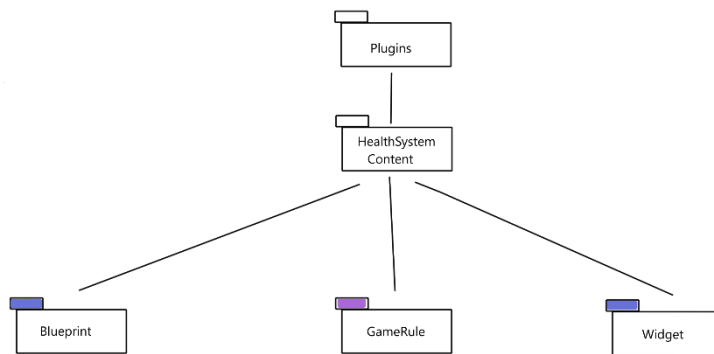
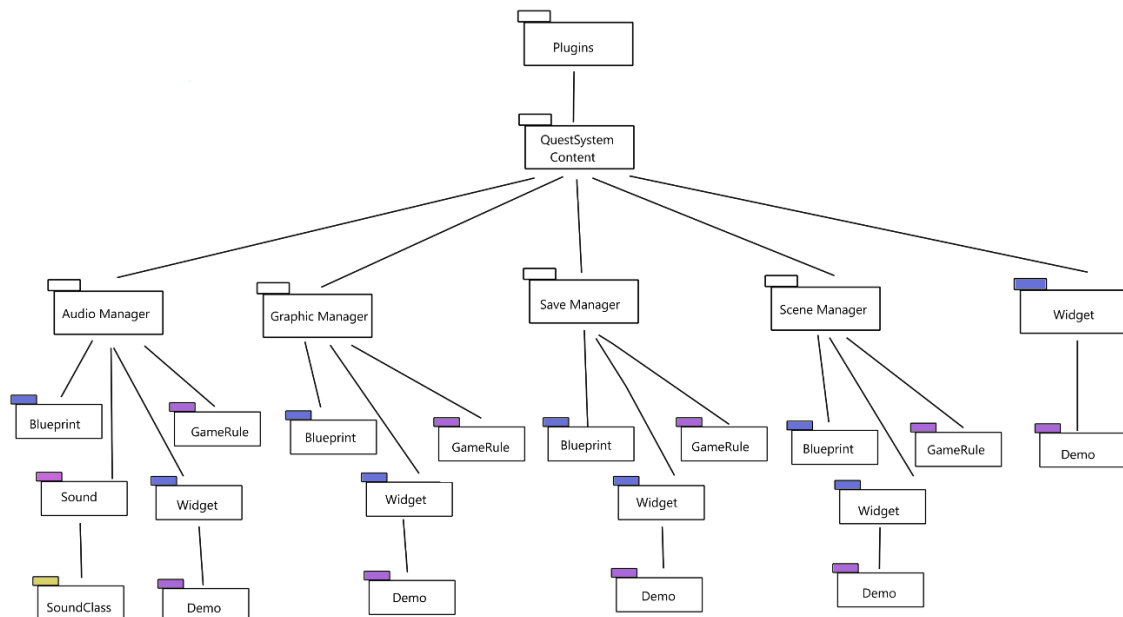


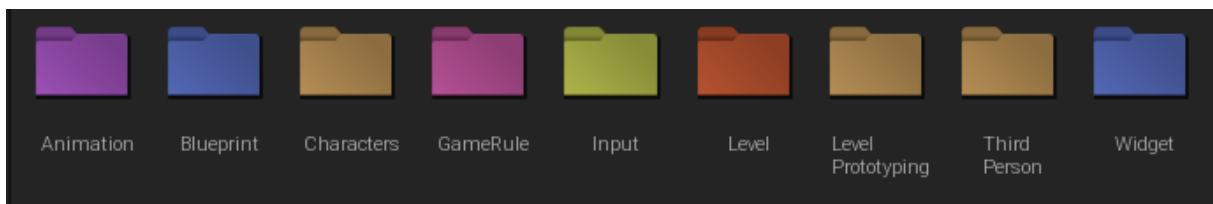
Diagramme architecture plugins – SettingManager



Fonctionnement :

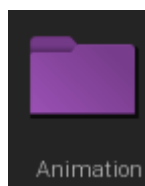
- **Projet :**

Nous allons tous d'abord regarder l'architecture du projet pour être sûr de ne pas se perdre.



Pour commencer tous les dossiers colorés sont des dossiers non-générer de base et dont le code couleur à une signification :

- **Violet :**



Comme son nom l'indique, vous trouverez les animations ainsi que l'Animation Blueprint (AnimBP) qui sera très important pour relier les actions à une animation.

- **Bleu :**



Le dossier « **Blueprint** » contient tous les éléments de code réalisé pour le prototype. Ce sera dans ce dossier que vous pourrez retrouver par exemple les point de score, que vous pourrez personnaliser (graphiquement) et poser dans la scène.

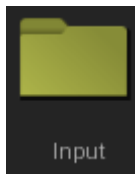
Le dossier « Widget » quant à lui contient tous les éléments UI/UX à mettre en place pour le jeu.

- **Rose :**

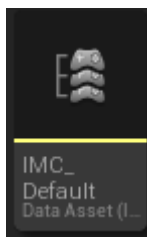


Ce dossier sera important pour vous, car il contient les tableaux de données modifiables pour rendre votre jeu unique (ex : hp du joueur).

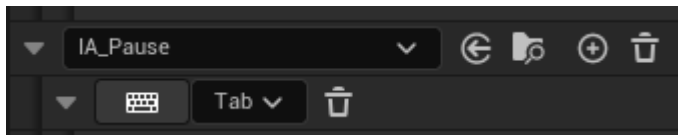
- **Jaune :**



Comme son nom l'indique, il contient toutes les informations relatives aux touches possibles. Vous pourrez modifier les touches configurées dans le fichier :



Il vous suffira de modifier la touche rentrée ici :



- **Orange :**

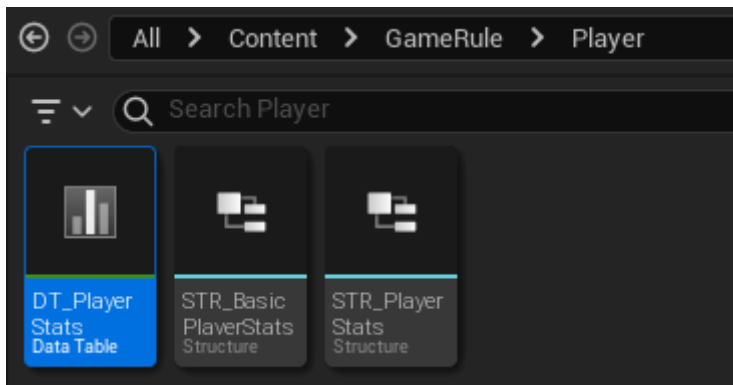


Enfin le dossier « Level » contiendra toutes vos scènes.






- **Joueur :**

Dans le cadre du joueur vous aurais 3 fichiers à modifier pour pouvoir le personnaliser comme bon vous semble.

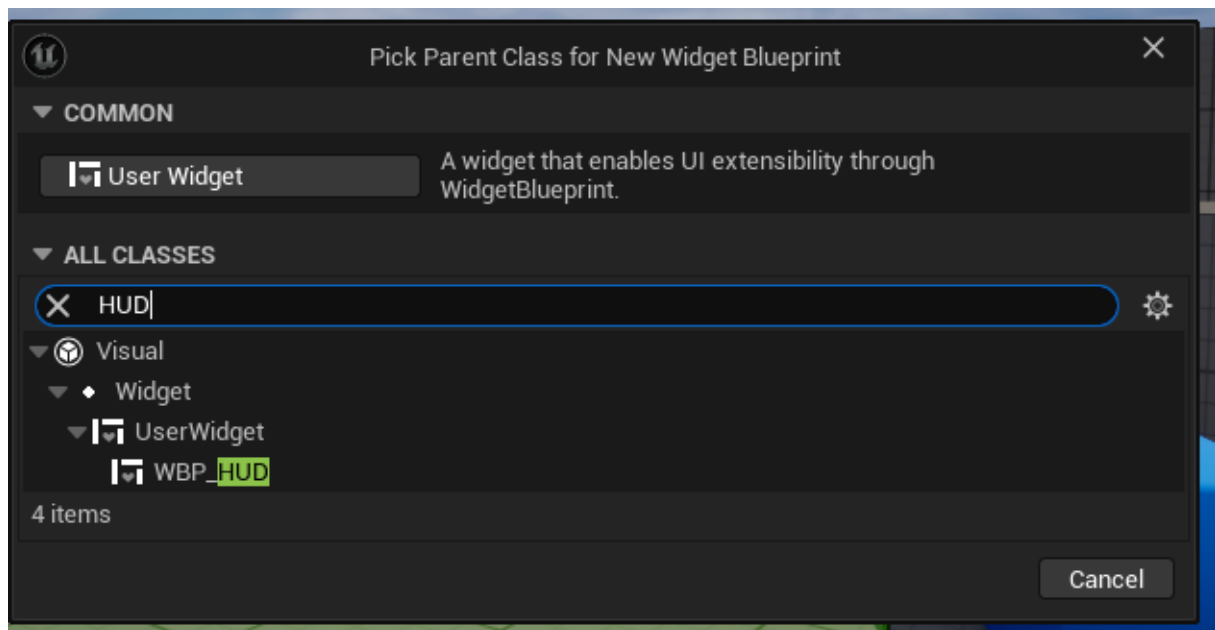
Pour la partie changement de valeur, vous devrais ouvrir le tableau « **DT_PlayerStats** »



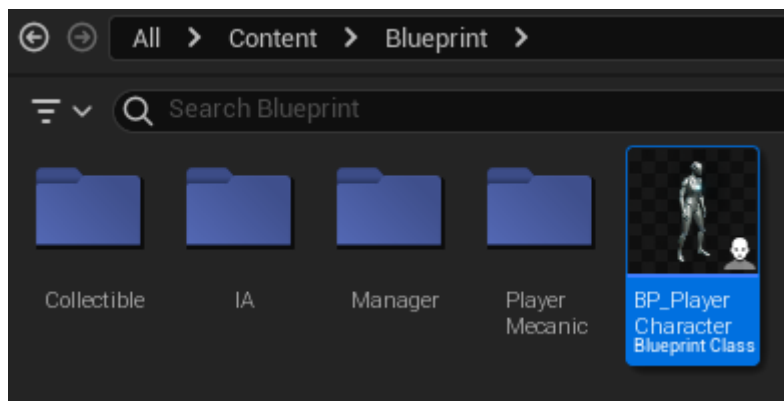
Vous y trouverez dedans les différentes valeurs modifiables du joueur.

PlayerStats	
IsDashActive	<input checked="" type="checkbox"/>
DelayDash	1.0
PowerDash	2.5
JumpSpeed	500.0
JumpHoldTime	0.3
IsCrouchActive	<input checked="" type="checkbox"/>
CrouchHalfHeight	50.0
CrouchSpeed	150.0
IsSlideActive	<input checked="" type="checkbox"/>
SlidePower	1.25
HUD_Class	WBP_HUD     

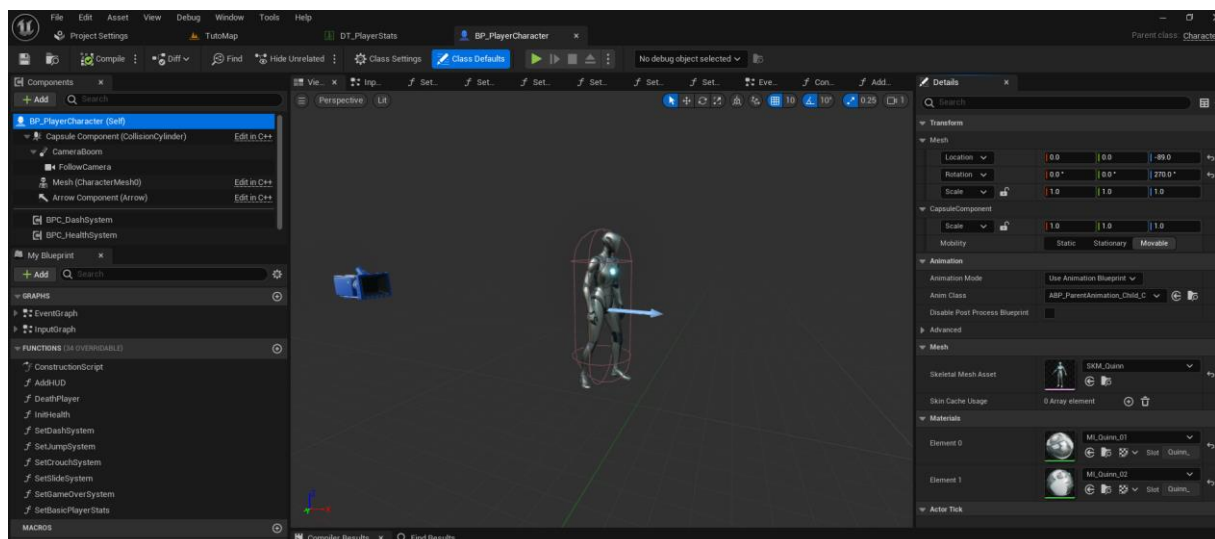
Vous pourrez à l'aide du tableau activé ou non par exemple le système de dash ou encore changer la vitesse de base du joueur. Pour les variables demandant un widget (ex : HUD) il vous faudra, si vous créez le vôtre, utilisé le même parent, dans le cas de l'HUD vous devrez forcément faire un enfant de « **WBP_HUD** ».



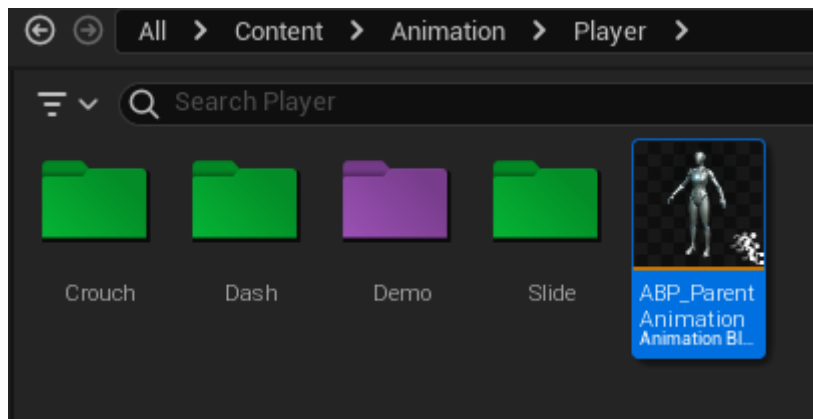
Vous pourrez également modifier le visuel du joueur en ouvrant le Blueprint « **BP_PlayerCharacter** ».



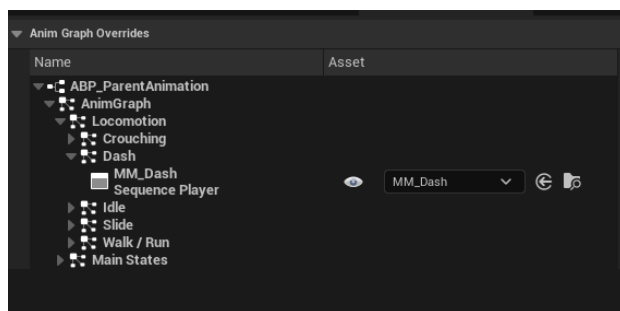
Une fois ouvert, vous aurez accès au différent composant modifiable du joueur.



Enfin il ne faut pas oublier de faire un AnimBP et de lui assigner pour pouvoir avoir vos propres animations sur le personnage, il faudra que ce soient des enfants de « **ABP_ParentAnimation** », vous avez la partie démo qui vous montre un exemple.

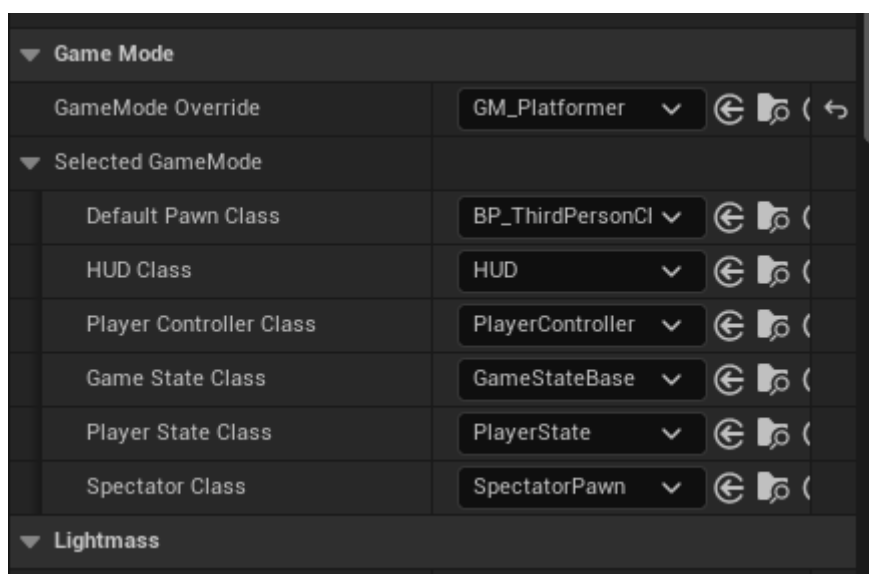


Et imaginons que dans cette enfant vous vouliez changer l'animation du dash, il vous suffira de changer la valeur de « **MM_Dash** ».

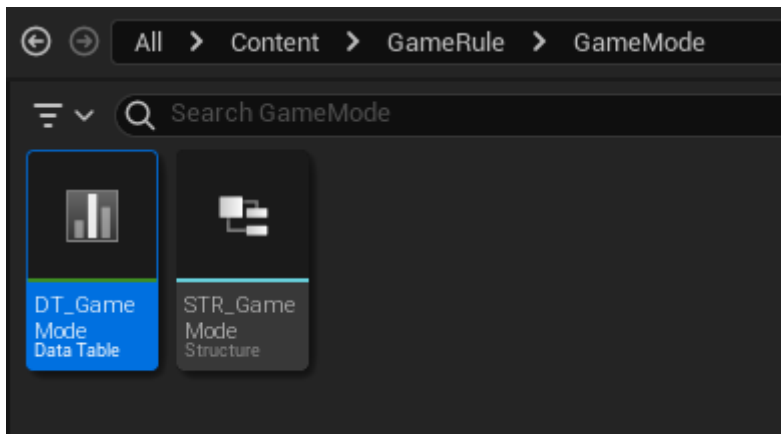


- **GameMode :**

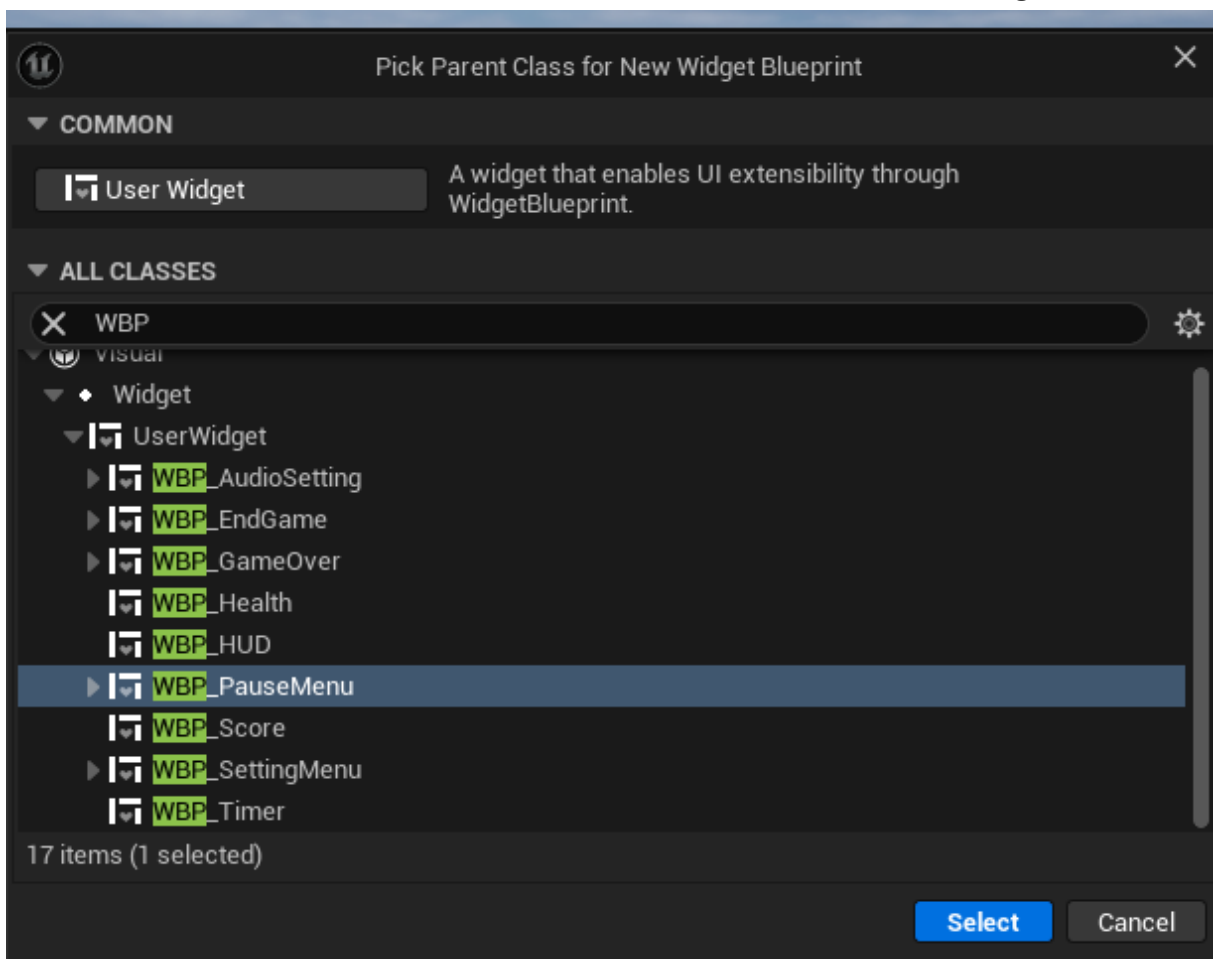
Pour la partie **GameMode**, il vous suffira de sélectionner le « **GM_Platformer** » dans les « **WorldSetting** » de l'éditeur Unreal Engine.



Puis, pour toutes la partie customisation, vous pourrez modifier les valeurs du tableaux « **DT_GameMode** ».

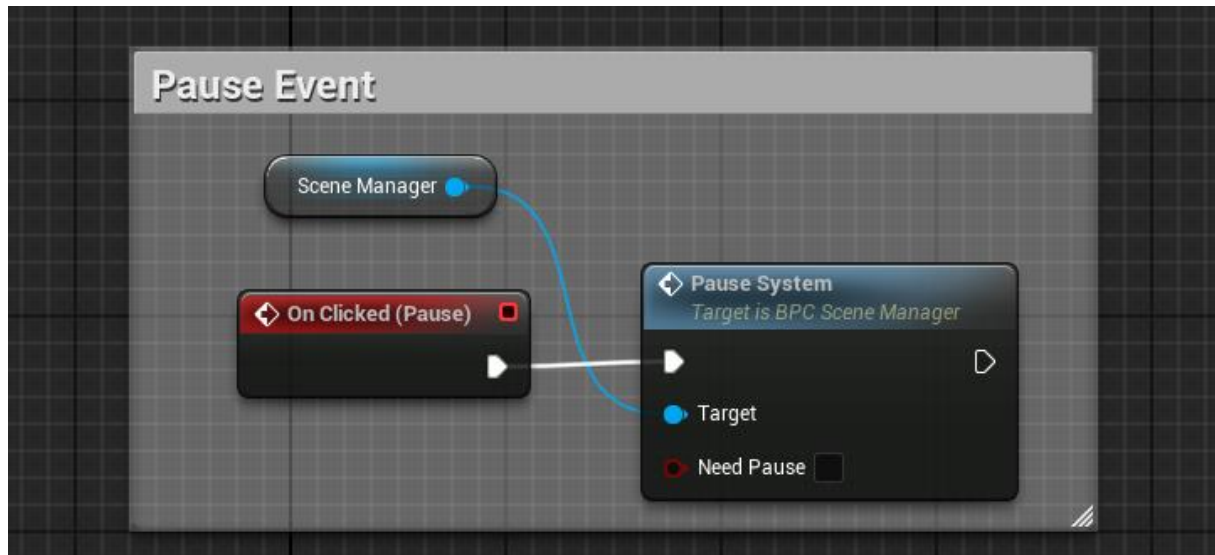


Une fois dedans, vous pourrez décide de si vous voulez activer ou non le **timer**. Il vous faudra également référencer les différente UI du jeu (**AudioSetting**, **GameOver**, **EndGame**, **Pause**, **Setting**) qui seront des enfants de widget créé dans le plugin **Setting Menu**, il vous faudra les sélections au moment de la création de votre widget.

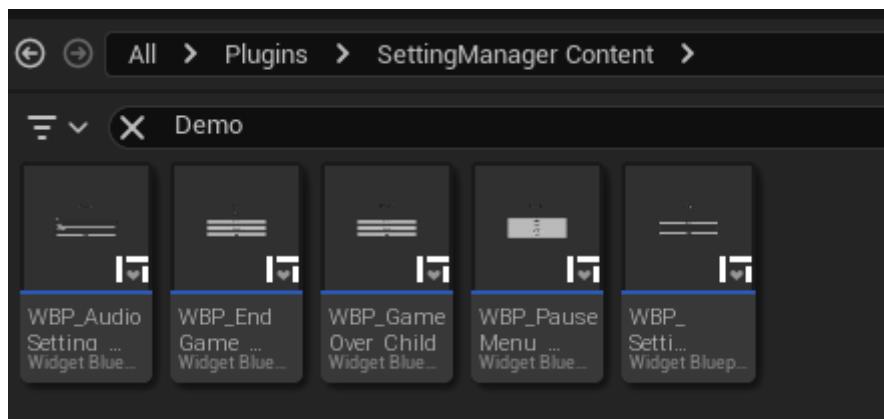


Ici c'est un exemple de quoi sélectionner comme parent si vous voulez créer un nouveau Menu Pause. Une fois créer il vous faudra faire vos UI et les connecté vous-même aux fonctions.

Exemple du système de pause.



Pour mieux comprendre je vous invite à parcourir les dossiers des « **Demo** » dans le plugin « **SettingManager** » qui vous permettra de mieux comprendre.

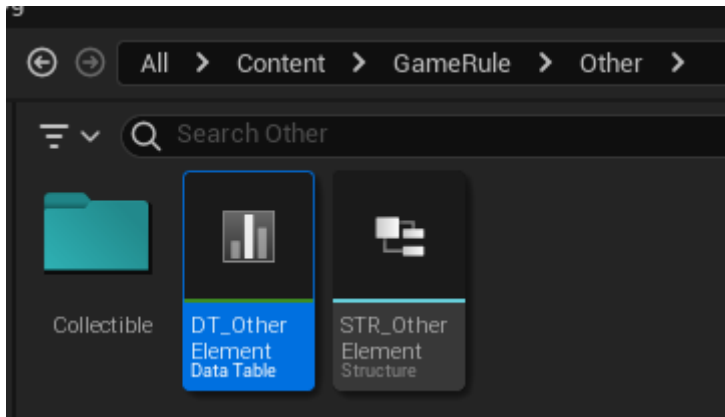


Une fois tous cela fait, il vous suffira de renseigner vos widgets dans le tableau.

Row Editor	
GameMode	
▼ GameMode	
IsTimerActive?	<input checked="" type="checkbox"/>
▼ AudioManager	
AudioWidgetClass	WBP_AudioSetting_Child
▼ SceneManager	
GameOverWidgetClass	WBP_GameOver_Child
EndHGameWidgetClass	WBP_EndGame_Child
PauseWidgetClass	WBP_PauseMenu_Child
SettingWidgetClass	WBP_SettingMenu_Child

- **Objets collectables :**

Comme pour les autres parties, pour le remplissage de valeur il vous faudra passer par le tableau de valeur « **DT_OtherElement** ».



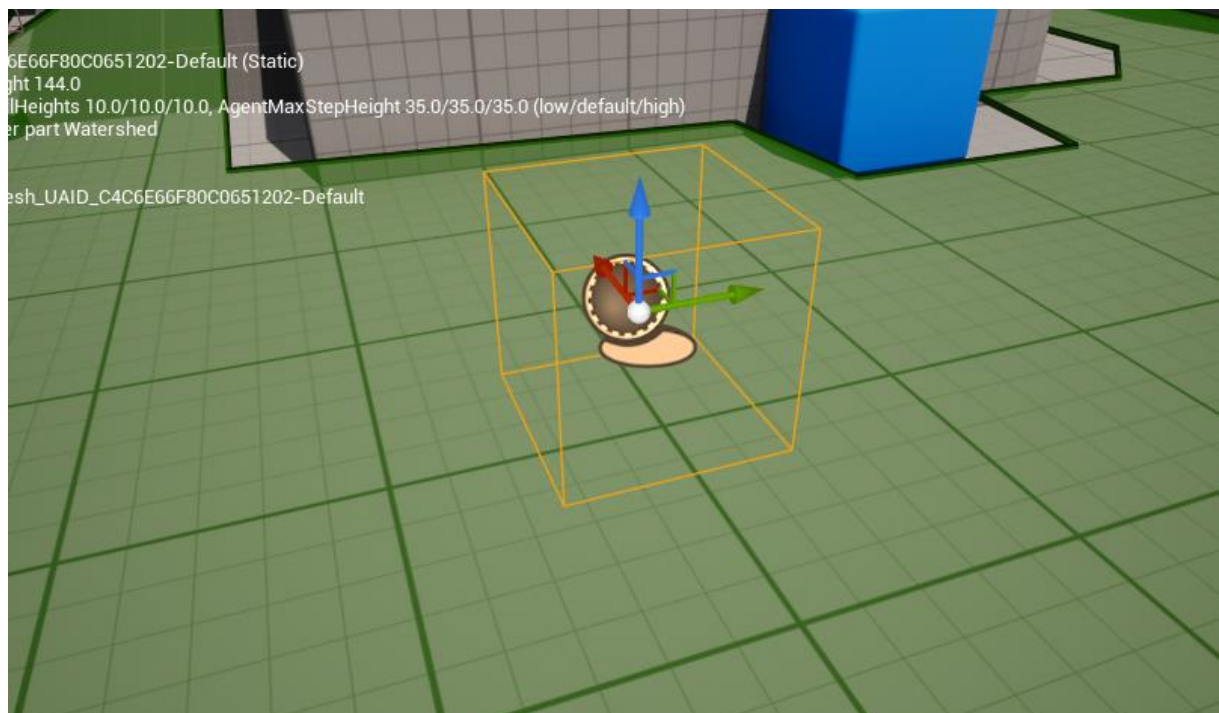
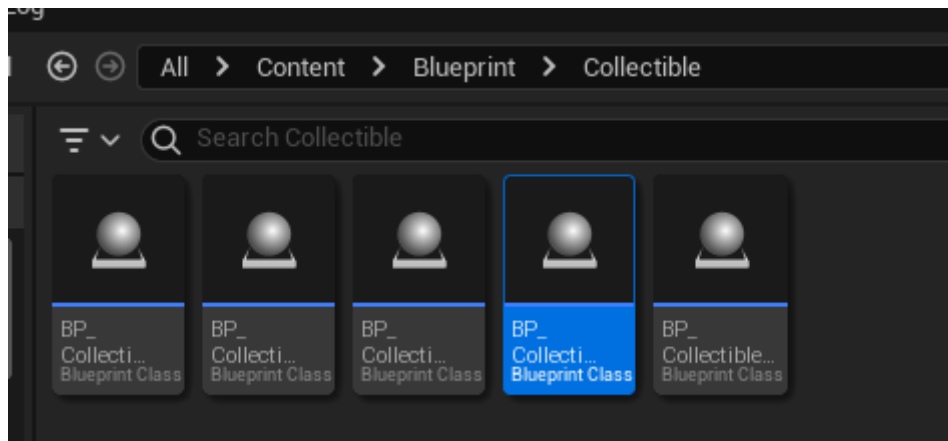
A l'intérieur du tableau vous pourrez créer des profils d'éléments pour avoir ainsi des valeurs uniques.

▼ OtherElement	
▼ CollectibleScore	2 Map elements + -
▼ normal	1 members ▼
Score	15
▼ big	1 members ▼
Score	50

Par exemple dans le tableau correspondant au Collectable de score, nous avons 2 profils différents (« **normal** » et « **big** »). Vous pourrez l'assigné à votre objet pour qu'il puisse en prendre la valeur. Pour créer de nouveau tableau il vous suffira de cliquer sur (+) au niveau du tableau voulue.

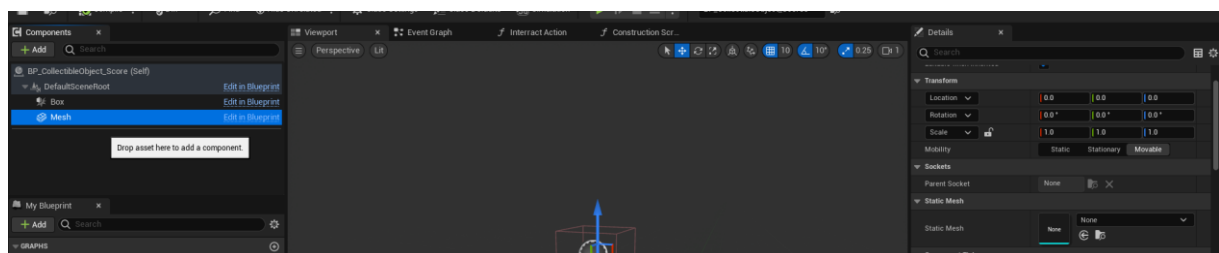
▼ CollectibleScore	2 Map elements +
--------------------	------------------

Avant de pouvoir utilisé votre profil, il vous faut avant tout placer votre élément dans la scène. Pour cela il faudra vous rendre dans le dossier « **Collectible** » et glisser-déposer celui qui vous intéresse.

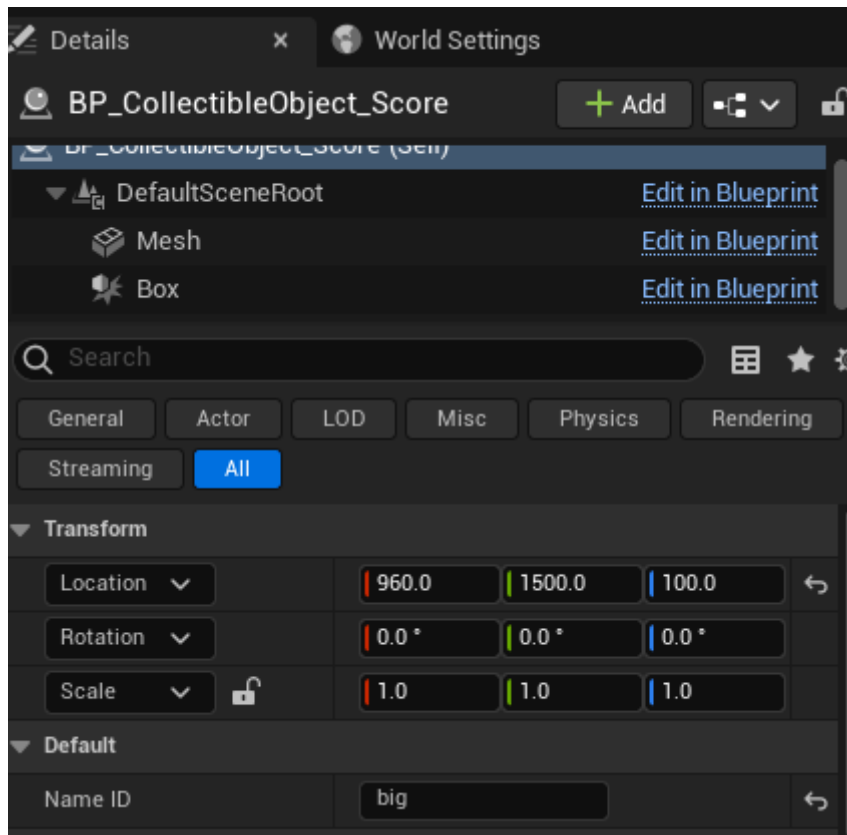


Par exemple ici j'ai sélectionné celui de score, le voici dans la scène.

Pour le modifier graphiquement, il vous faudra ouvrir le blueprint, et y placer votre nouveau **mesh**. ATTENTION pensez à redéfinir la taille du box collider (carré jaune) qui correspond à la hitbox de l'objet.

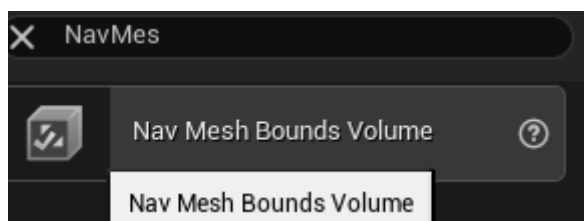


Une fois tous cela fait, dans votre scène vous pourrez assigner un profil d'éléments à votre objet en rentrant son nom dans la variable « **NameID** ».

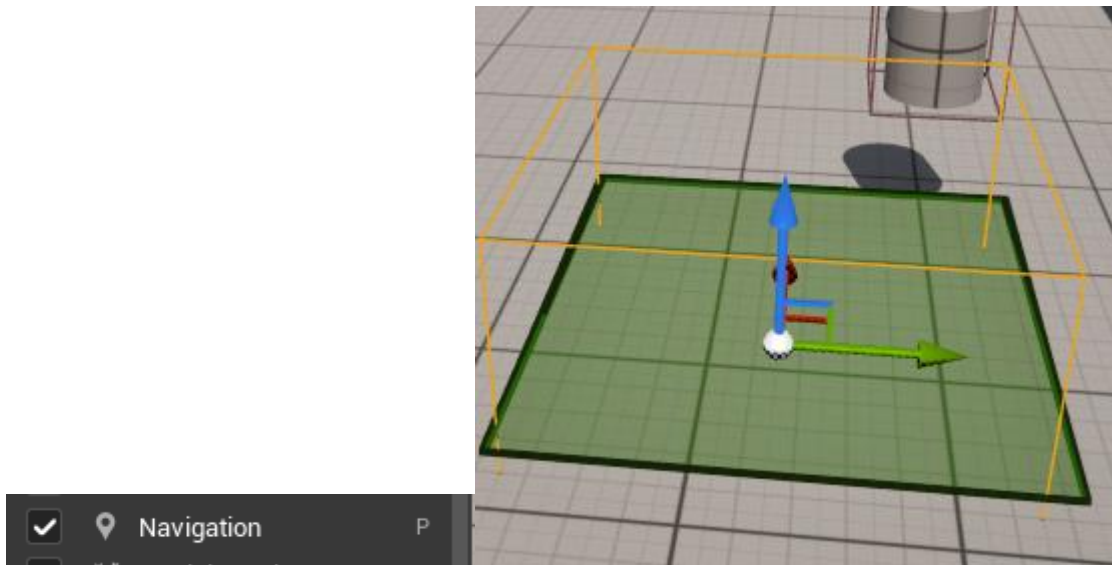


- IA:

Avant toute chose, pour vous assurer du bon fonctionnement des IA, il vous faut ajouter un **NavMeshBoundsVolume** (il est normalement de base dans la scène de tuto)

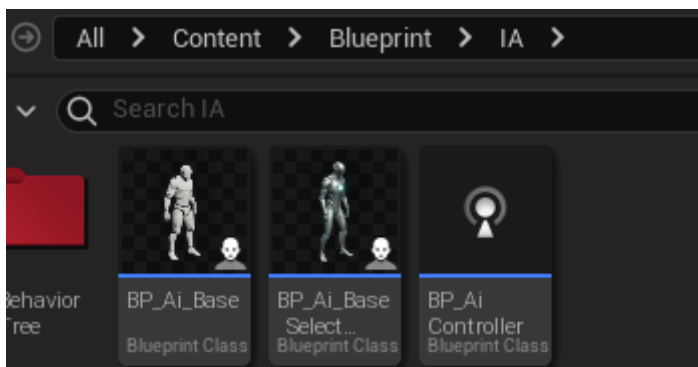


Ce NavMesh correspond à la zone possible de déplacement de l'IA, pour bien voir cela, je vous conseille d'appuyer sur « **P** » dans l'éditeur ou bien dans le menu show -> Navigation, cela vous permettra de prévisualiser la zone.



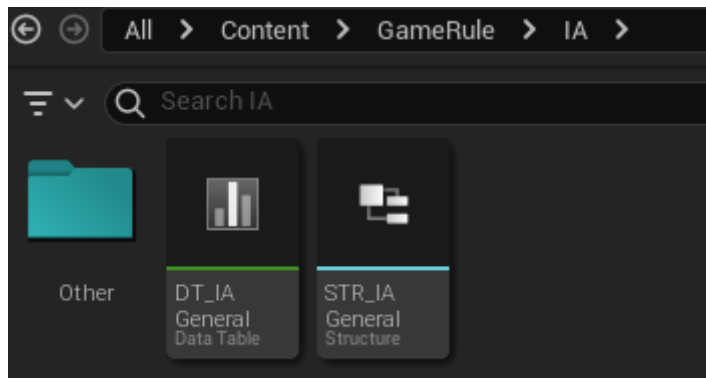
Il vous suffira plus qu'agrandir le **NavMesh** pour qu'il couvre la zone où vous voulez faire déplacer l'IA (la zone vert correspond à son endroit possible de déplacement pour l'IA).

Pour ce qui est de l'IA, vous pourrez modifier tous l'aspect graphique directement sur l'IA en créant un enfant de l'IA.

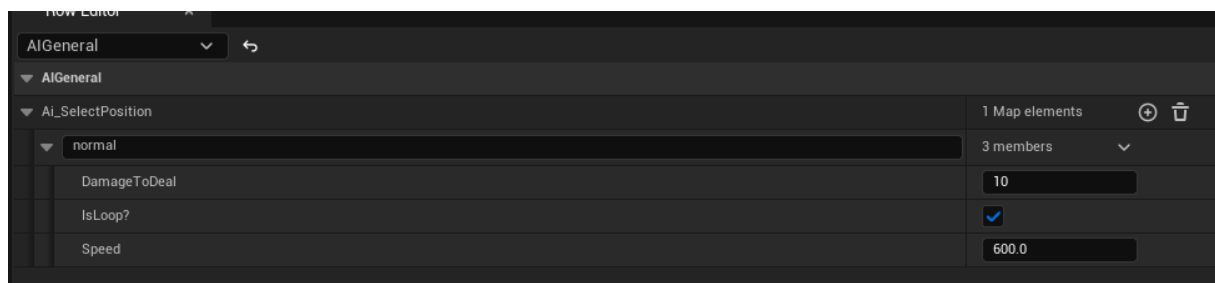


L'IA customisable est la 2^e (**BP_Ai_Base_SelectPosition**). En l'ouvrant vous pourrez modifier le **mesh**, et également changer son animation (penser à garder une cohérence entre l'animation et le mesh).

Pour ce qui est relatif aux valeurs de l'IA (**hp**, **dmg**, etc), cela se trouve au niveau des « **GameRule** » dans le fichier « **DT_IAGeneral** ».



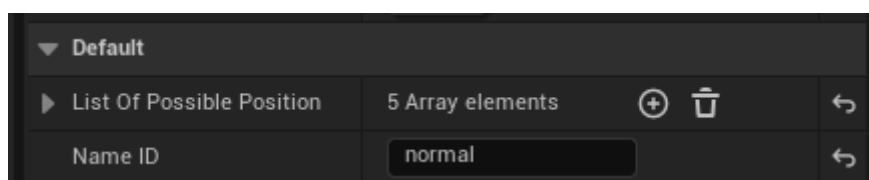
Vous pourrez personnaliser des profils d'IA et ainsi les appliquer sur l'IA.



Par exemple le profil « normal » dans le tableau « **Ai_SelectPosition** » contient une valeur de dégâts de 10, une vitesse de 600 m/s et « **isLoop ?** » vrai ce qui signifie que les déplacements seront en boucle. Nous pourrions également en créer une autre avec le nom que nous souhaitons en appuyant sur le (+) :



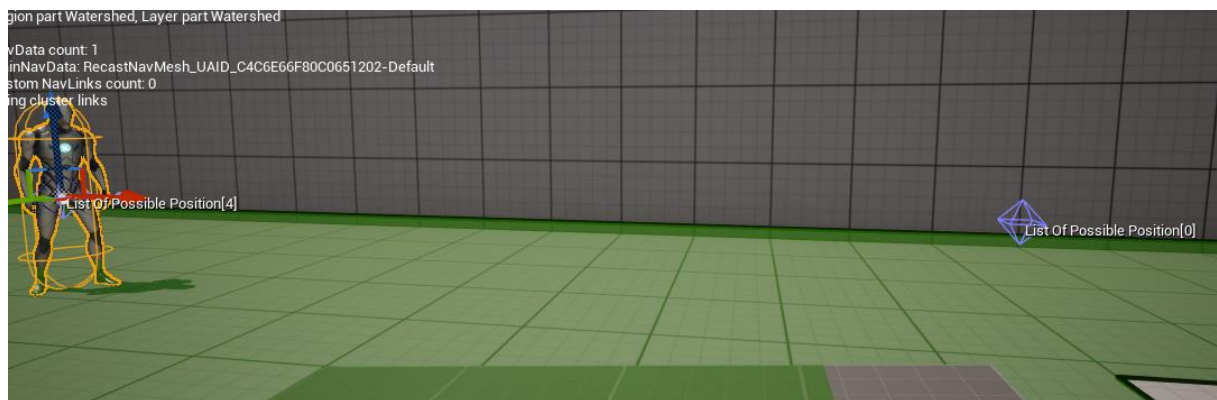
Puis une fois le profil configuré, il nous suffit de l'appliquer à l'IA dans la scène, en modifiant sa valeur de « **NameID** » :



Passons maintenant à la partie importante qui est, la création et le placement des points de passage de l'IA. Vous trouverez la liste des positions nommé « **List Of Possible Position** » au côté de « **NameID** ».

List Of Possible Position	5 Array elements			+	🗑	↩
▶ Index [0]	820.0	0.0	0.0	▼	↩	↩
▶ Index [1]	1510.0	710.0	0.0	▼	↩	↩
▶ Index [2]	1730.0	2680.0	0.0	▼	↩	↩
▶ Index [3]	-100.0	2420.0	0.0	▼	↩	↩
▶ Index [4]	0.0	0.0	0.0	▼	↩	↩

Voici le tableau de position des points. Pas de panique pour le placement dans la scène, les points seront pris par le positionnement des Gizmo.



Voici par exemple ou sont placé le point à l'index 0 et celui à l'index 4. Pour les voir il faudra au préalable sélectionner l'IA en question dont vous voulez déplacer ses points. Vous pourrez les déplacer avec les flèches d'axes comme un objet classique en les sélectionnant.

Si vous voulez ajouter il suffit d'appuyer sur le bouton (+) au niveau du tableau. Le point apparaîtra automatiquement au niveau du centre de l'IA.



Voici toutes les informations nécessaires pour créer facilement une IA.

Conclusion :

Voici toutes les informations importantes à prendre en compte pour l'utilisation du Prototype de **Platformer**. N'hésitez pas à ouvrir les différents fichiers pour mieux comprendre le fonctionnement mais faite attention à rien casser.