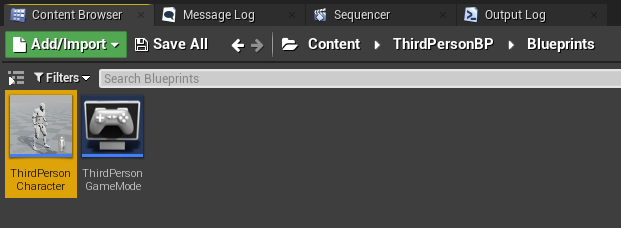
Système de dialogue, Plugin UE4

Dans ce document nous allons voir comment créer un système de dialogue grâce à un plugin développé pour effectuer cette tâche.

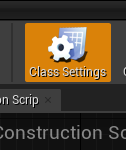
Prérequis :

* Avoir copié le Plugin dans le dossier Plugin du projet.
* Avoir créé un projet Third Person.

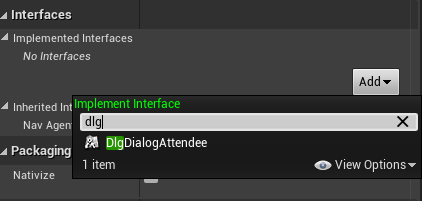
1. Edition du Blueprint du personnage



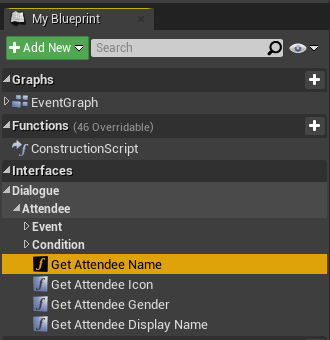
Double cliquer sur le ThridPerson Character Blueprint



Cliquer sur Class Settings, c’est là que nous allons rajouter l’appel à la fonction DlgDialogAttendee

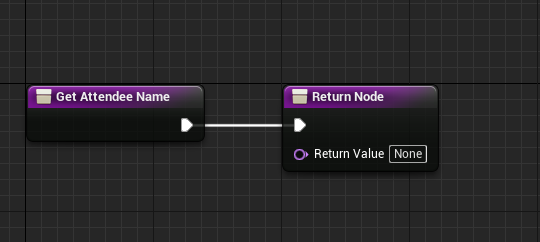


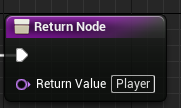
Cliquer sur Add puis taper Dlg… Le nom de la fonction devrait apparaître.



Dans la fenêtre My Blueprint, on peut constater que l’interface à la fonction a été rajoutée. On va rajouter cela au script de construction du personnage joueur.

Simplemenr double cliquer sur Get Attendee Name et cela va le rajouter au script de construction.

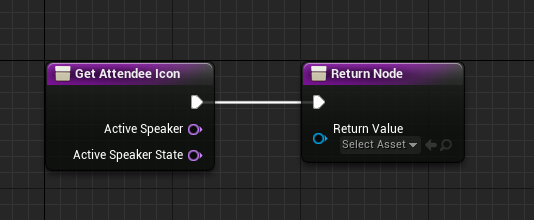




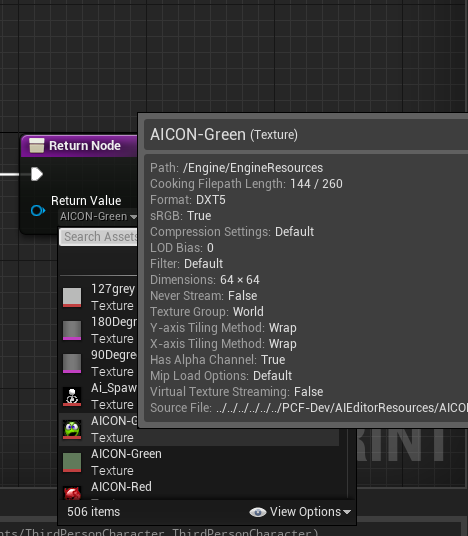
Remplacer None par un nom de variable pour le joueur. Ici Player



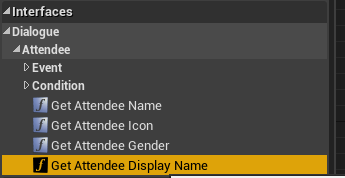
Double cliquer sur Get Attendee Icon



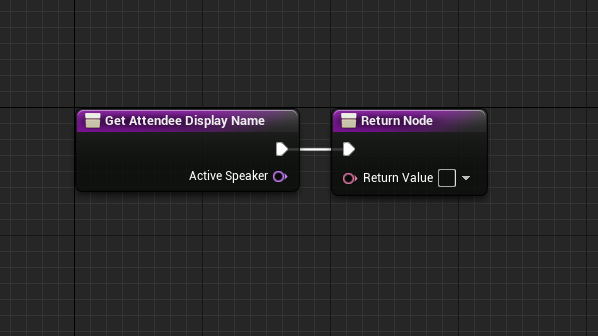
Cela créé le script d’initialisation de la fonction Get Attendee Icon



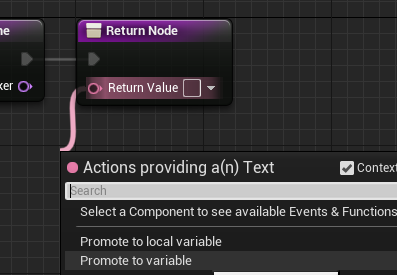
Sélectionner une icône pour le Player



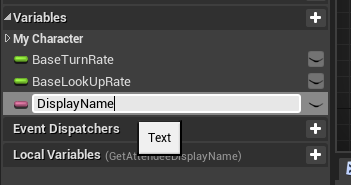
Double cliquer sur Get Attendee Display Name afin d’initialiser la fonction



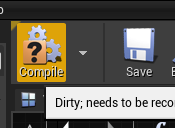
Voilà ce qui devrait apparaître. On va maintenant créer une variable de retour afin de pouvoir retourner le nom du joueur.



Sélectionner Promote To Variable



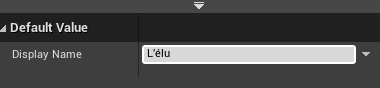
Que l’on nommera DisplayName



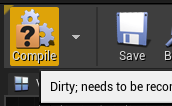
Compiler/Sauver



On peut alors éditer le nom du joueur.

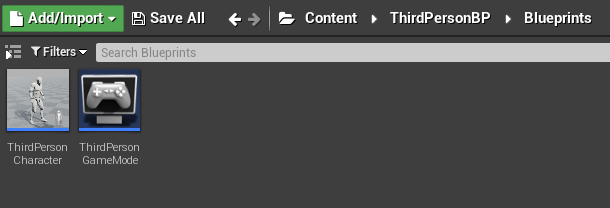


Par exemple…

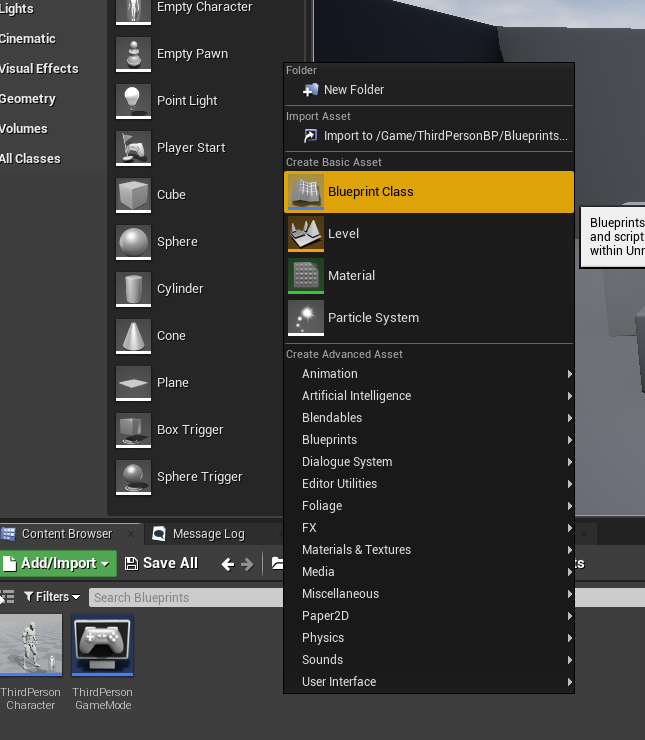


Compiler/Sauver

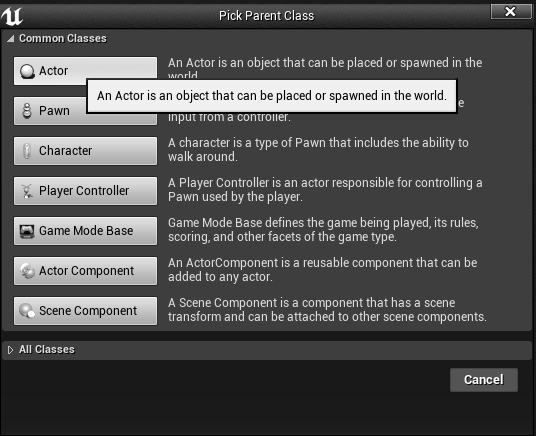
1. Création du NPC dont on pourra hériter



On va revenir sur notre Content Browser, et on va rajouter une Class de type Actor



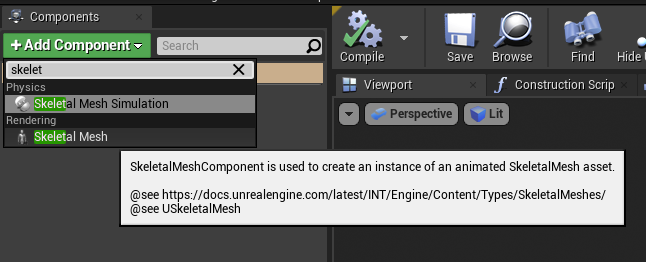
Clique droit -> Create Basic Asset -> Blueprint Class



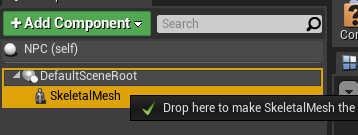
Choisir Actor



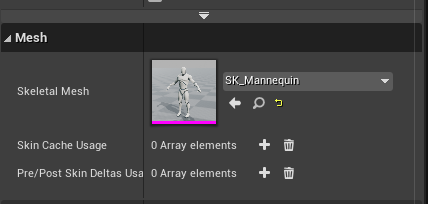
Que l’on va nommer NPC



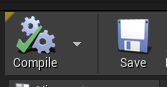
Double cliquer sur le Blueprint NPC et ajouter un component de type Skeletal Mesh



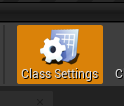
Drag and Drop SkeletalMesh sur le DefaultSceneRoot afin qu’il devienne Root



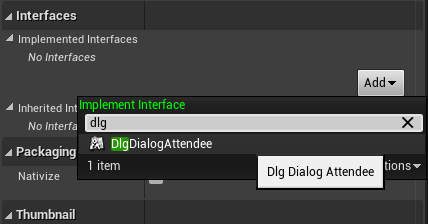
Dans le Mesh, sélectionner SK\_Mannequin par exemple



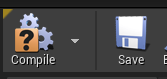
Save/Compile



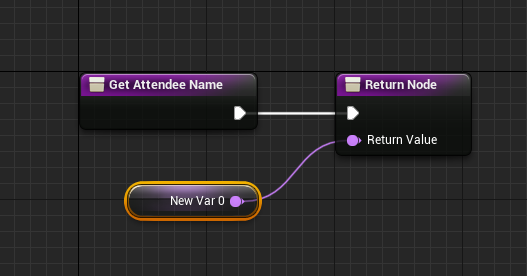
De la même manière que le perso principale, nous allons lui affecter des fonctionnalités du Plugin en cliquant sur Class Settings.



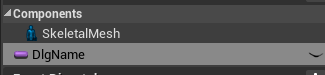
Ajouter DlgDialogAttendee



Compiler/Sauver pour activer les fonctions.



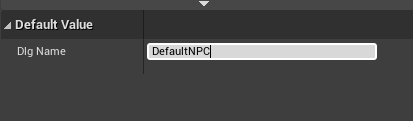
Double cliquer sur GetAttendeeName afin de créer l’initialisation de la fonction et lui affecter un nom de variable.



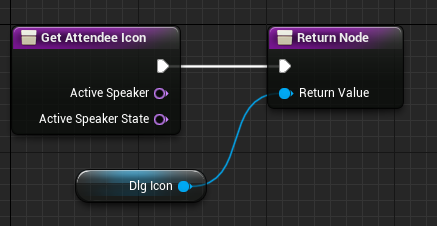
On la nomme DlgName



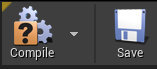
Compiler/Sauver afin de pouvoir modifier le contenu de la variable



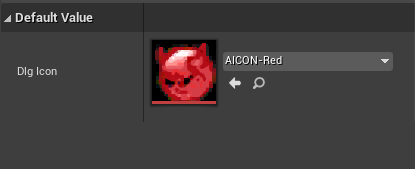
On lui affecte un nom par défaut.



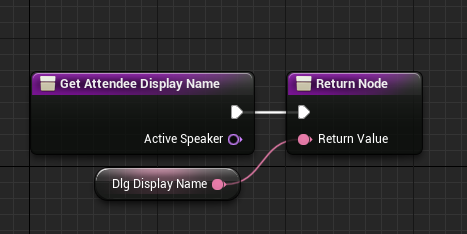
Double cliquer sur Get Attendee Icon, et lui affecter lui affecter une variable que l’on nommera DlgIcon afin de pouvoir l’exploiter dans les NPC hérités.



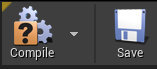
Compiler/Sauver afin d’affecter une valeur par défaut.



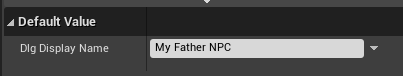
Par exemple.



Double cliquer sur GetAttendeeDisplayName et promouvoir en variable la Retun Value que l’on nommera DlgDisplayName

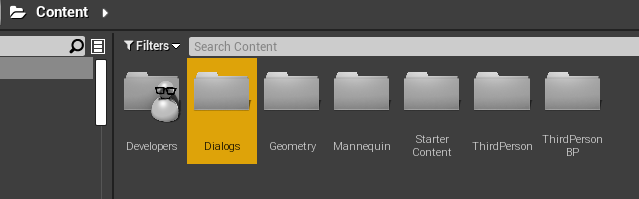


Compiler/Sauver afin d’affecter une valeur par défaut.

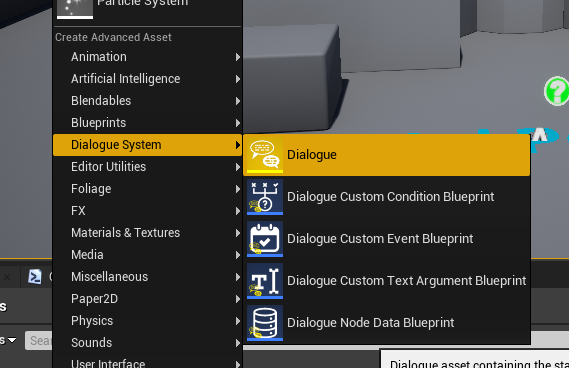


Nommer la variable DlgDisplayName avec My Father NPC par exemple.

1. Les ressources des dialogues



Tout d’abord, créer un dossier Dialogs à la racine de Content



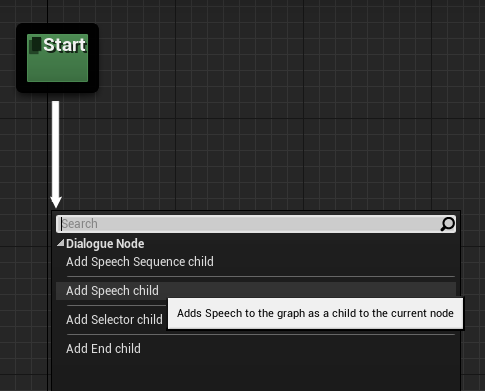
A l’intérieur du dossier, nous allons créer une ressource de type Dialogue



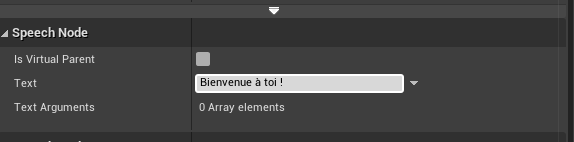
Que l’on nommera FirstDialog par exemple



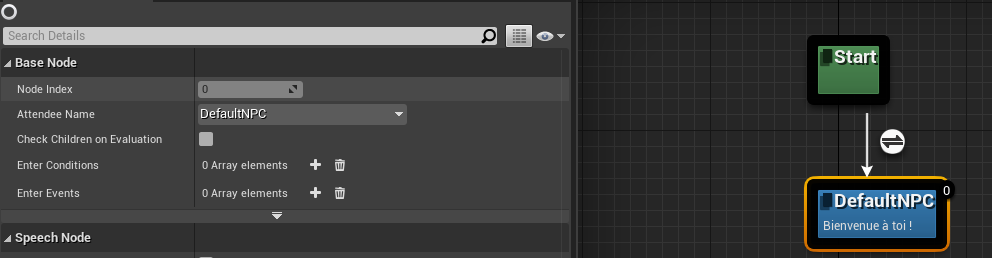
On double clique sur FirstDialog et on obtient la fenêtre ci-dessus qui va nous permettre d’éditer une séquence de dialogue.



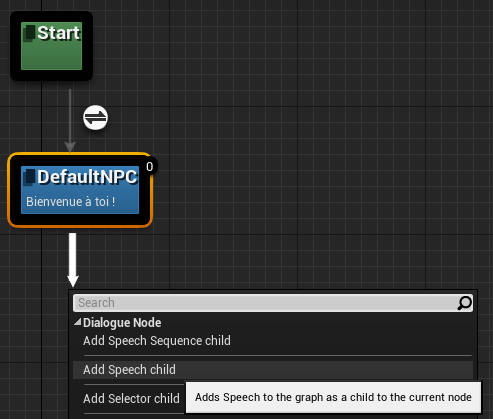
Créer un nœud et sélectionner Add Speech child



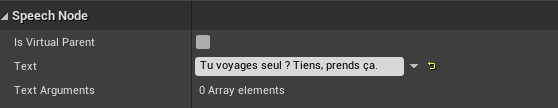
Ajouter un texte de bienvenue.



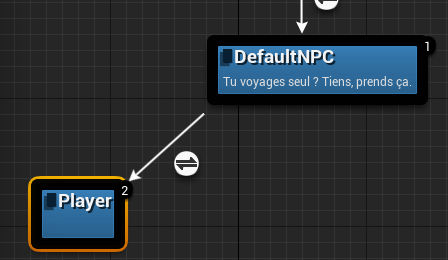
Sélectionner le Node 0 et lui affecter DefautNPC pour l’Attendee Name



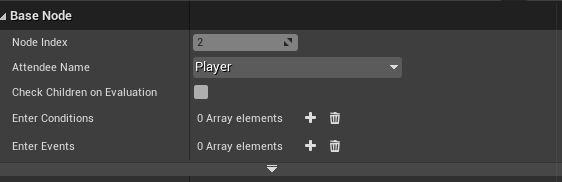
Ajouter un autre Speech child



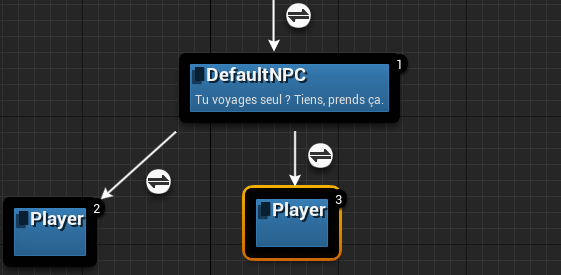
Ajouter un texte



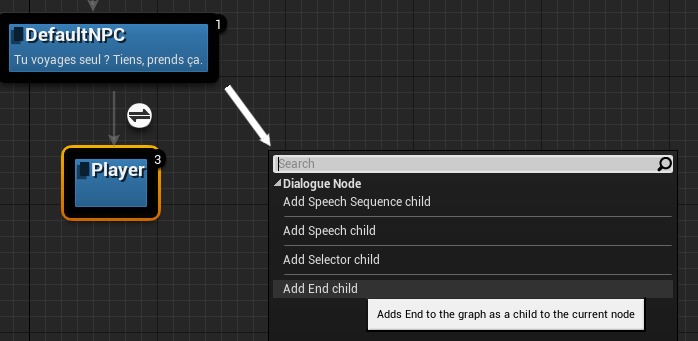
Ajouter un nœud, mais cette fois ci l’affecter au joueur (Player)



Comme ceci



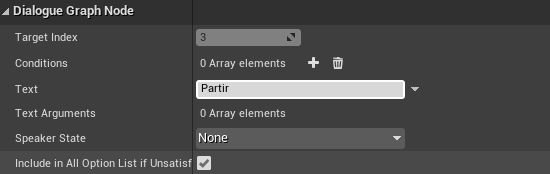
Ajouter un second nœud de type Speech child sur Player



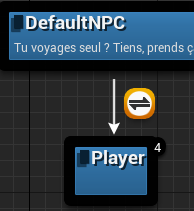
Ajouter un nœud de fin Add End child



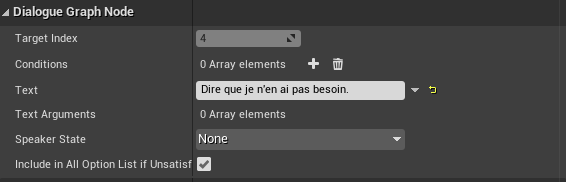
Cliquer sur l’icone de milieu de section afin de pouvoir créer un texte de fin



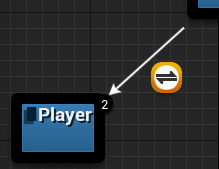
Partir s’affichera et permettra au joueur de quitter la conversation



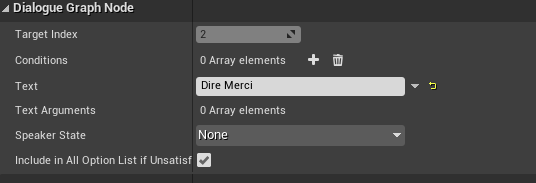
Sélectionner l’intersection suivante



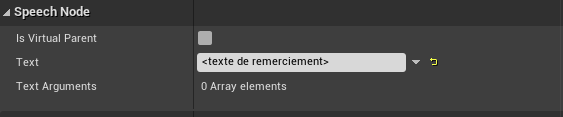
Donner un sens à tout ça…



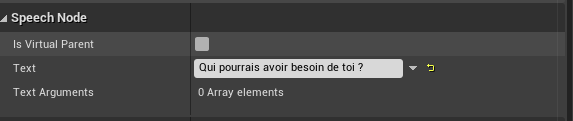
Sélectionner la première intersection



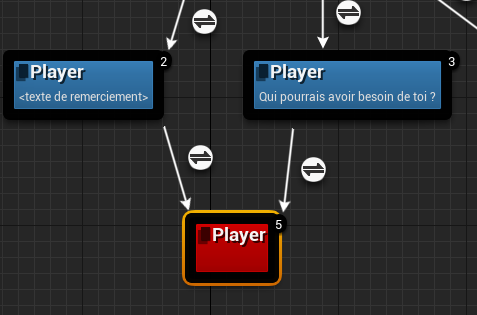
Afficher le choix de dire merci



Sélectionner le nœud Player 2 et ajouter un texte de référence entre <>

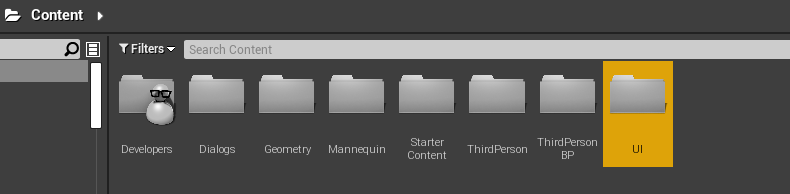


Sélectionner le nœud Player 3 et ajouter le texte

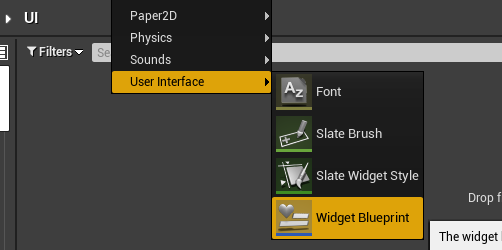


Ajouter un End child pour finir la conversation

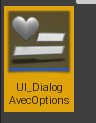
1. Interface utilisateur



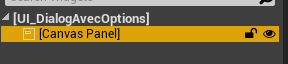
Dans le dossier Content créer un sous-dossier UI



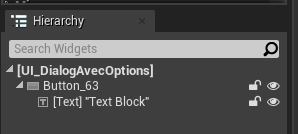
Dans ce dossier, ajouter un Widget Blueprint



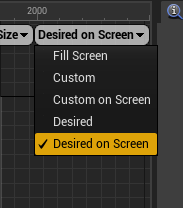
Nommer le Widget



On va supprimer Canvas Panel pour les tests



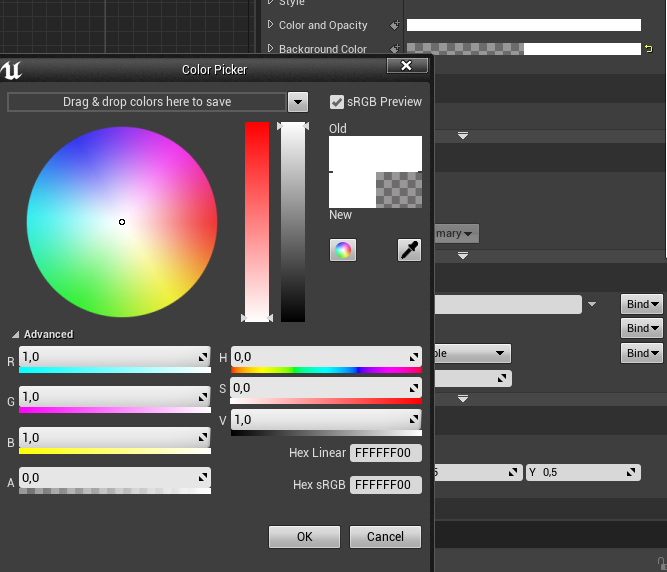
Insérer un bouton et un Text Block



Sélectionner Desired on Screen



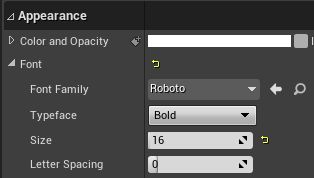
Sélectionner le bouton



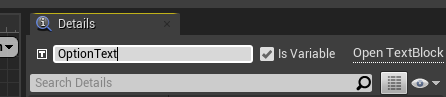
Passer l’Alpha à 0 du Background Color



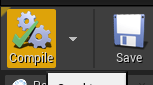
Sélectionner TextBlock



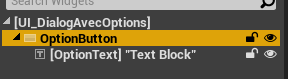
Passer la taille de la font à 16 par exemple



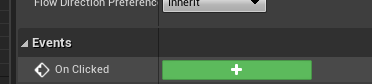
Remplacer le nom du bouton par un nom de variable et cocher la case Is Variable



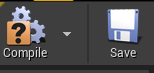
Sauver/Compiler



Renommer le bouton en OptionButton

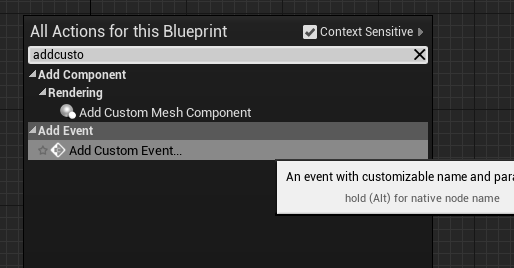


Ajouter l’événement OnClicked



Sauver/Compiler

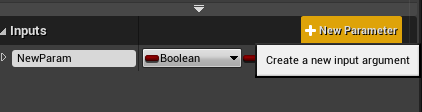
On va maintenant retourner sur l’Event Graph du Player afin de gérer les touches du clavier.



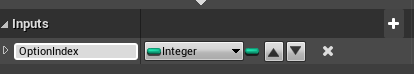
Créer un Custom Event



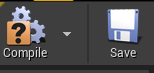
On l’appellera SelectDialogOption



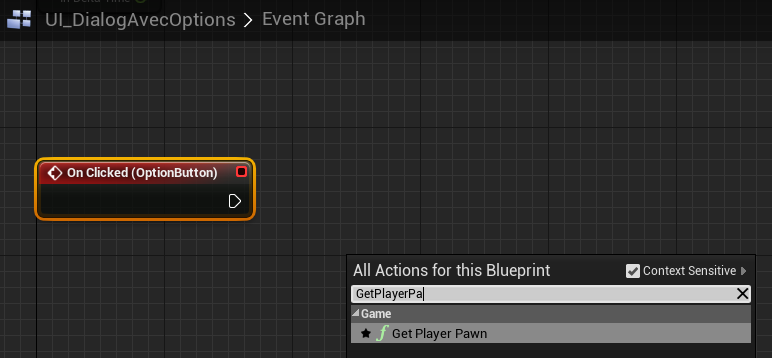
Sur la partie Details, on va ajouter un paramètre



De type entier que l’on appellera OptionIndex



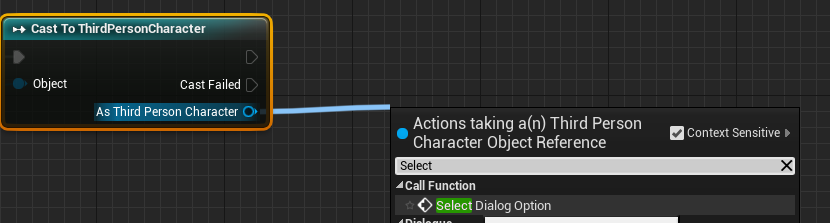
Sauver/Compiler



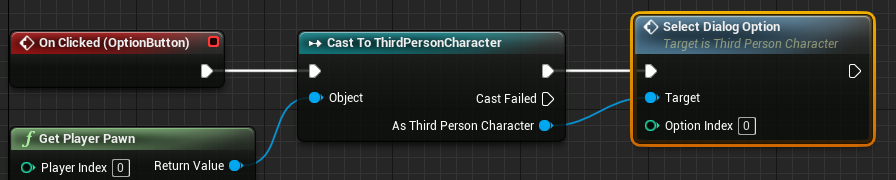
Dans UI\_DialogAvecOptions on va rajouter l’appel à GetPlayerPawn



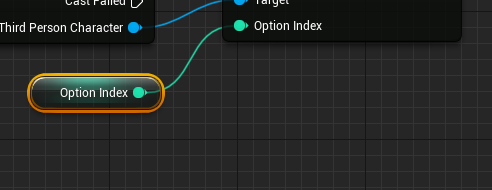
Auquel on va chercher le Cast to ThirdPersonCharacter



Et on appel la fonction SelectDialogOption du personnage



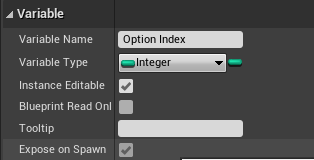
Maintenant il faut définir l’index (Option Index)



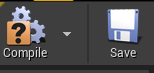
On Promote to Variable Option Index



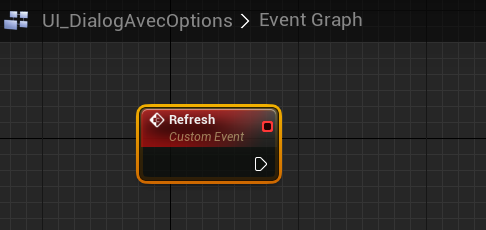
Et on expose la variable (le petit œil à droite).



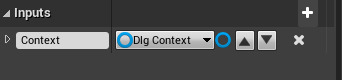
Expose as Spawn pour pouvoir régler la valeur grâce à un pin.



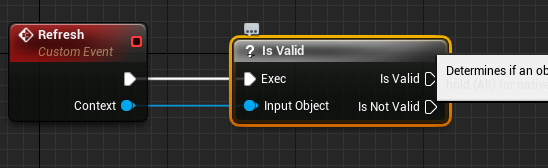
Sauver/Compiler



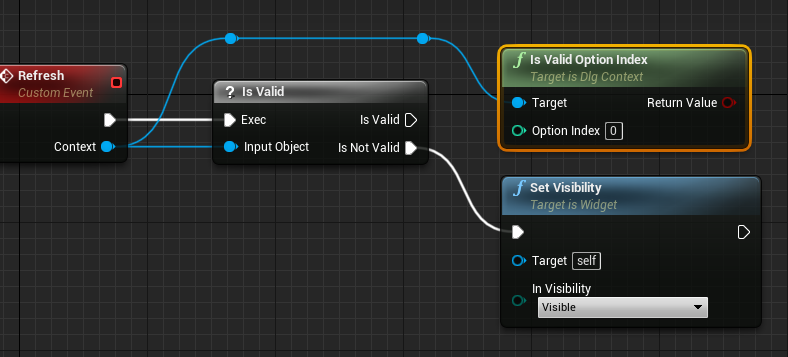
Ajouter un CustomEvent que l’on nommera Refresh



On ajoute une variable d’entrée de type Object Reference sur DlgContext que l’on nommera Context

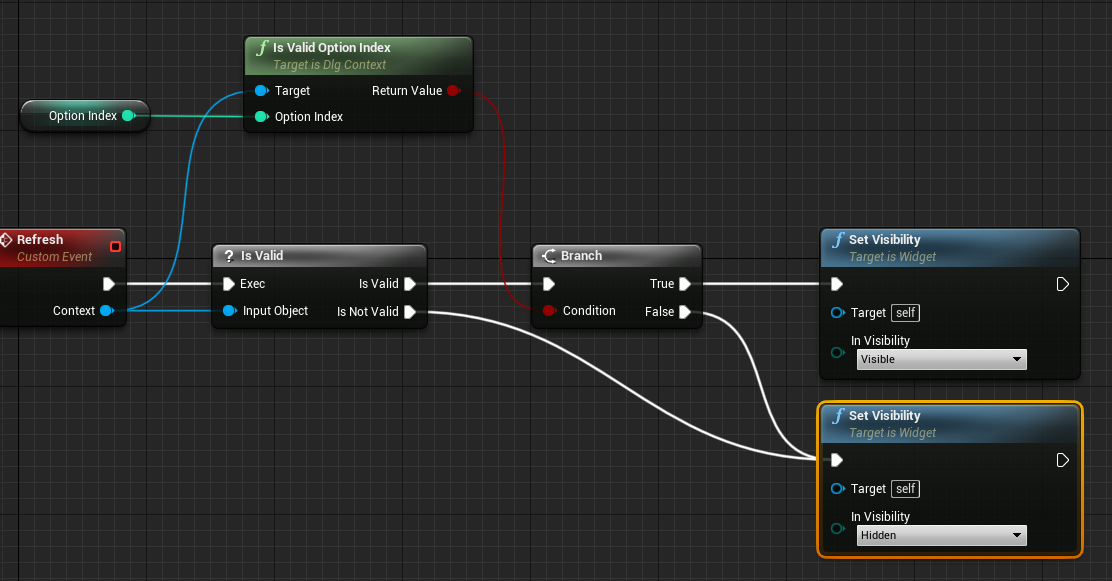


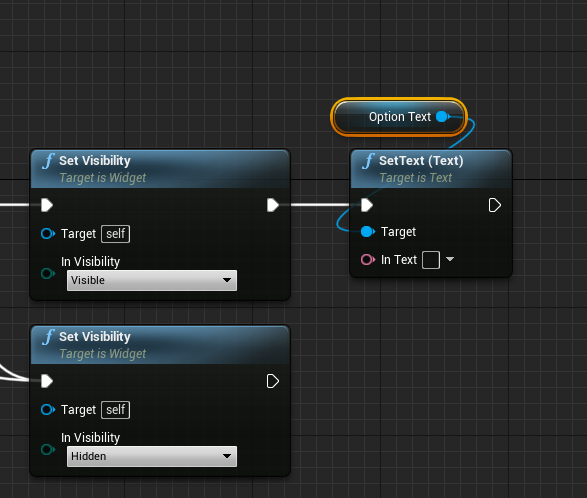
Ajouter une condition de validité de l’objet depuis Context.

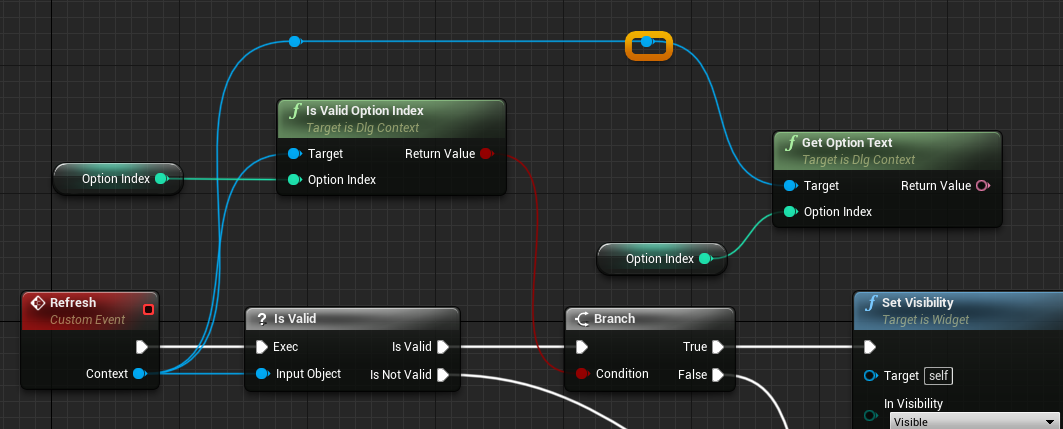


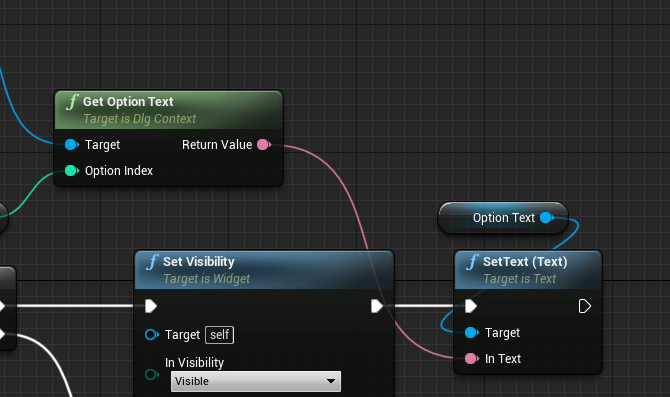
Ajouter un appel à SetVisibility dans le cas où l’objet n’est pas valid

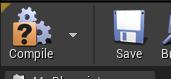
Ajouter un test de validité sur l’index



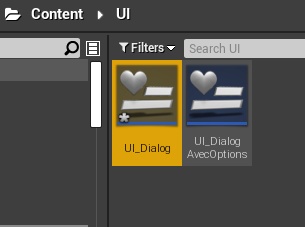




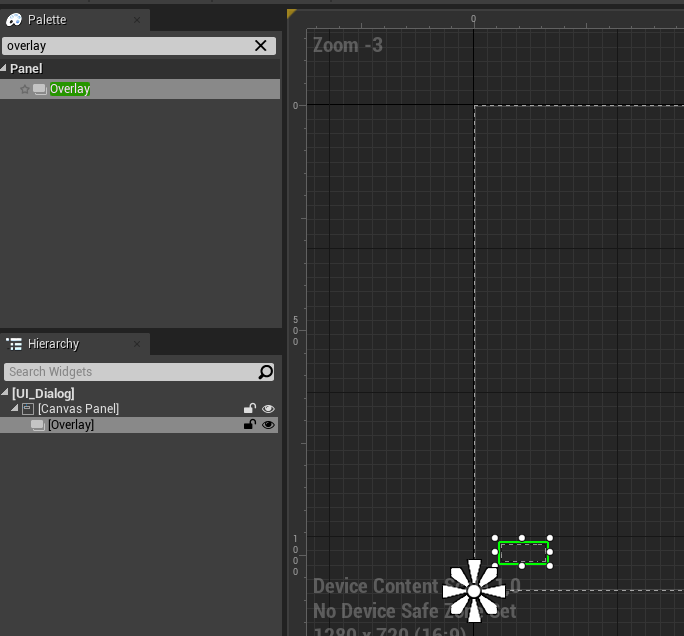




Sauver/Compiler



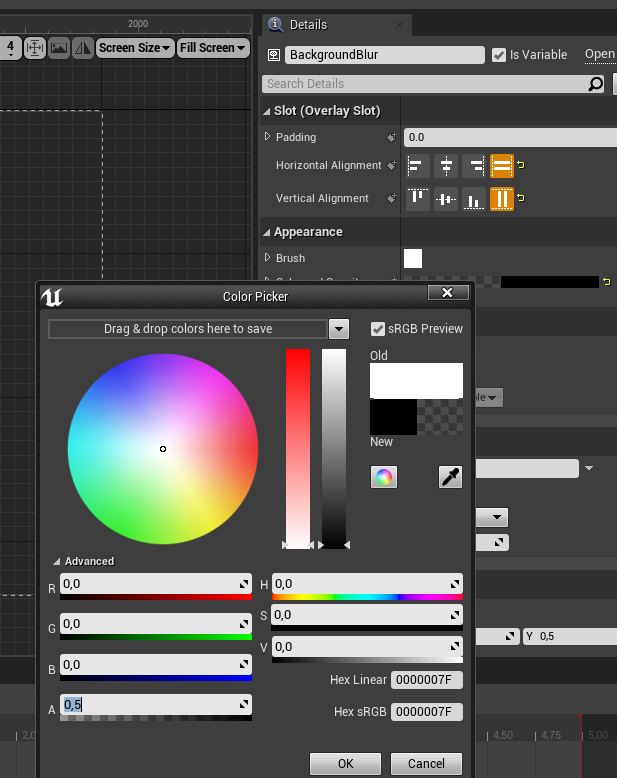
Créer un autre Widget UI\_Dialog



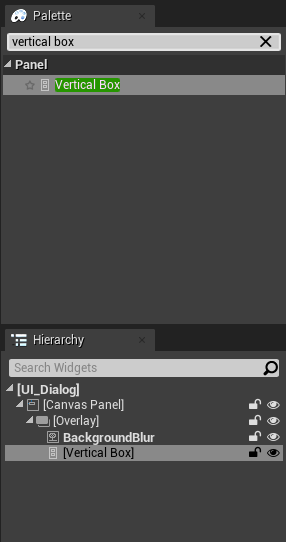
Ajouter un overlay avec Anchor en bas à gauche



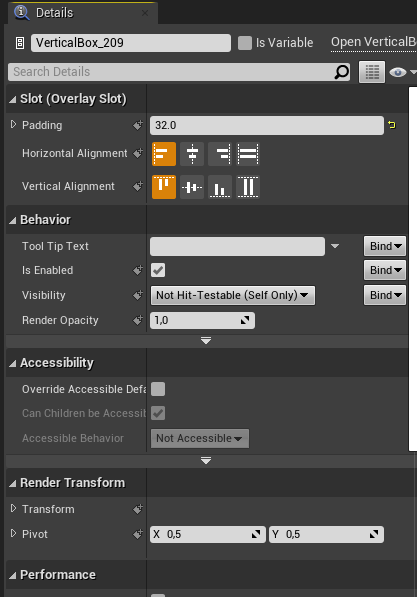
Ajouter une image de fond



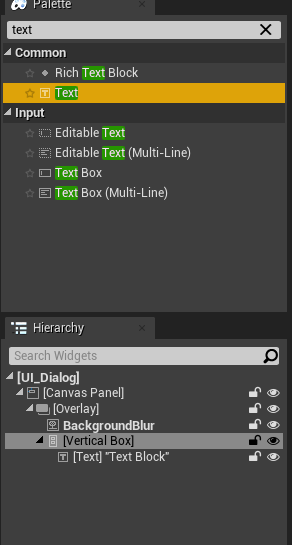
Les paramètres du background



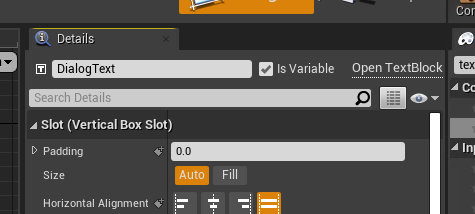
Ajouter une vertical box



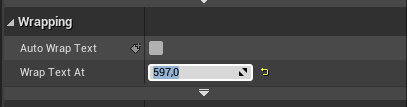
Padding 32



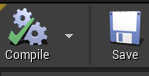
Ajouter une text box à la vertical box



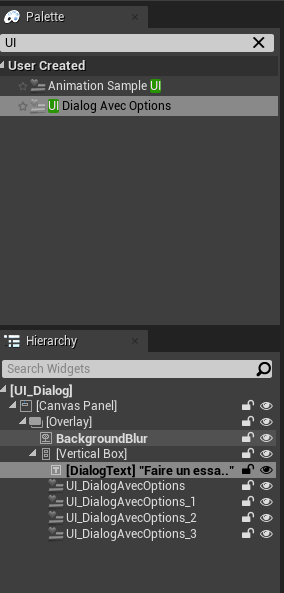
Que l’on nommera DialogText en cochant la case variable



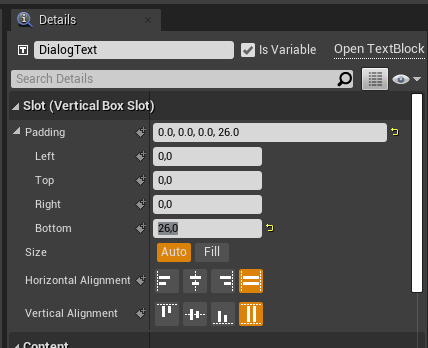
Penser à wrapper le texte, faire des essais…



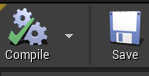
Sauver/Compiler



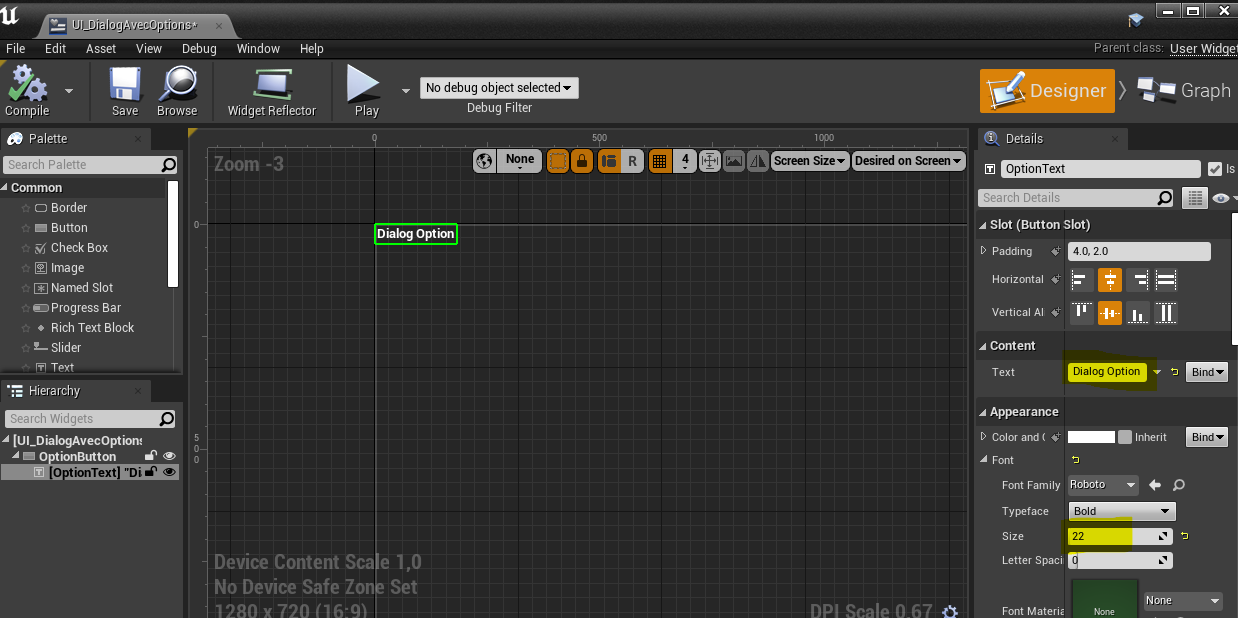
Créer 4 UI\_DialogAvecOptions



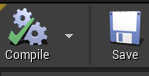
Définir un padding pour les textes



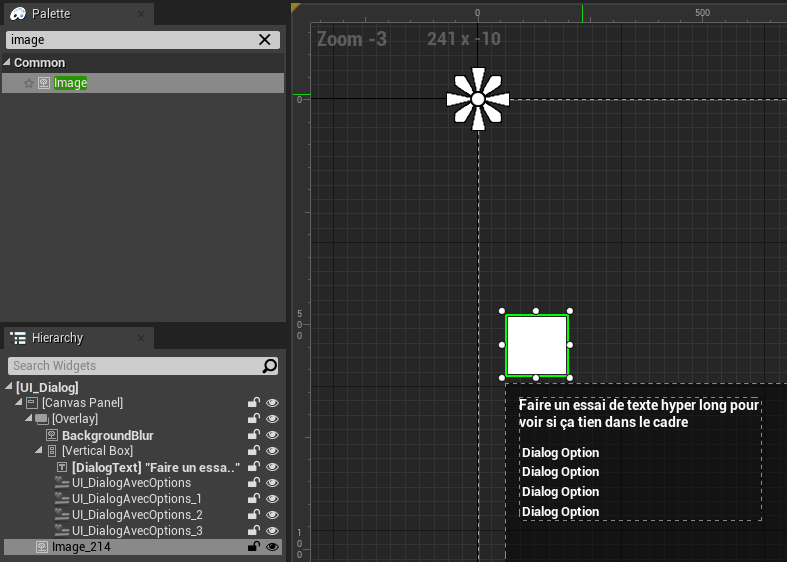
Sauver/Compiler



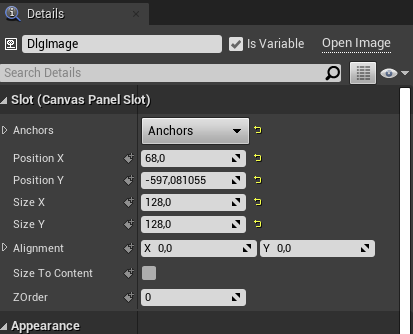
Dans le module Designer de UI\_DialogAvecOptions, remplace le texte par Dialog Option et mettre une taille de caractère 22 par exemple



Sauver/Compiler



Dans UI\_Dialog, on va rajouter une image qui servira à mettre l’icône du personnage qui parle.



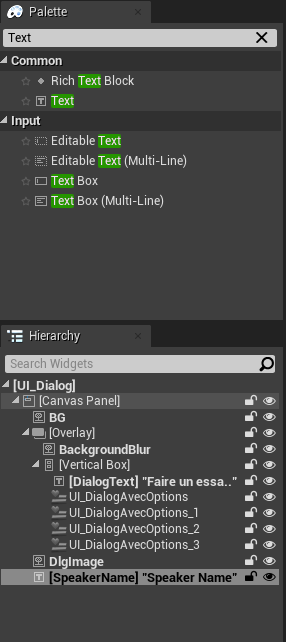
Nommer DlgImage en tant que variable, Anchor en bas à gauche taille des icônes 128x128 pixels.



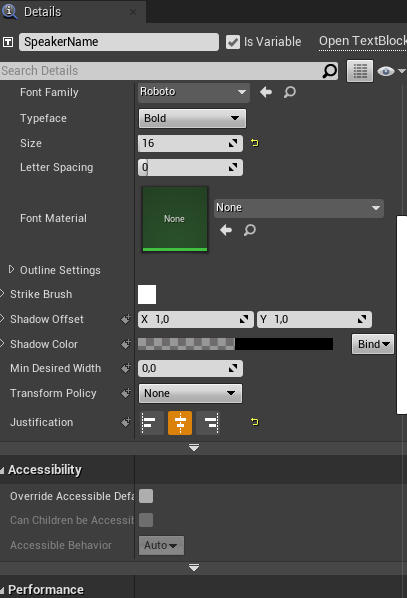
Copier Coller DglImage et renommer en BG



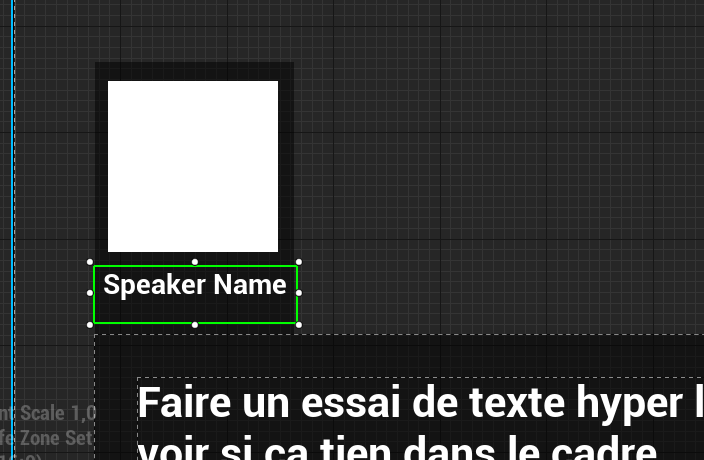
Encadrer l’icône



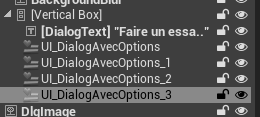
Ajouter un Text Box, la nommer Speaker Name



La définir en tant que variable

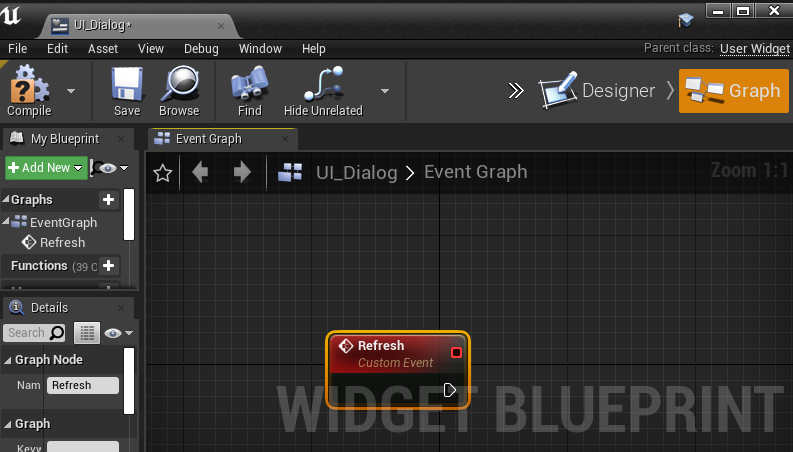


Devrait donner un truc dans le genre…

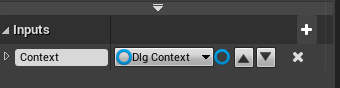




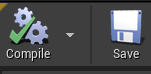
Pour chacune des options, modifier l’index en fonction.



Revenir dans l’Event Graph du Widget et ajouter un custom Event

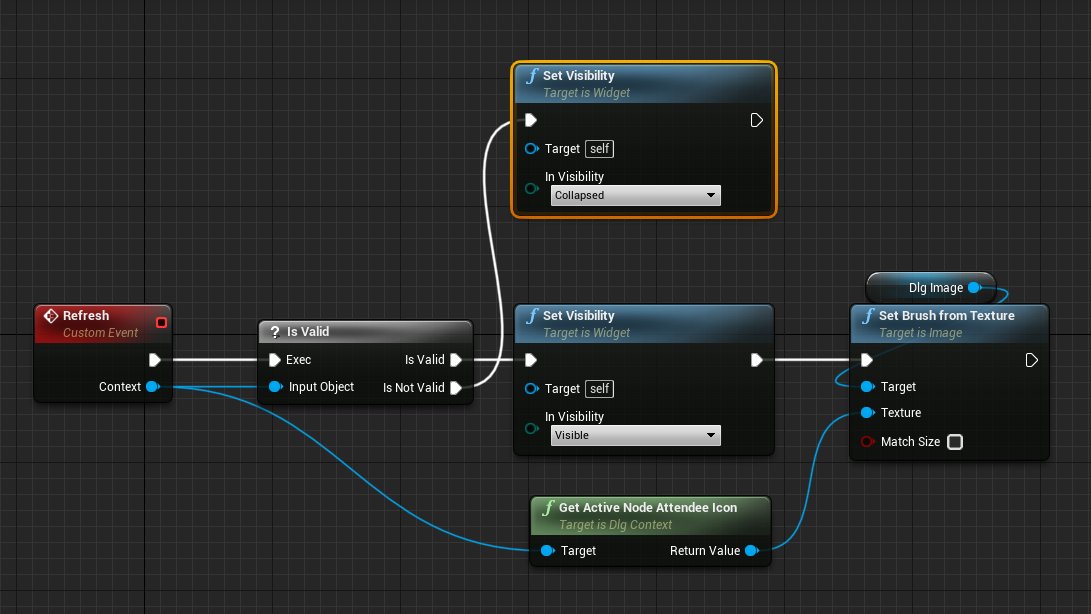


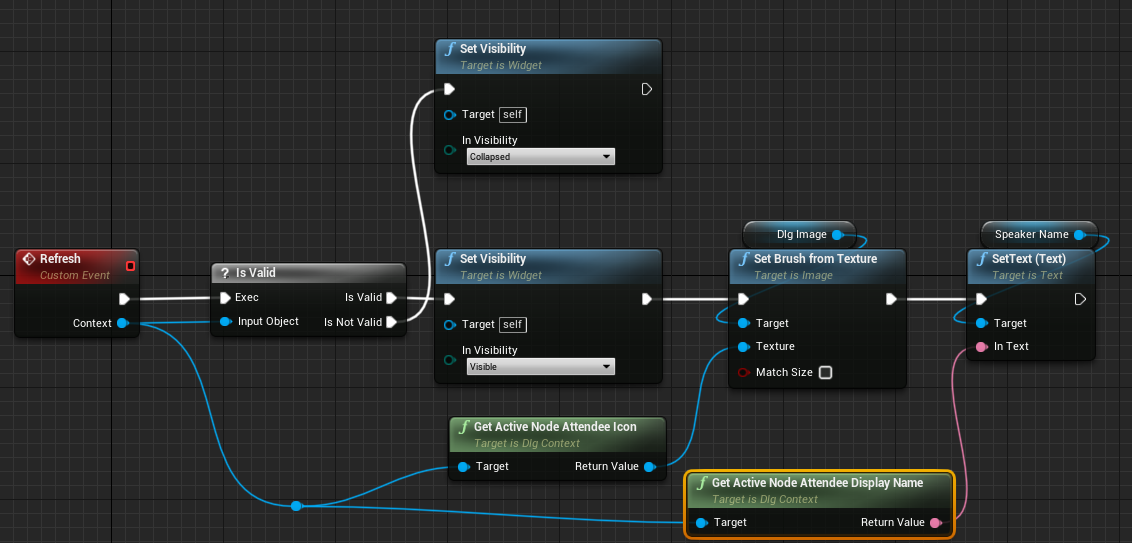
Ajouter une variable d’entrée Context de type Object Reference sur DlgContext

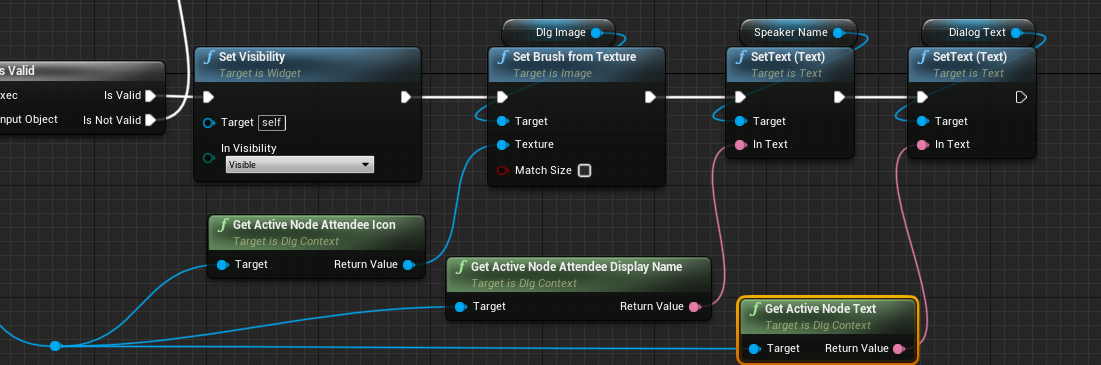


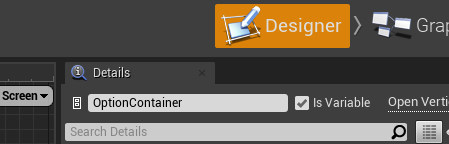
Sauver/compiler



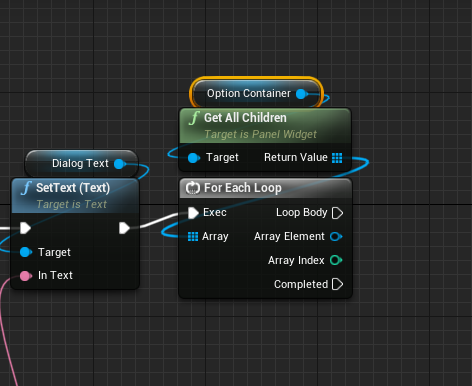


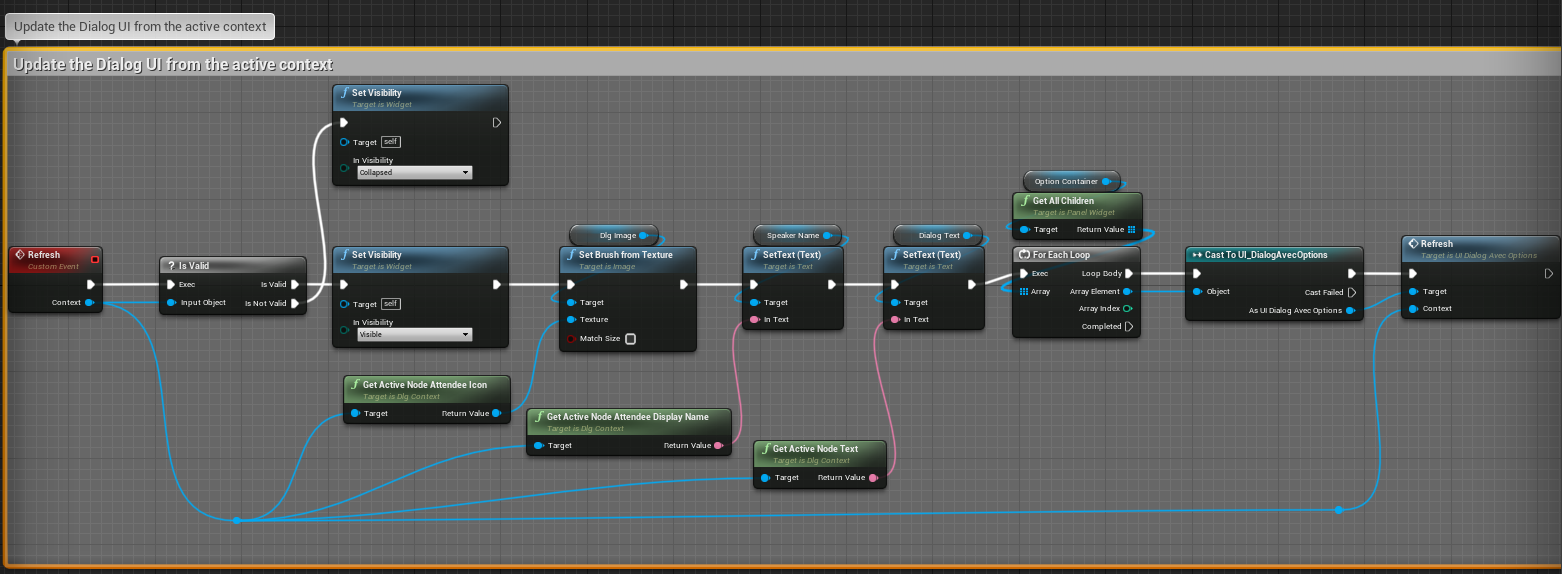






Dans le Designer, wrapper les variables UI\_DialogOption, 1, 2, 3 dans une VerticalBox, et déclarer celle-ci en tant que variable OptionContainer.

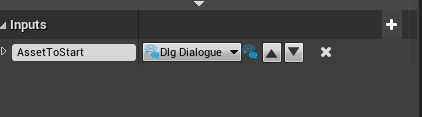




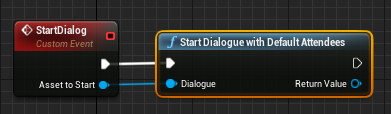
1. Contrôle des dialogues



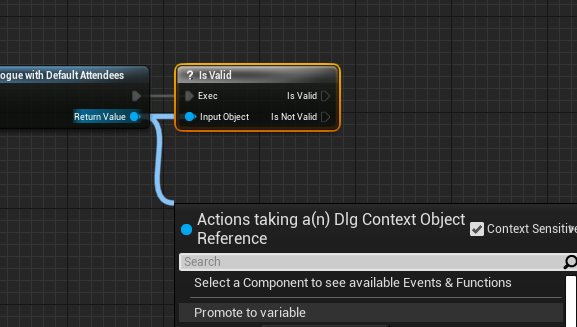
Dans le Blueprint du ThirdPersonCharater, ajouter un custom event StartDialog



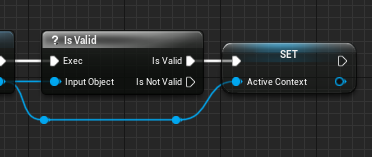
Ajouter une variable d’entrée de type DlgDialogue que l’on nommera AssetToStart



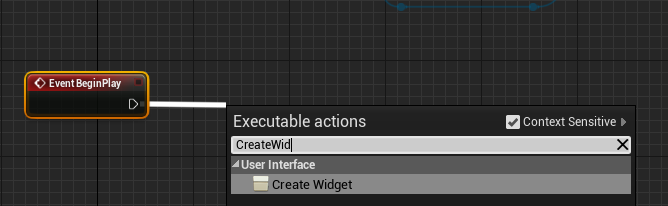
Appeler la fonction de démarrage de dialogues



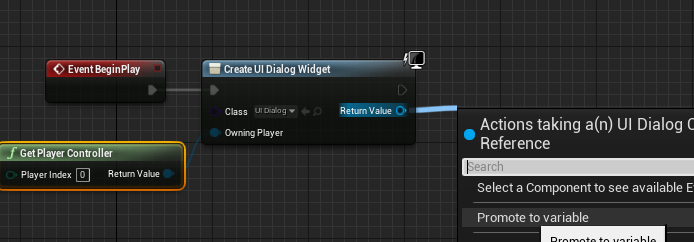
Promouvoir en variable la valeur retournée afin de pouvoir s’en servir.



Que l’on nommera ActiveContext.



Ajouter l’événement BeginPlay qui va permettre de créer le Widget

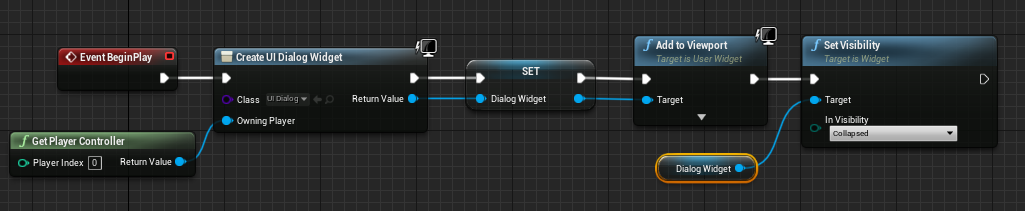


Sélectionner la Class UI\_Dialog et Owning Player proviendra de GetPlayerController.

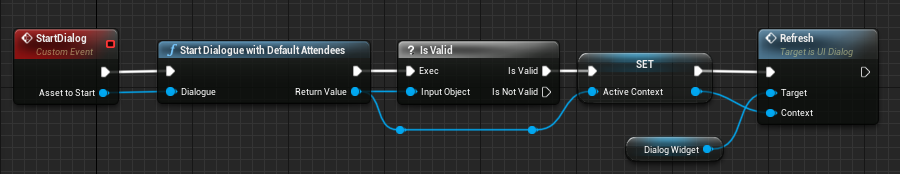
Promouvoir la valeur de retour en variable.



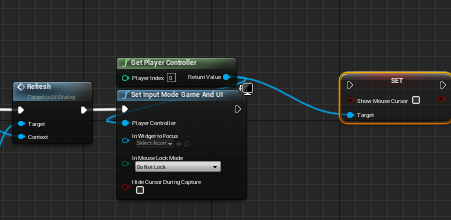
On nommera la variable DialogWidget puis on appel la fonction d’ajout au viewport



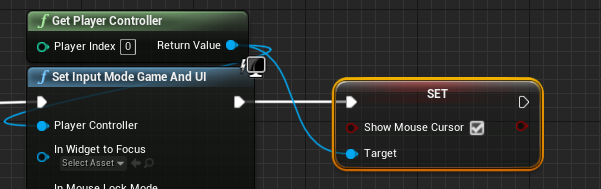
Initialisation du Widget par BeginPlay

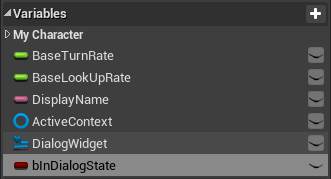


On peut continuer StartDialog en utilisant la variable DialogWidget qui est maintenant initialisée afin d’effectuer un Refresh pour afficher l’état du dialogue.

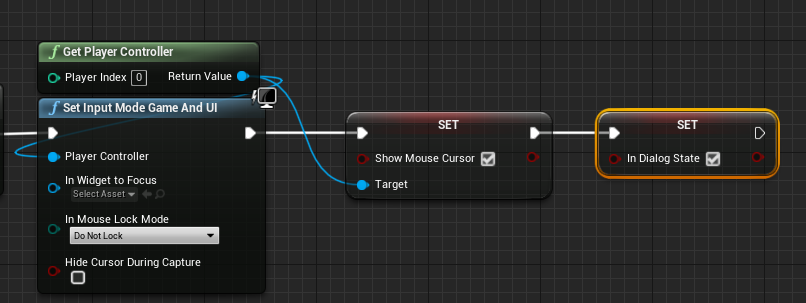


Définir le mode d’entrée Game et UI et afficher le curseur de la sourie dans ce cas.

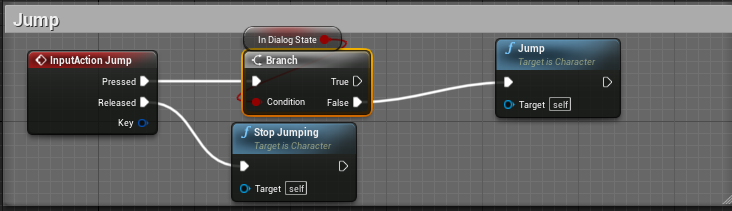




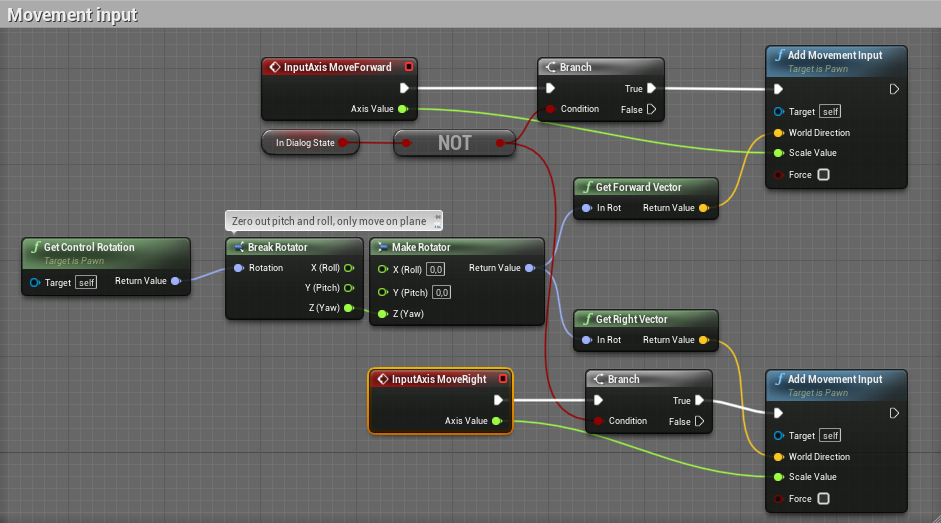
Ajouter une variable booléenne InDialogState afin de conserver l’état dialogue ou non.



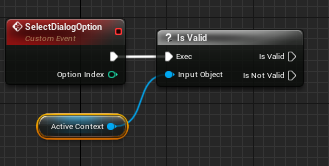
Passer la variable InDialogState à vrai.



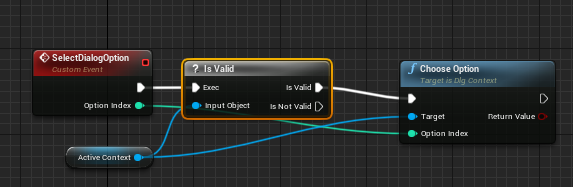
On va interdire de sauter pendant un dialogue.



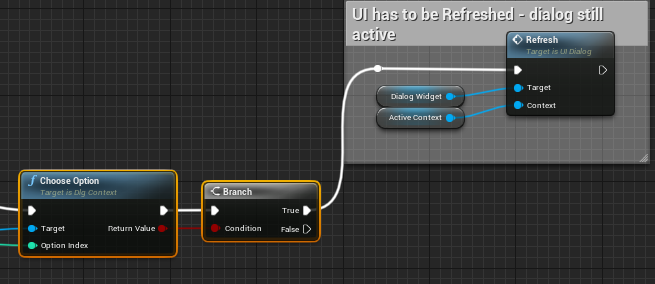
Idem pour les mouvements du personnage (rotation, etc..)



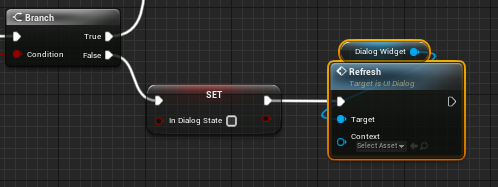
Passons à SelectDialogOption en vérifiant d’abord si le contexte est valide.



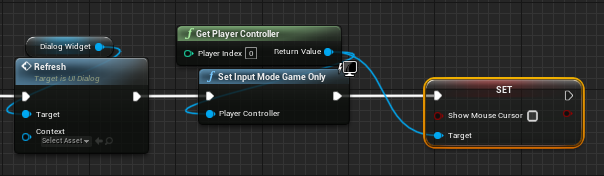
On choisi alors l’option en fonction du contexte



On doit rafraichir l’UI car le dialogue est encore actif.

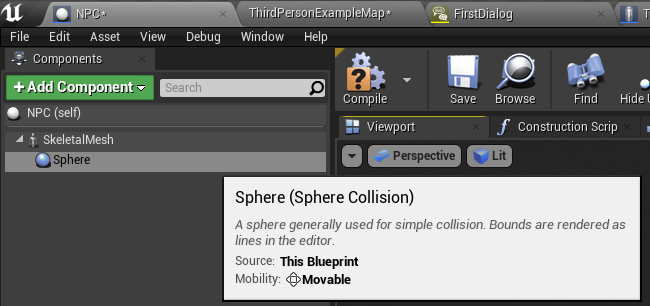


Sinon, on rétablie la variable booléenne et on refresh l’UI.

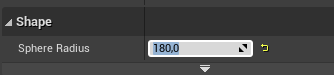


On rétabli l’input mode en mode jeu seulement et on désactive la souris.

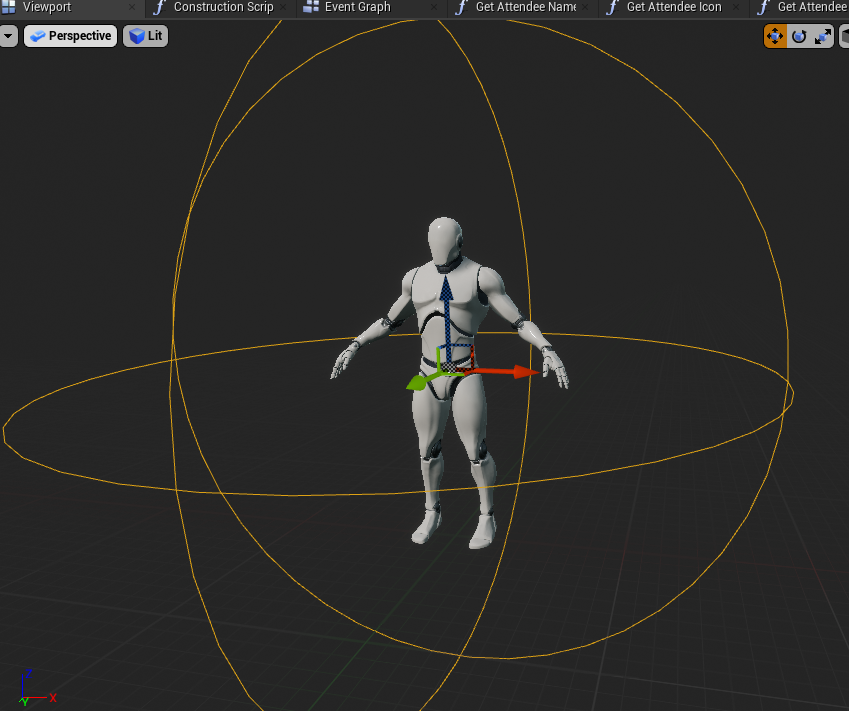
1. Déclenchement du dialogue avec le NPC



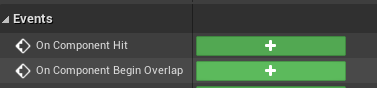
Ajouter un Component Sphere Collision



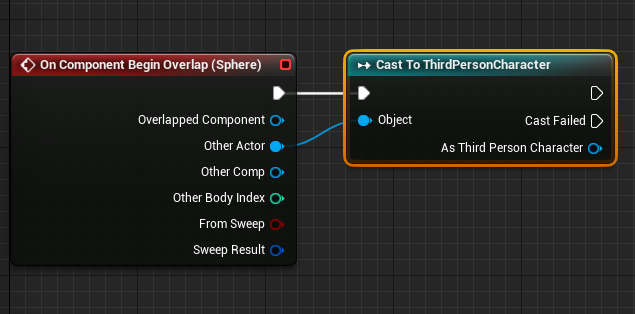
Définir son radius à 180



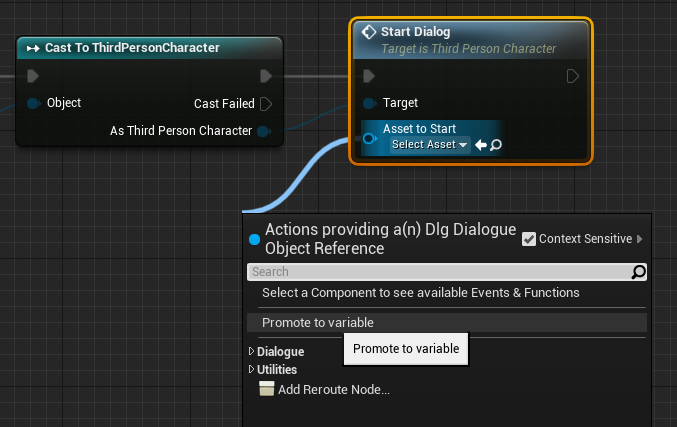
Et placer la sphère au centre du personnage



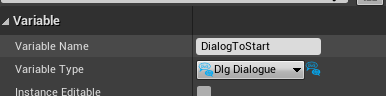
Ajouter la fonction OnComponentBeginOverlap qui va servir à déclencher le dialogue.



Déclencher le dialogue dans le cas où c’est le personnage joueur qui passe sur lui.

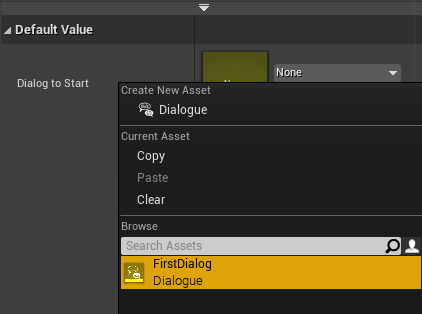


Promouvoir en variable



La nommer DialogToStart



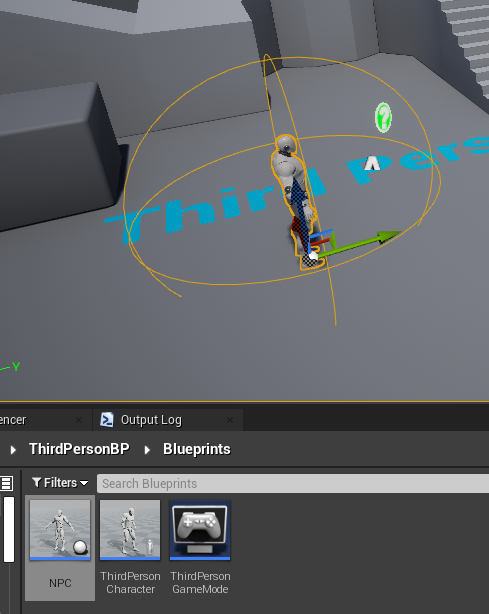


Sélectionner le dialogue FirstDialog

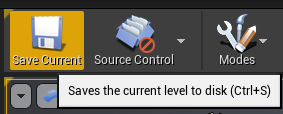


Compiler/Sauver

1. Test :



Ajouter le NPC dans la scene



Sauver et lancer par la touche Play.