DOCUMENTATION PROJET :

P0rT4L

SOMMAIRE :

I – Manager :

- FPSCount ---------- 1
- Gamemanager ---------- 1

II – Player :

- CameraFollow ---------- 2
- GravityGun ---------- 2
- Move ---------- 2
- OrbitalControl ---------- 4

III – Props :

- Ascensor ---------- 4
- ActionLight ---------- 5
- Button ---------- 5
- ButtonAction ---------- 5
- Cam ---------- 6
- Checkpoint ---------- 6
- DestructorAction ---------- 6
- DisableChamp ---------- 7
- Door ---------- 7
- DoorEnd ---------- 7
- ObjectAction ---------- 7
- Portal ---------- 8
- Projectile ---------- 8
- RecepteurAction ---------- 9
- ShooterEnergyBall ---------- 9
- TPWall ---------- 9

I – Manager :

- **FPSCount :**

Permet au jeu de gérer le calcul des FPS.

CalculateFPS()

Fonction permettant le calcul du nombre d'FPS actuel.

- **GameManager :**

Permet la gestion du jeu ainsi que de l'affichage.

Il existe une instance du GameManager qui peut être utiliser avec : Gamemanager.instance

OnPause(InputAction.CallbackContext context)

Fonction qui permet de gérer quand le joueur appui sur la touche de pause (**Echap**).

Pause()

Fonction qui gère la mise en pause du jeu ainsi que de l'affichage du menu de pause.

End()

Fonction appelée quand le joueur fini le niveau, il permet l'affichage du menu de fin.

LoadTheGame(int index)

Fonction permettant de charger la scène liée à l'index passé en paramètre.

ChangeSlide(bool isBlue, float _value)

Fonction permettant de changer la valeur des sliders **(_value**) qui correspond au portail (**isBlue**).

II – Player :

- **CameraFollow :**

Permet à la caméra de se fix au joueur.

- **GravityGun :**

Permet la gestion des fonctionnalités du saisi d'objet fait par le PortalGun ainsi que l'interaction avec l'environnement.

OnGravityGun(InputAction.CallbackContext context)

Fonction appelée quand le joueur presse la touche de saisi d'objet (**E**), elle permet de détecter par rapport à une distance définie si le joueur peut saisir ou non l'objet ou d'interagir avec un élément.

- **Move :**

Permet de gérer toutes les fonctionnalités liées au joueur.

SetWaypoint(Transform _waypoint)

Fonction permettant de changer les coordonnées **(_waypoint**) du waypoint du joueur.

OnMove(InputAction.CallbackContext context)

Fonction appelée quand le joueur presse les touches de déplacement (**ZQSD**), elle permet de récupérer la direction du mouvement saisi.

OnJump(InputAction.CallbackContext context)

Fonction appelée quand le joueur presse la touche de saut (**Espace**), elle permet d'utiliser la fonction de Saut si le joueur est au sol.

OnShootOrange(InputAction.CallbackContext context)

Fonction appelée quand le joueur presse la touche du portail orange (**click droit**), elle permet l'utilisation de la fonction de tire de portail en lui spécifiant de qu'elle portail il s'agit.

OnShootBlue(InputAction.CallbackContext context)

Fonction appelée quand le joueur presse la touche du portail bleu (**click gauche**), elle permet l'utilisation de la fonction de tire de portail en lui spécifiant de qu'elle portail il s'agit.

isGrouded()

Fonction renvoyant un booléen pour savoir si l'on touche bien le sol.

Lauch(bool isOrangeSide)

Fonction permettant le lancer du projectile permettant la création du portail en fonction du qu'elle portail on veut créer (**isOrangeSide**).

LimiteSpeed()

Fonction permettant la limitation de la vitesse du joueur.

Jump()

Fonction permettant le saut du joueur.

ResetJump()

Fonction permettant de réactiver le jump.

Death()

Fonction gérant la mort du joueur en le téléportant au dernier waypoint passé.

- **OrbitalControl :**

Permet la gestion de la caméra orbitale du joueur.

OnMouseY(InputAction.CallbackContext context

Fonction appelée quand le joueur déplace **la souris sur l'axe X**, elle permet de changer la rotation de la caméra sur l'axe X.

OnMouseX(InputAction.CallbackContext contextZ

Fonction appelée quand le joueur déplace **la souris sur l'axe Y**, elle permet de changer la rotation de la caméra sur l'axe Y.

DisableControl(bool _isDisable)

Fonction permettant de désactiver les contrôles de la caméra en fonction d'un booléen (_**isDisable**).

III – Props :

- **Ascensor**

Permet la gestion des animations de l'ascenseur.

Elle dérive de la classe **ObjectAction**.

Action()

Fonction permettant de faire monter l'ascenseur.

ActionStop()

Fonction permettant de faire descendre l'ascenseur.

- **ActionLight**

Permet la gestion des lights lié au récepteur

Elle dérive de la classe **ObjectAction**.

Action()

Fonction permettant e changer la couleur des light.

- **Button**

Permet de gérer les boutons au sol

OnTriggerEnter(Collider other)

Fonction permettant d'activer l'objet lié à ce bouton si le joueur ou le companion cube passe sur le bouton.

OnTriggerExit(Collider other)

Fonction permettant de désactiver l'objet lié à ce bouton a la suite d'un temps d'attente, si le joueur ou le companion cube passe sorte du bouton.

Reset()

Fonction permettant le retour à l'état de base de l'objet lié.

- **ButtonAction**

Permet de gérer les boutons à activer avec la touche d'interaction.

Action()

Fonction permettant de gérer l'activation de l'objet lié

Reset()

Fonction permettant le retour à l'état de base de l'objet lié.

- **Cam**

Permet de gérer la caméra de surveillance qui regarde en permanence le joueur.

OnCollisionEnter(Collision collision)

Fonction permettant de faire tomber la caméra si le joueur tire dessus

- **Checkpoint**

Permet de gérer les checkpoints.

OnTriggerEnter(Collider other)

Fonction permettant de déplacer le waypoint lié au joueur quand le joueur passe sur le checkpoint.

- **DestructorAction**

Permet la gestion de l'incinérateur.

Elle dérive de la classe **ObjectAction**.

Action()

Fonction permettant la destruction du couvercle quand le joueur l'active.

OnTriggerEnter(Collider collision)

Fonction permettant l'activation de l'objet lié quand l'on détruit le companion cube à l'aide de l'incinérateur.

- **DisableChamp**

Permet la gestion du champ de désactivation des portails.

OnTriggerEnter(Collider other)

Fonction permettant la désactivation des portails ainsi que du lancé quand le joueur passe dans ce champ.

- **Door**

Permet la gestion des animations de la porte.

Elle dérive de la classe **ObjectAction**.

Action()

Fonction permettant d'ouvrir la porte.

ActionStop()

Fonction permettant de fermer la porte.

- **DoorEnd**

Permet de gérer l'ascenseur de fin.

OnTriggerEnter(Collider other)

Fonction permettant d'activer la fin quand le joueur rentre dans l'ascenseur de fin.

- **ObjectAction**

Interface permettant la création d'objet pouvant être activé ou désactivé.

- **Portal**

Permet de gérer l'utilisation des portails.

OnTriggerEnter(Collider other)

Fonction permettant la téléportation du joueur ou bien d'une boule d'énergie a l'autre portail.

- **Projectile**

Permet de gérer les projectiles servant pour les portails ou bien comme boule d'énergie.

ResetLaunch()

Fonction permettant de reset le lancer du projectile.

Launch(Vector3 direction, float force)

Fonction permettant la tire des projectiles en fonction d'une direction (direction) avec une certaine force (force).

Erase()

Fonction permettant la destruction du projectile.

OnCollisionEnter(Collision collision)

Fonction permettant de gérer la réaction des projectiles de portail avec les murs ou nous pouvons placer des portails mais aussi la collision des boules d'énergie avec le joueur.

- **RecepteurAction**

Permet de gérer le récepteur de boule d'énergie.

OnTriggerEnter(Collider other)

Fonction permettant l'activation de l'objet lié si une boule d'énergie touche le récepteur.

- **ShooterEnergyBall**

Permet de gérer le tireur de boule d'énergie.

Launch()

Fonction permettant la tire de la boule d'énergie.

Light()

Fonction permettant de gérer le changement de couleur des lumières.

- **TPWall**

Permet de gérer le mur ou l'on peut placer un téléporteur.

GetTarget()

Fonction renvoyant la cible à regarder par le portail.

GetSpawn()

Fonction renvoyant la zone ou poser le portail.