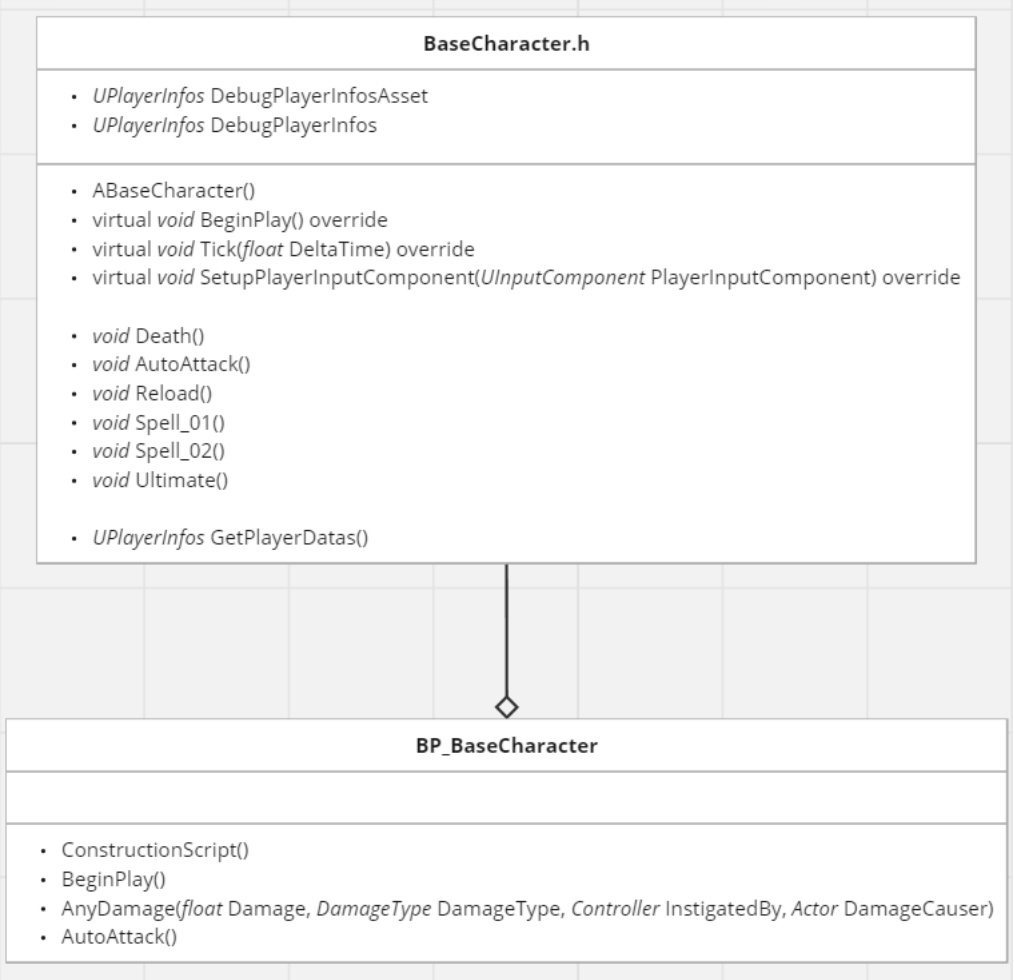
Documentation Workflow MoBArtFX

M4-GD/Prog 2023-2024

Table des matières

Character Hierarchy



BaseCharacter.h

BaseCharacter est le niveau le plus bas de chaque personnage intégré à ce projet. Il doit être utilisé obligatoirement.

Il contient différentes fonctions héritables :

Death()	Fonction pré-implémenté à faire hériter dans l'objectif de prendre à charge la mort du personnage
AutoAttack() Spell01() Spell02() Ultimate()	Fonctions pré-implémentées à faire hériter dans l'objectif de prendre en charge les différentes attaques du personnage
Reload()	Recharge les munitions lorsque GetDataPlayer()->CurrentAmmo <= 0
GetPlayerData()	Fonction à utiliser obligatoirement en C++ comme en Blueprint pour récupérer une/des information(s) du personnage, quelle(s) qu'elle(s) soi(en)t. Elle communique avec le PlayerState pour Get/Set les données du joueur.

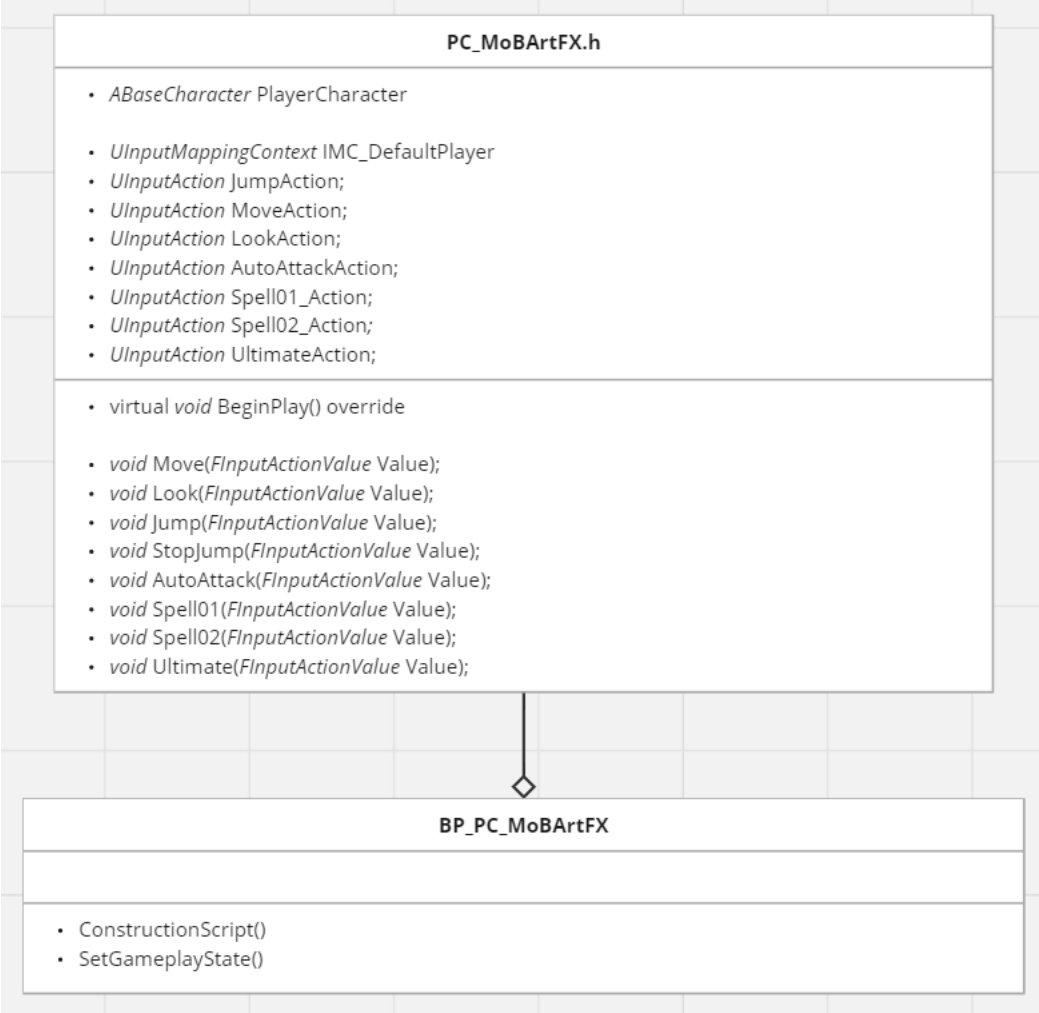
BP_BaseCharacter

BP_BaseCharacter est la base de votre personnage en Blueprint. Il peut être utilisé dans votre hiérarchie, mais n'est en aucun cas obligatoire. Il est cependant vital aux différentes fonctions de debugs prévues par le Workflow.

Les informations importantes sont les suivantes :

DebugPlayerInfosAsset	SoftReference disponible dans les **ClassDefaults** permettant de renseigner un PrimaryDataAsset *(hérité de UPlayerInfos)*, comprenant des PlayerInfos de Debug utiles aux différentes fonctionnalités de test implémentées.
BeginPlay()	Instancie le DataAsset de Debug, si DebugPlayerInfosAsset n'est pas null
AnyDamage()	Fonction de Gestion basique de perte des PVs lorsque le personnage est attaqué. Elle peut être override dans l'enfant.
AutoAttack()	Décrémente les munitions à la vitesse de GetPlayerData()->AutoAttack_CD

PlayerController Hierarchy



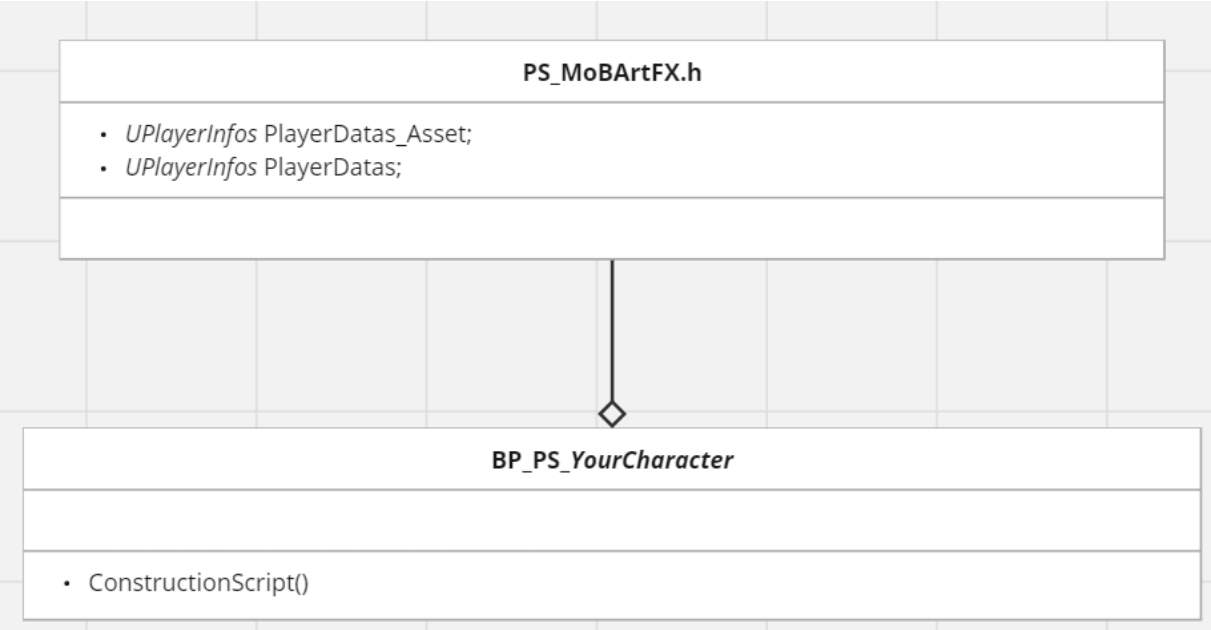
PC_MoBArtFX.h

Il est le cœur du contrôle de votre personnage. Son rôle est d'appeler les différentes Fonctions présentes dans BaseCharacter lors de la détection des Inputs.

BP_PC_MoBArtFX

Il est utilisé principalement pour du Debug. Il a cependant, en plus de son parent, une Interface permettant de changer son GameState. Ainsi, il est facile de changer l'état du personnage visé pour le passer en stun/root/etc.

PlayerState Hierarchy



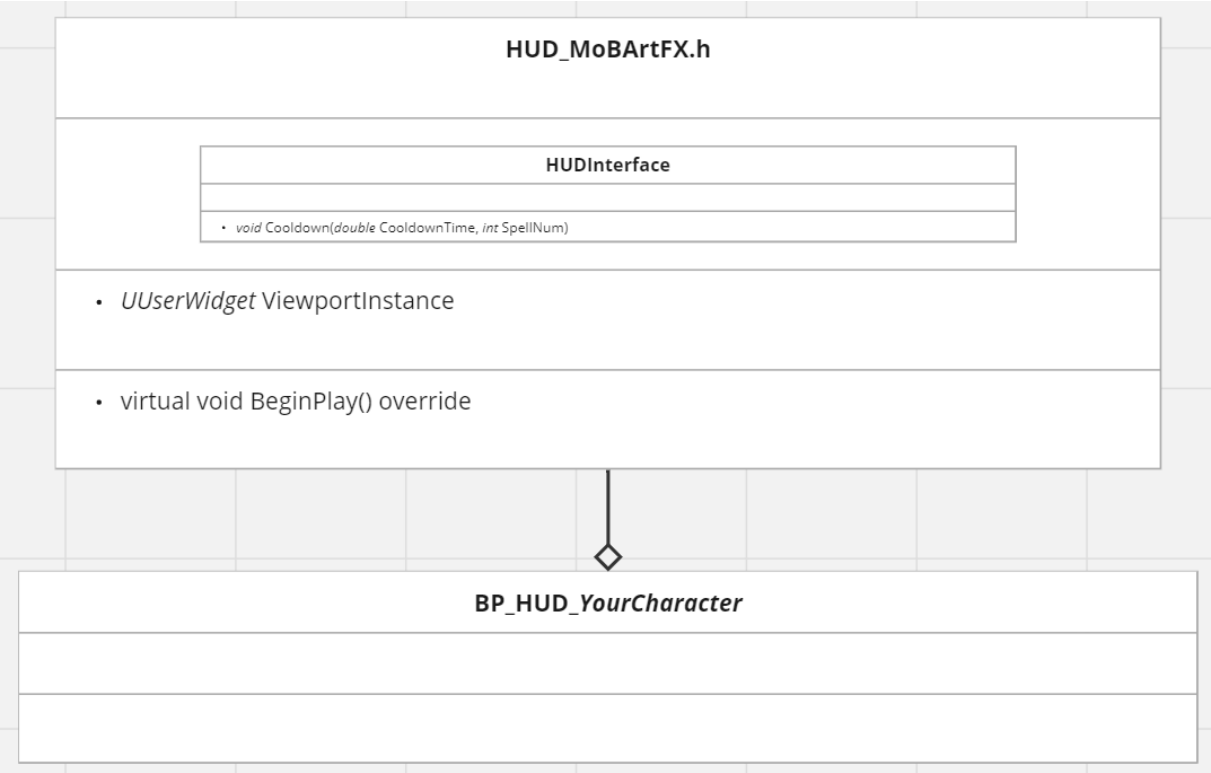
PS_MoBArtFX.h

Il est utilisé dans le simple objectif de venir charger le DataAsset du personnage selectionné

BP_PS_*YourCharacter*

Implémenter un enfant à PS_MoBArtFX.h permet de venir modifier à la main le DataAsset utilisé par le personnage joué. Cependant, cela doit uniquement être utilisé en cas de Debug. En jeu, il faut venir modifier dynamiquement le DataAsset directement de PS_MoBArtFX lors de la sélection du personnage.

HUD Hierarchy



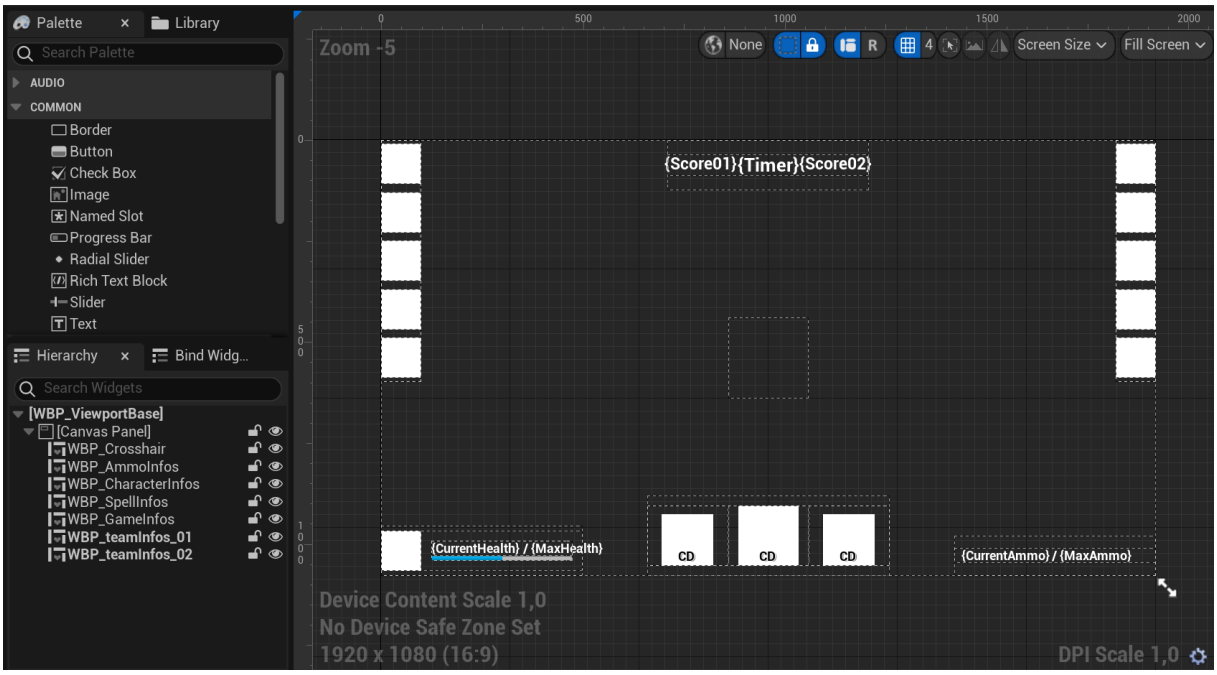
HUD_MoBArtFX.h

Il est utilisé dans l'unique objectif d'ajouter le WidgetBlueprint de Viewport sur le PlayerScreen. Il en garde une référence, et implémente ensuite une Interface permettant de manipuler les éléments du Viewport facilement.

BP_HUD_*YourCharacter*

Il peut être utile de créer un enfant de HUD_MoBArtFX dans l'objectif soit de venir debugger des informations.

Viewport Hierarchy



WBP_ViewportBase

Il implémente tous les éléments de base que doit comprendre le Viewport. On peut séparer ses informations en 2 groupes :

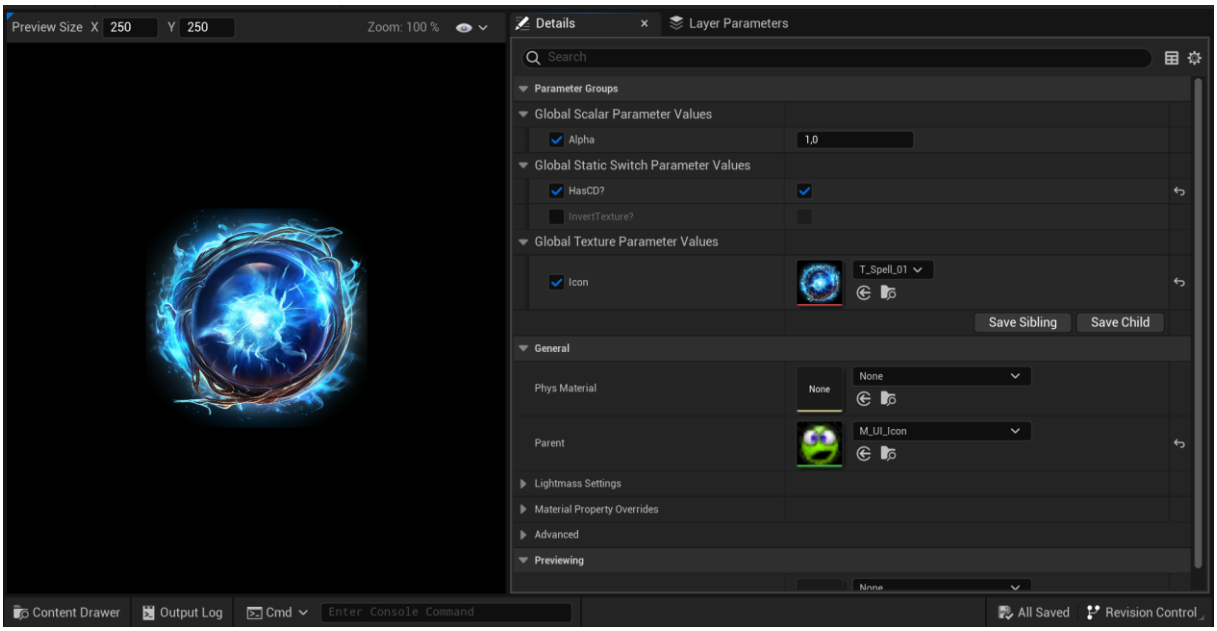
- PlayerDatas : Partie inférieure de l'écran *(Spells, Ammos, Health, etc)*
- GameDatas : Partie supérieure de l'écran *(Equipes, scores, timer, etc)*

Chacun de ses éléments sont Set grâce au PlayerData récupéré dans le PlayerState du personnage selectionné, ainsi que sur le GameData, présent dans le GameState.

WBP_Viewport_YourCharacter

Il peut être utile de créer un enfant de WBP_ViewportBase dans l'objectif soit de venir debugger des informations, soit pour venir ajouter des éléments au Viewport de base, qui peuvent être utile à l'UX de vos personnages.

UI Icons Hierarchy



Le MaterialMaster à utiliser pour afficher les icones de l'UI est **M_UI_Icon**. Il comprend une texture, une inversion des couleurs pour les Masks, ainsi qu'un booléen lui permettant d'être pris en charge par le système de Cooldown, communiqué par l'interface du HUD.