

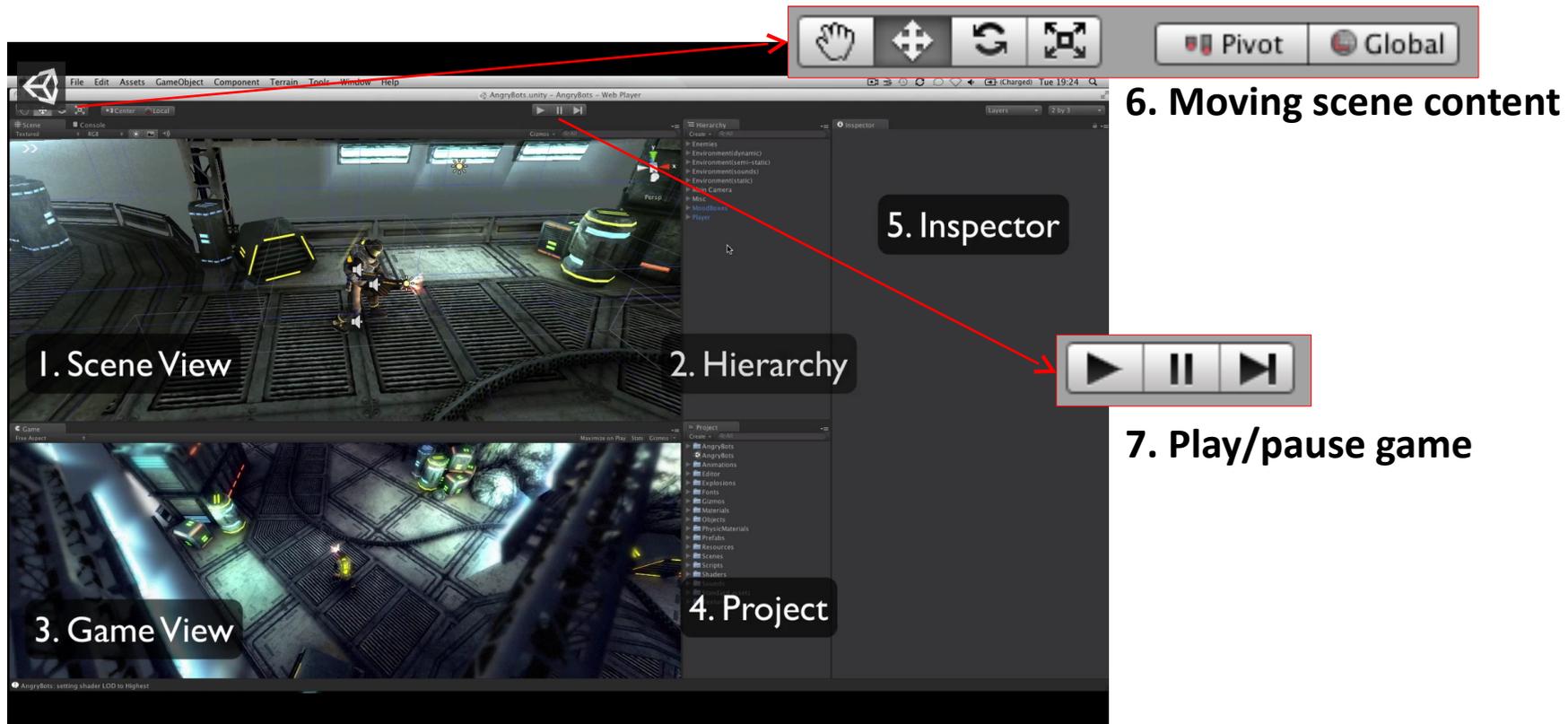
Introduction à Unity 3D

Les bases de la création de scène 3D interactive



La création de contenu 3D à l'aide du moteur de jeu Unity 3D

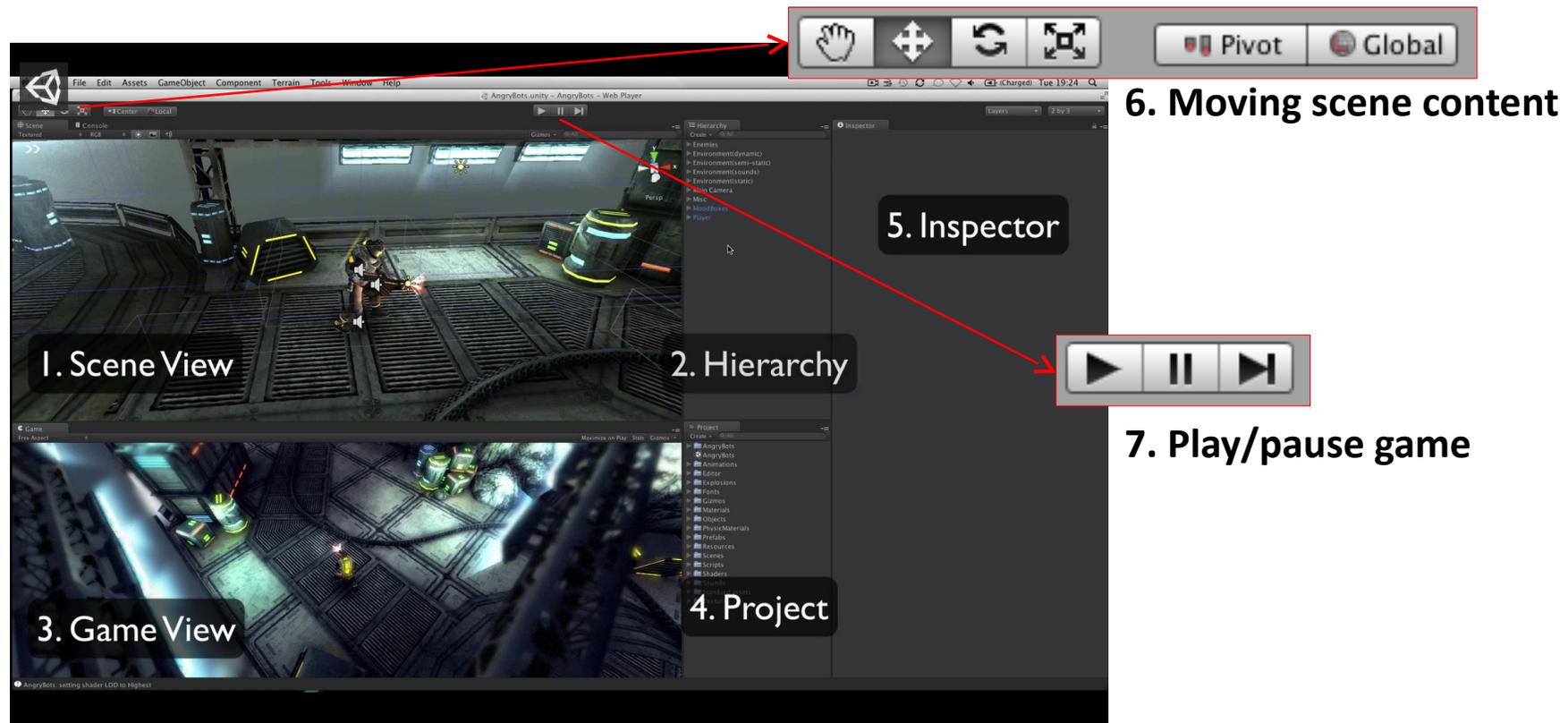
L'interface



1. Scene : vue sur la scène (statique). On peut choisir entre différentes projections orthographiques Front/Right/left/... et des vues en perspective

La création de contenu 3D à l'aide du moteur de jeu Unity 3D

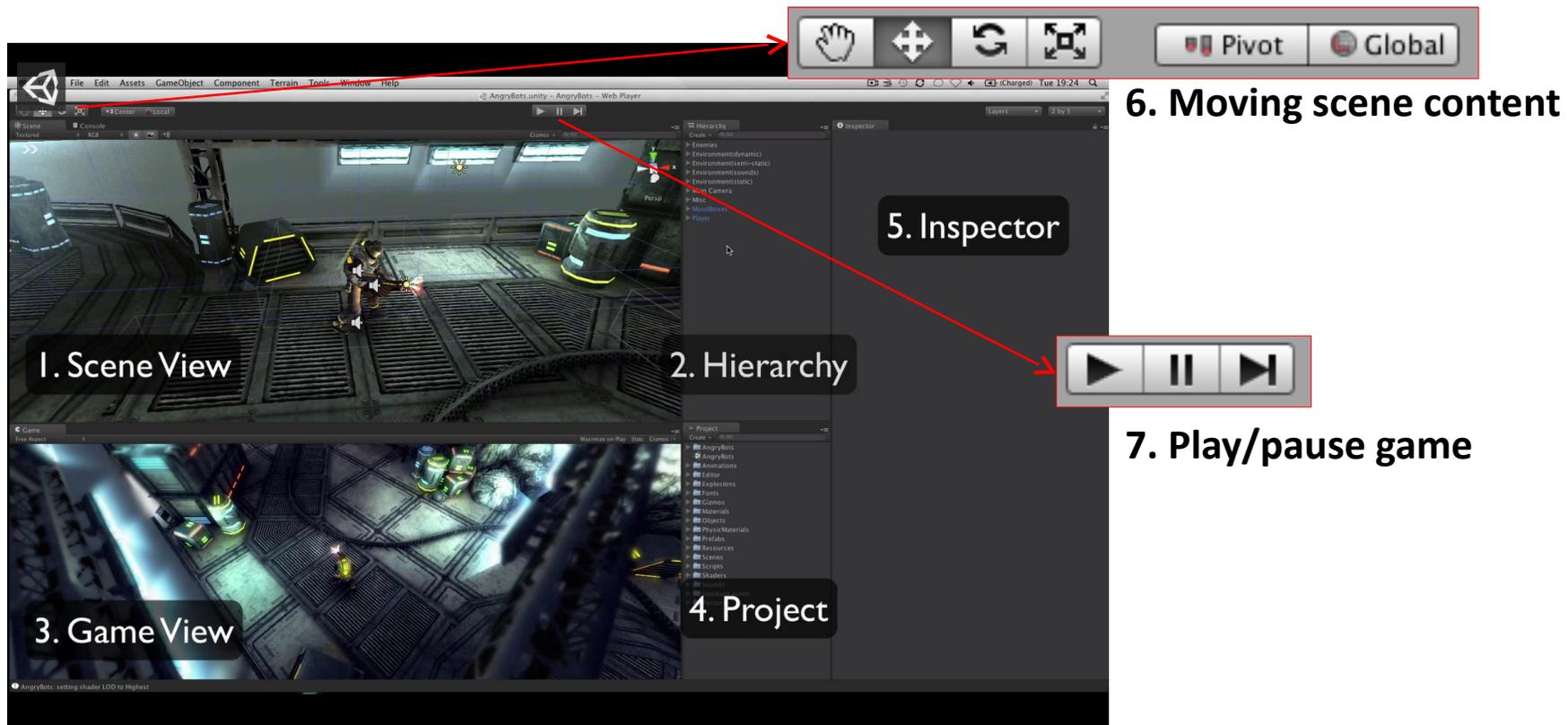
L'interface



2. Game : vue de la caméra sur la scène (dynamique) en situation de jeu.

La création de contenu 3D à l'aide du moteur de jeu Unity 3D

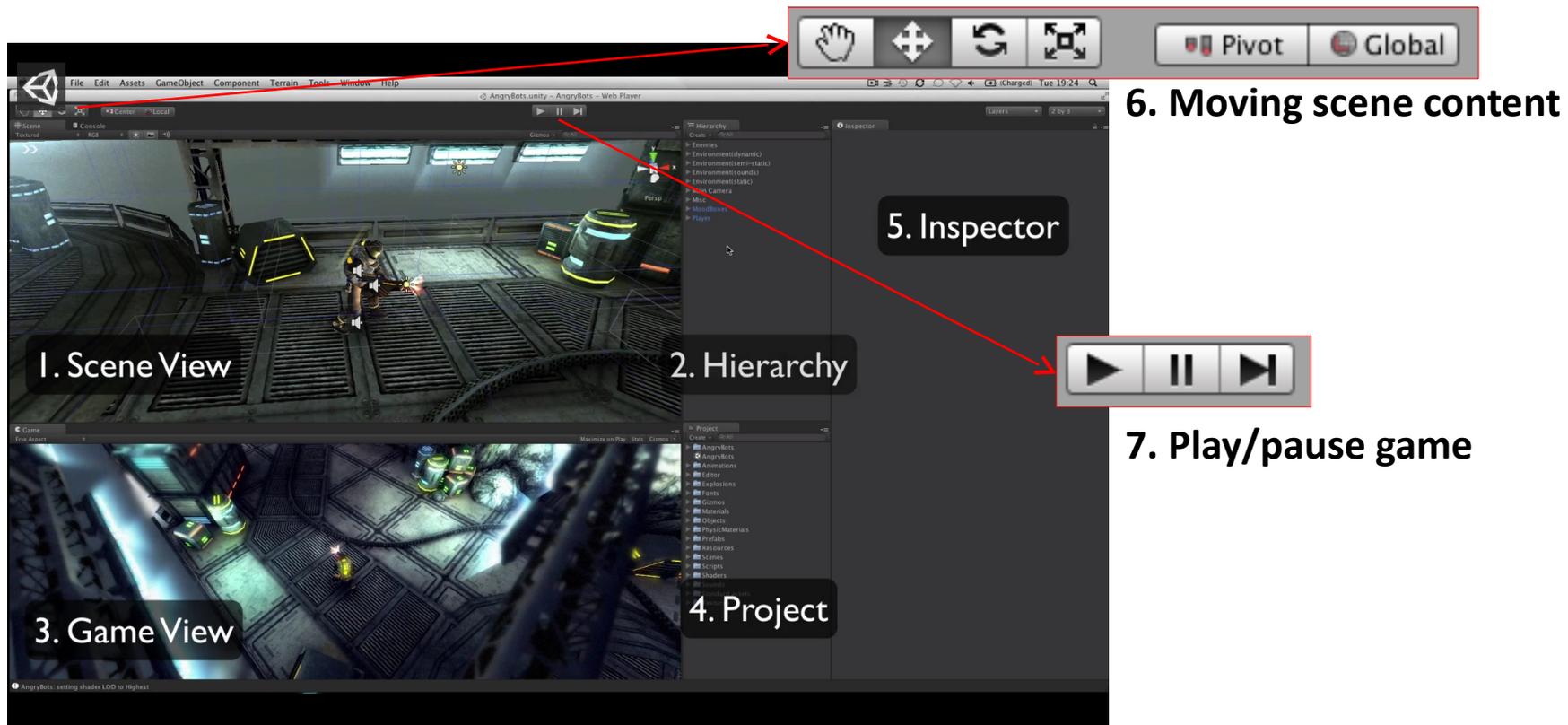
L'interface



3. Hierarchy : listes des objets chargés dans la scène, de tous leurs objets « enfants ».

La création de contenu 3D à l'aide du moteur de jeu Unity 3D

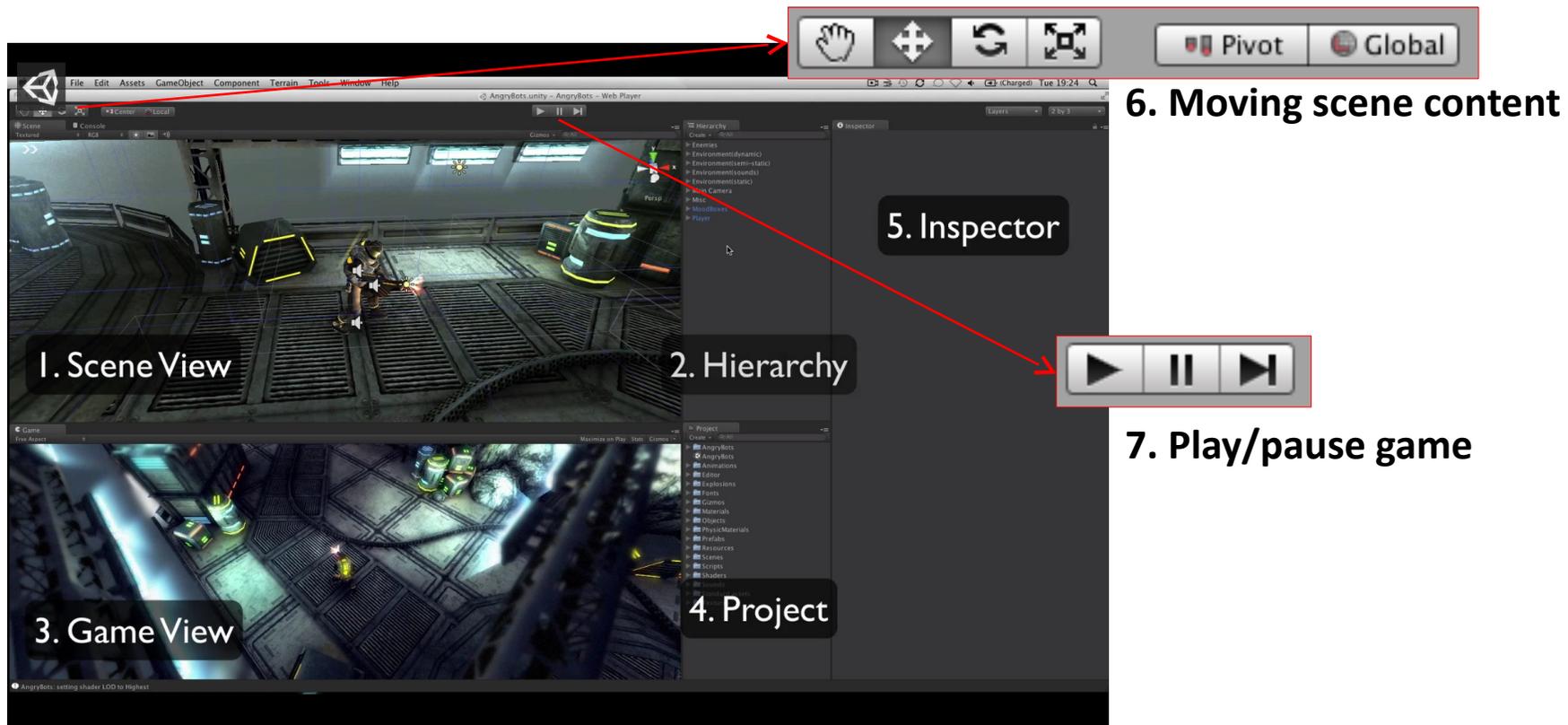
L'interface



4. Project (assets) : tous les objets « préfabriqués » disponibles (mesh, image, son, scripts, etc.)

La création de contenu 3D à l'aide du moteur de jeu Unity 3D

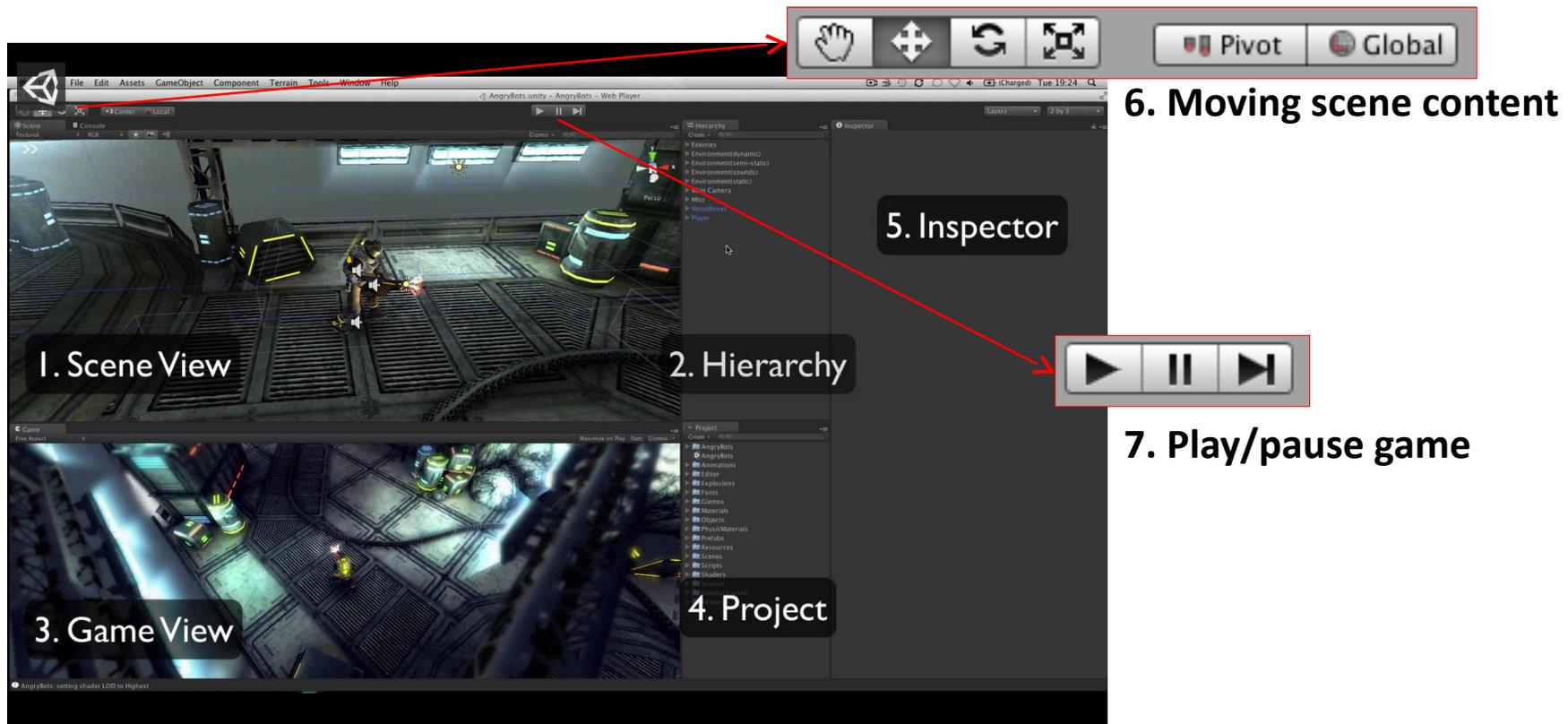
L'interface



5. Inspector : permet d'afficher et de modifier les propriétés des objets sélectionnés et les propriétés globales de la scène

La création de contenu 3D à l'aide du moteur de jeu Unity 3D

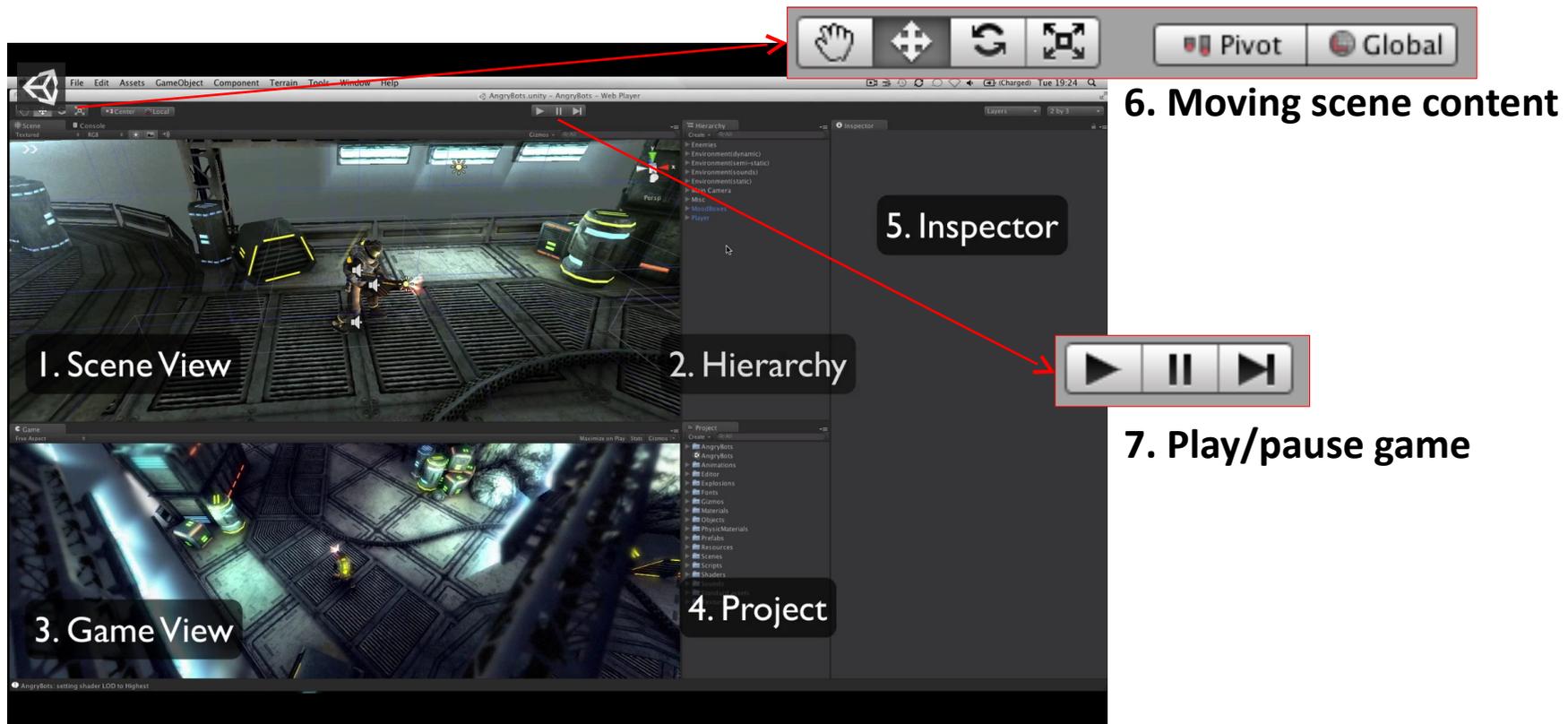
L'interface



6. Fonction de déplacement : Pan/Move selected/Rotate selected/Scaling

La création de contenu 3D à l'aide du moteur de jeu Unity 3D

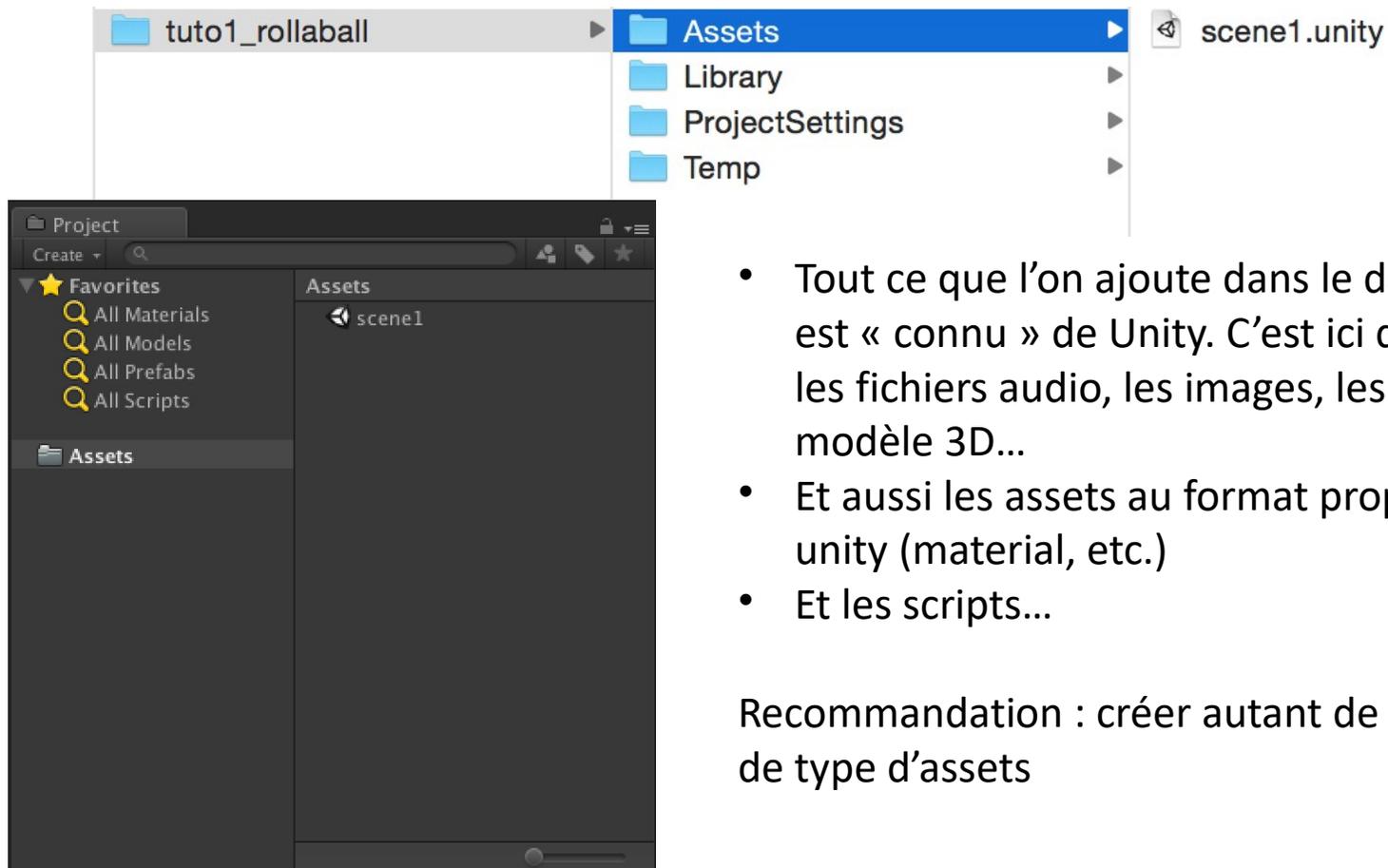
L'interface



7. Play/pause : tester el jeu dans la « game view »

La création de contenu 3D à l'aide du moteur de jeu Unity 3D

La structure de fichier



- Tout ce que l'on ajoute dans le dossier « Assets » est « connu » de Unity. C'est ici que l'on ajoute les fichiers audio, les images, les images, les modèle 3D...
- Et aussi les assets au format propriétaire de unity (material, etc.)
- Et les scripts...

Recommandation : créer autant de dossier qu'il y a de type d'assets

La création de contenu 3D à l'aide du moteur de jeu Unity 3D

Créer sa première scène

Une fois créé un nouveau projet, on sauve la scène :

-> Création d'une nouvelle scène : un nouveau monde 3D, de nouvelles « assets »

Economie de ressource machine : lorsque l'on doit construire des mondes très complexes (tout l'univers d'un jeu), on crée plusieurs scènes que l'on charge au fur et à mesure de l'avancée du joueur.

Pour vos projets, vous n'aurez toujours qu'une seule scène.

La création de contenu 3D à l'aide du moteur de jeu Unity 3D

Créer sa première scène

Une fois créé un nouveau projet, on sauve la scène :

-> Création d'une nouvelle scène : un nouveau monde 3D, de nouvelles « assets »

Economie de ressource machine : lorsque l'on doit construire des mondes très complexes (tout l'univers d'un jeu), on crée plusieurs scène que l'on charge au fur et à mesure de l'avancée du joueur.

Pour vos projets, vous n'aurez toujours qu'une seule scène.

La création de contenu 3D à l'aide du moteur de jeu Unity 3D

Créer sa première scène

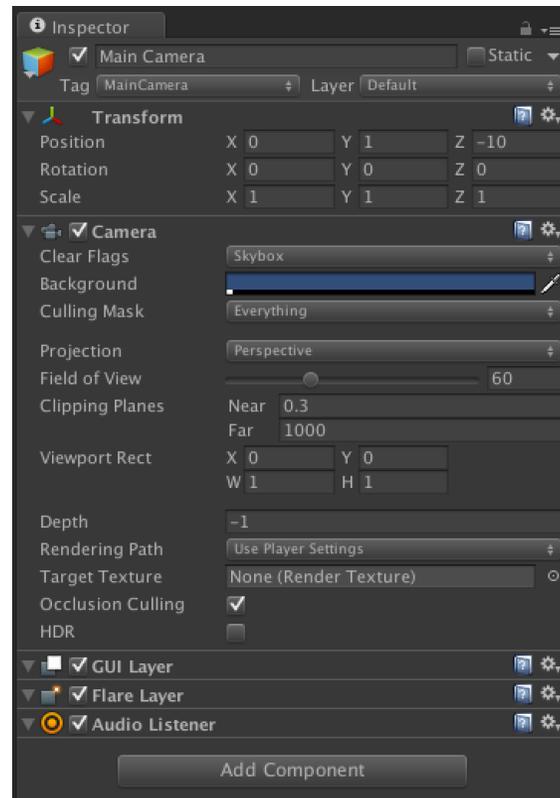
- Dans Unity, tout objet de la scène est un **GameObject**
- Un nombre arbitraire de **Component** sont attachés à un **GameObject** pour définir son comportement
- **GameObject**(name, layer, tags)
 - **Transform**(position, rotation, scale, parent)
 - **Mesh Renderer**(pour le "dessin de l'objet")
 - **Box Collider**(définit le "volume" utiliser pour la détection des collision)
 - **Rigidbody**(détermine les propriétés physique de l'objet – Objet solide...)

La création de contenu 3D à l'aide du moteur de jeu Unity 3D

Créer sa première scène

Par défaut :

- Une caméra



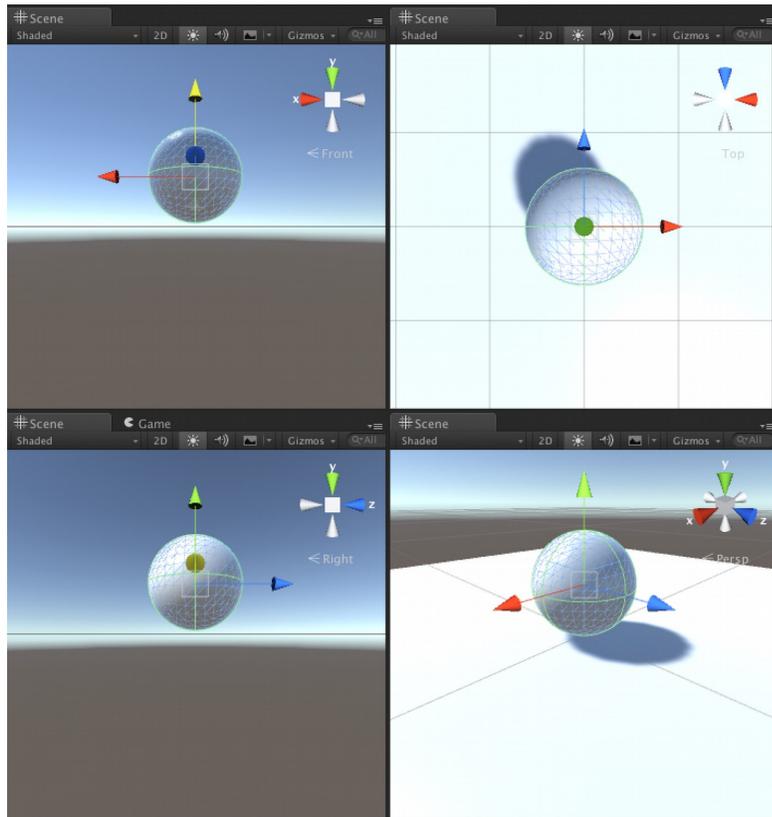
- Une lumière



La création de contenu 3D à l'aide du moteur de jeu Unity 3D

Créer sa première scène

Ajouter dans la scène :



- Un plan 3D
 - > changer le nom (« Floor »)
 - > changer sa taille
- Une sphère
 - > changer le nom (« Player »)
 - > touche « F » pour positionner la caméra sur l'objet
 - > la déplacer vers le haut

Ajouter dans le projet :

- Un material
 - > changer le nom (« background »)
 - > changer sa couleur

Appliquer le material au sol

La création de contenu 3D à l'aide du moteur de jeu Unity 3D

Créer sa première scène

Déplacer le « Player »

- On va ajouter des propriétés physique à l'objet :
Component > Physics > Rigidbody ou directement dans l'inspecteur
- Ajouter un script pour récupérer les entrées du clavier et déplacer la sphère en conséquence

« Add Component » dans l'inspecteur de la sphère

-> New Scripts

-> Nommer le script « PlayerController »

-> Choisir le langage : C#

Déplacer le script dans le dossier « Scripts »

Editer le script avec Mono

La création de contenu 3D à l'aide du moteur de jeu Unity 3D

Créer sa première scène

Déplacer le « Player »

- On enlève les classes déjà présente
- Ce qu'on veut faire : « vérifier à chaque frame, si l'utilisateur a appuyé sur une touche et déplacer l'objet »

-> deux choix :

« Update() » : avant le rendu de la frame

et « FixedUpdate() » : avant le calcul de physique

Qu'est-ce qu'on écrit ?

- On utilise la classe « [Input](#) » pour récupérer les touches du clavier. Mais comment savoir ce qu'il nous faut en plus ?

Mac:
CMD + '

PC:
CTRL + '

Ouvre le document de référence de l'API

La création de contenu 3D à l'aide du moteur de jeu Unity 3D

Créer sa première scène

Déplacer le « Player »

- `Input.GetAxis(...)`

Le concept de “Axis” fait référence à un déplacement saisi par l'utilisateur à l'aide d'un dispositif d'entrée... S'adapte automatique à la plateforme et aux contrôleurs disponibles (clavier, souris, multi-touch, gyroscope, etc.)... `GetAxis` donne automatique des valeurs entre -1 et +1 dans l'axe correspondant.

Exemple pour le clavier :

Vertical Axis : default keys = "Up Arrow" "Down Arrow" "W" and "S"

Horizontal Axis : "Left Arrow" "Right Arrow" "A" and "D"

La création de contenu 3D à l'aide du moteur de jeu Unity 3D

Créer sa première scène

Déplacer le « Player »

- On utilise la classe « [Rigidbody](#) » pour modifier les propriétés de notre objet physique (ex. appliquer une force qui déplace l'objet).

Voir la référence de l'API pour la méthode « [AddForce\(...\)](#) »

Pour accéder au Component "Rigidbody" de notre objet on doit tout d'abord créer une variable qui contient une « référence » de notre objet Rigidbody

```
4 public class PlayerController : MonoBehaviour {  
5  
6     private Rigidbody rg;  
7  
8     void Start () {  
9         rg = GetComponent<Rigidbody> ();  
10    }  
11
```

La création de contenu 3D à l'aide du moteur de jeu Unity 3D

Créer sa première scène

Déplacer le « Player »

- La méthode AddForce prend un objet **Vector3** en entrée

On doit construire un Vector3, à l'aide des nos 2 valeurs de déplacement horizontal et vertical, puis appliquer la force au Rigidbody

```
void FixedUpdate() {  
  
    float moveHorizontal = Input.GetAxis ("Horizontal");  
    float moveVertical = Input.GetAxis ("Vertical");  
  
    Vector3 movement = new Vector3 (moveHorizontal, 0.0f, moveVertical);  
  
    rg.AddForce (movement);  
  
}
```

La création de contenu 3D à l'aide du moteur de jeu Unity 3D

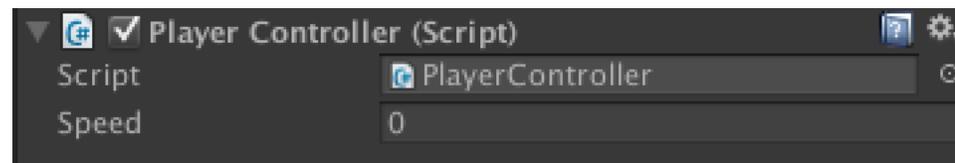
Créer sa première scène

Déplacer le « Player »

- Le déplacement est trop lent... Il nous manque un paramètre pour ajuster la vitesse.
-> variable publique qui augmente le pas d'incrémentatation du déplacement (0,x) et (0,z) à chaque frame.

Variable d'instance : « speed »

```
public float speed;
```



Le fait qu'elle soit publique, rend la valeur de la variable modifiable dans l'interface

La création de contenu 3D à l'aide du moteur de jeu Unity 3D

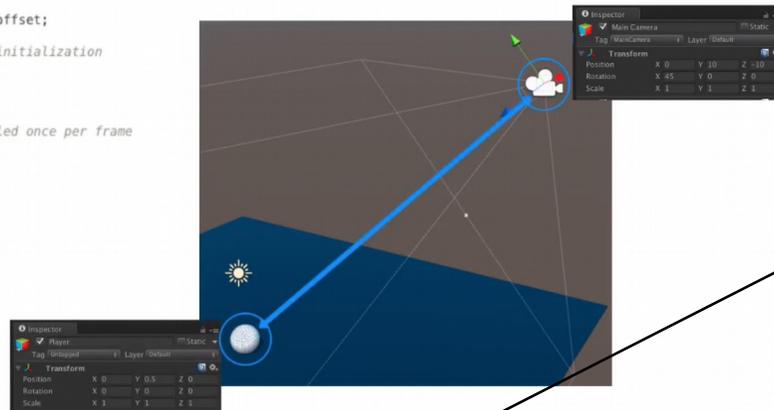
Créer sa première scène

Déplacer la « Camera »

- On a besoin de deux variables :
 - Une référence du « Player » (un GameObject)
 - Un vecteur qui donne la distance entre l'objet et la camera

```
1 using UnityEngine;
2 using System.Collections;
3
4 public class CameraController : MonoBehaviour {
5
6     public GameObject player;
7
8     private Vector3 offset;
9
10    // Use this for initialization
11    void Start () {
12
13    }
14
15    // Update is called once per frame
16    void Update () {
17
18    }
19 }
20
```

Start :



A la fin du calcul d'une frame (après un Update et un FixedUpdate)

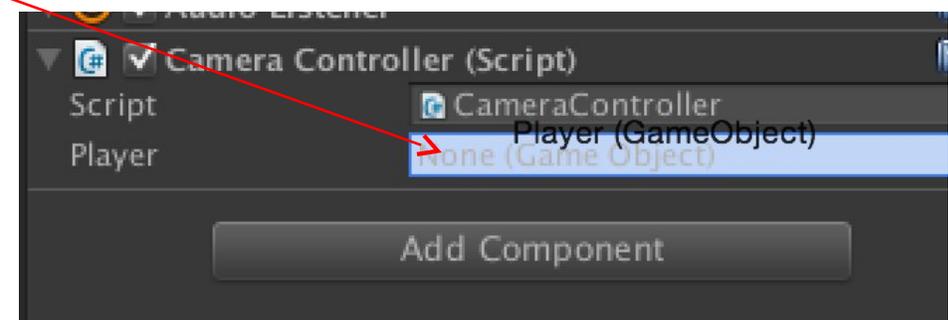
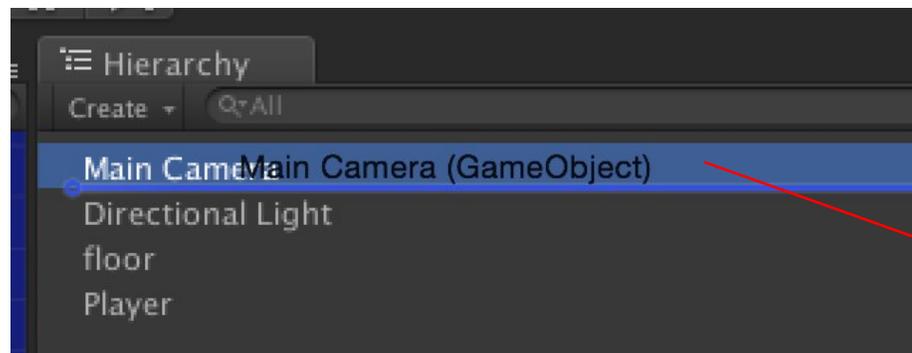
LateUpdate : **transform.position = player.transform.position + offset;**

La création de contenu 3D à l'aide du moteur de jeu Unity 3D

Créer sa première scène

Déplacer la « Camera »

- Enfin, on crée véritablement la référence vers le game object (Player) en glissant/déposant l'objet Player de la Hierarchy dans la



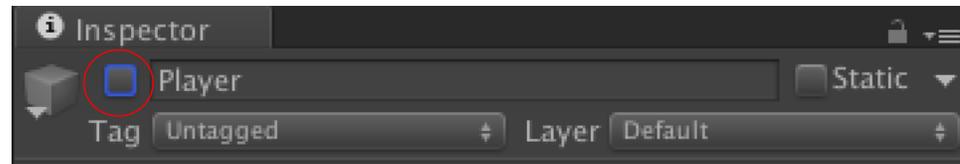
La création de contenu 3D à l'aide du moteur de jeu Unity 3D

Créer sa première scène

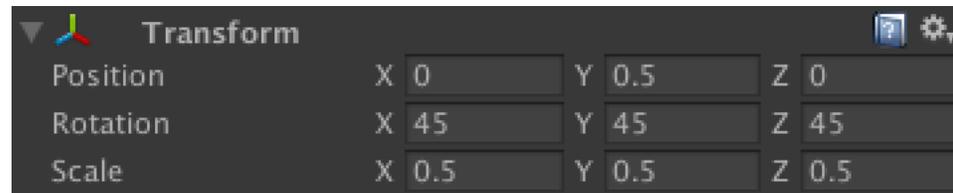
Créer des objets à collecter

Ajouter un nouveau cube dans la scène (reset position), appelé "PickUp"

Désactiver le Player pour y voir plus clair...



Modifier la taille et l'orientation de l'objet.



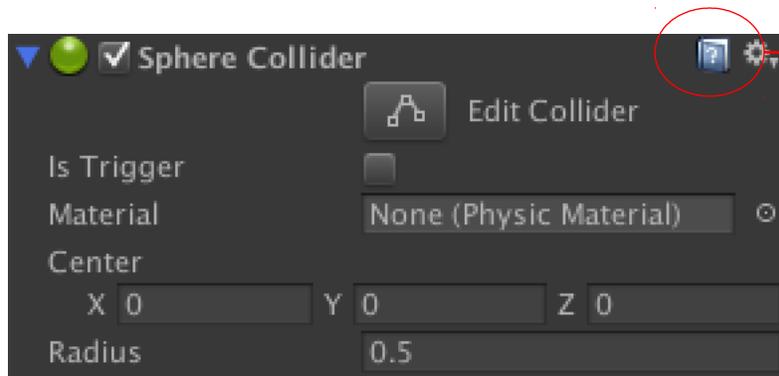
La création de contenu 3D à l'aide du moteur de jeu Unity 3D

Créer sa première scène

Créer des objets à collecter

Collecter les objets lorsque le Player rentre en collision avec lui.
-> détection de collision dans un script.

On a besoin de modifier le script du Player -> On utilise une classe Collider



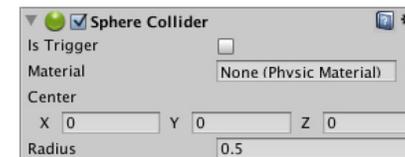
Ouvre le document de référence du Component.

Sphere Collider

SWITCH TO SCRIPTING

Ouvre le document de référence de l'API.

The Sphere Collider is a basic sphere-shaped collision primitive.



La création de contenu 3D à l'aide du moteur de jeu Unity 3D

Créer sa première scène

Créer des objets à collecter

Collecter les objets lorsque le Player rentre en collision avec lui.
-> détection de collision dans un script.

On a besoin de modifier le script du Player -> On utilise une classe Collider

On utilise le message « OnTriggerEnter »

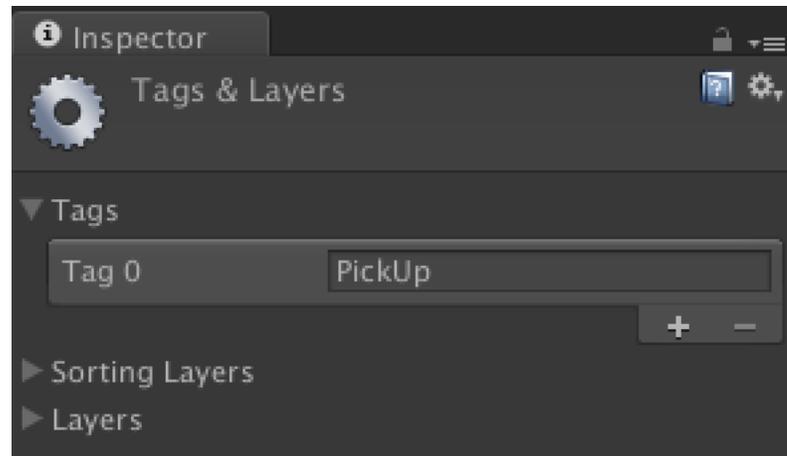
A l'intérieur de la méthode, on teste si l'objet qui rentre en collision avec le Player est un Pickup. Si c'est le cas, on le désactive (voir la référence de GameObject - On s'intéresse à Tag, SetActive et CompareTag...)

La création de contenu 3D à l'aide du moteur de jeu Unity 3D

Créer sa première scène

Créer des objets à collecter

On ajoute un Tag au Prefab du Pickup



La création de contenu 3D à l'aide du moteur de jeu Unity 3D

Créer sa première scène

Créer des objets à collecter

Collecter les objets lorsque le Player rentre en collision avec lui.
-> détection de collision dans un script.

On a besoin de modifier le script du Player -> On utilise une classe Collider

On utilise le message « OnTriggerEnter »

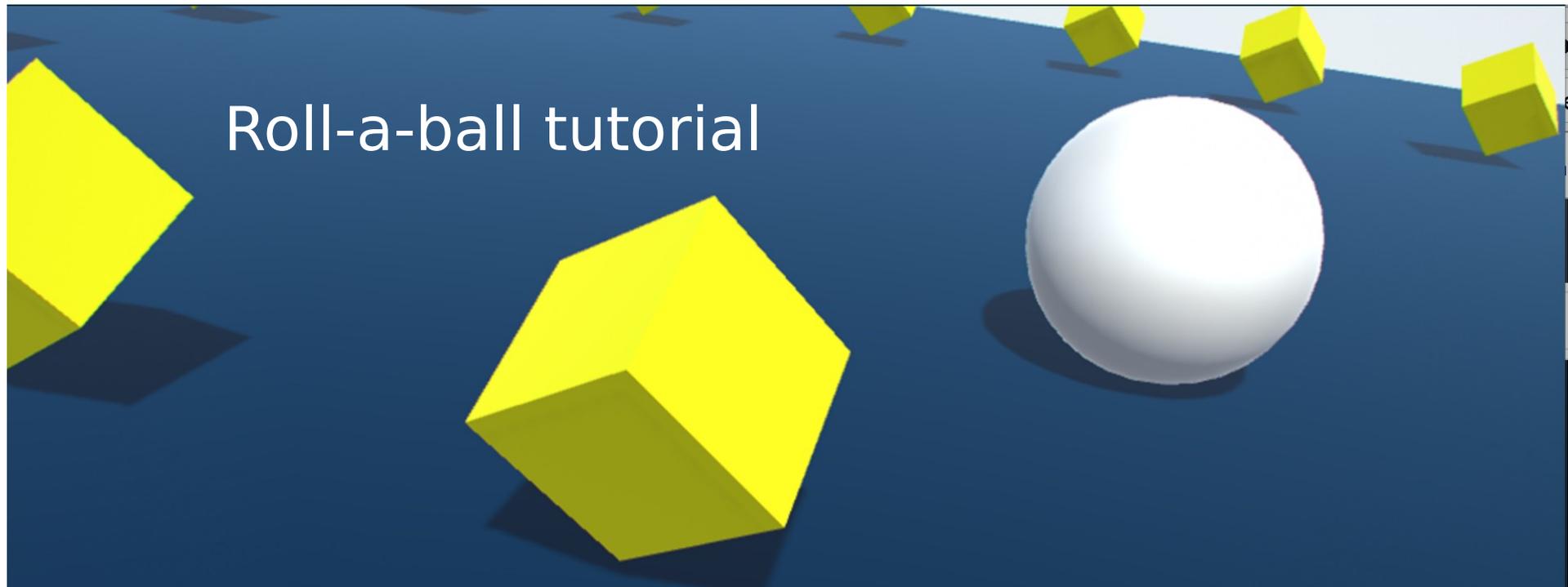
```
void OnTriggerEnter (Collider Other) {  
    if (Other.gameObject.CompareTag ("PickUp")) {  
        Other.gameObject.SetActive(false);  
    }  
}
```

La création de contenu 3D à l'aide du moteur de jeu Unity 3D

Créer sa première scène

Voir le détail dans :

<https://unity3d.com/fr/learn/tutorials/projects/roll-ball-tutorial>

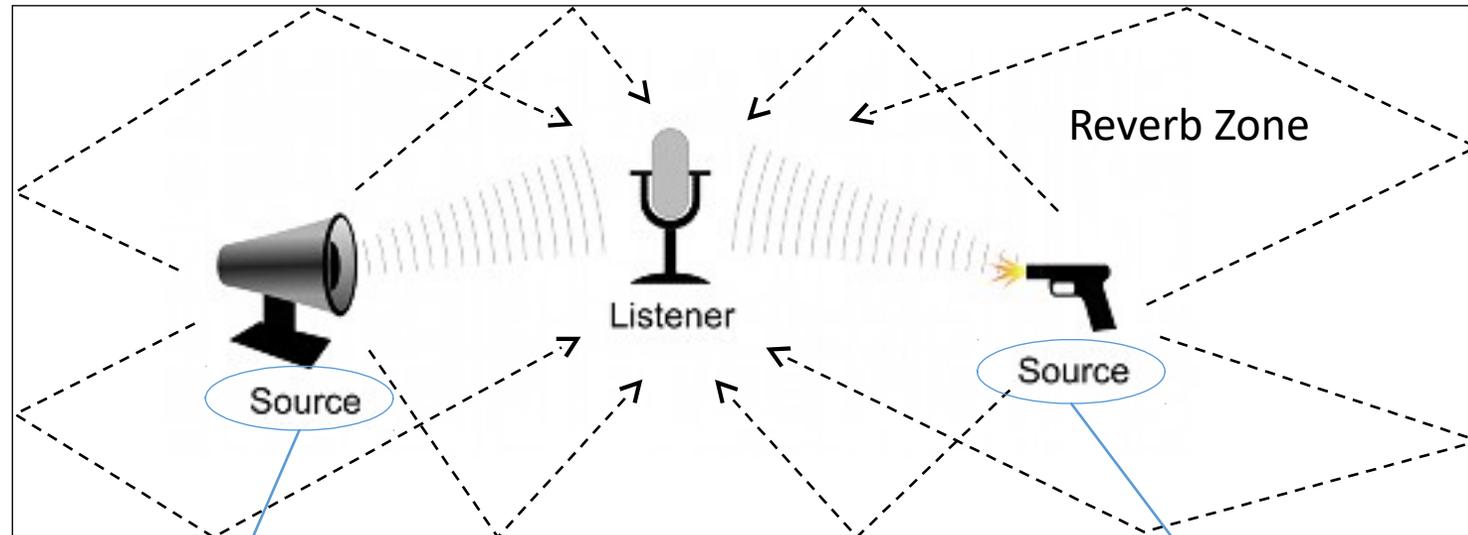


La création de contenu 3D à l'aide du moteur de jeu Unity 3D

Créer sa première scène

Le son

Trois concepts fondamentaux : **Listener**, **Audio Source** et **Audio Clip + Reverb Zone**



Audio Clip



Audio Clip

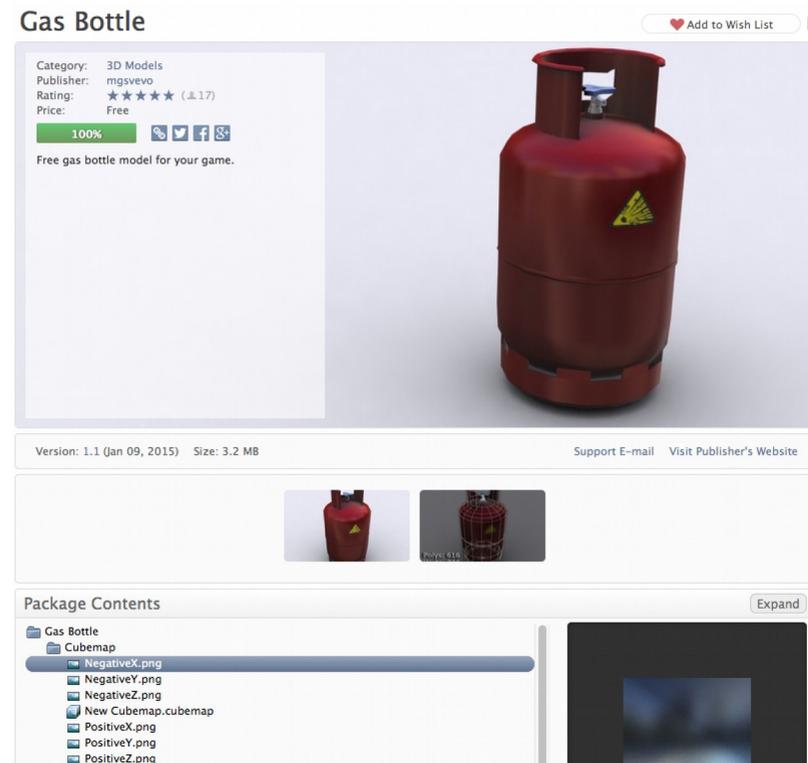
La création de contenu 3D à l'aide du moteur de jeu Unity 3D

Créer sa première scène

Ajouter une source sonore ponctuelle

1) Importer un modèle 3D de « Bouteille de gaz »

Asset Store

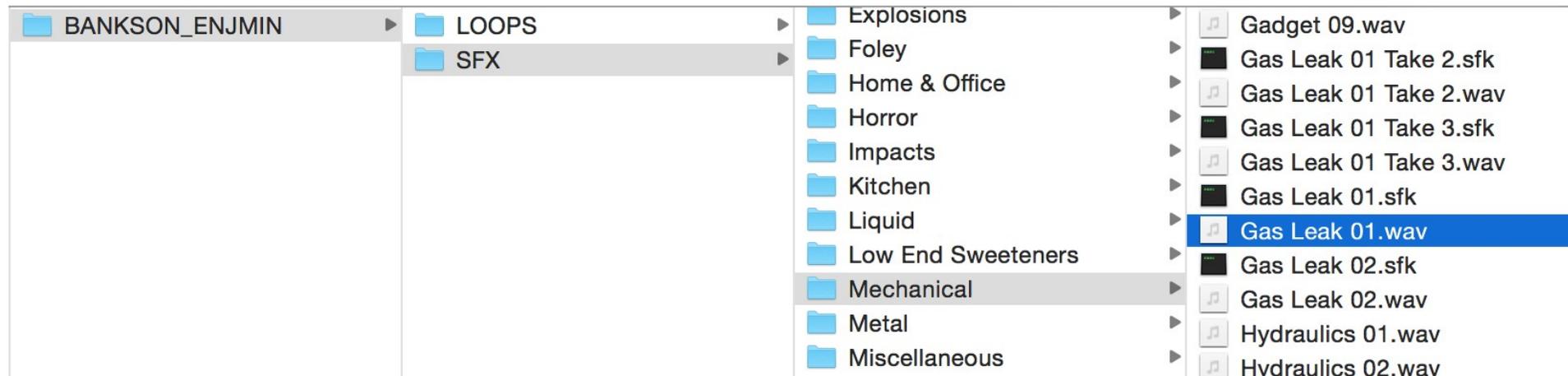


La création de contenu 3D à l'aide du moteur de jeu Unity 3D

Créer sa première scène

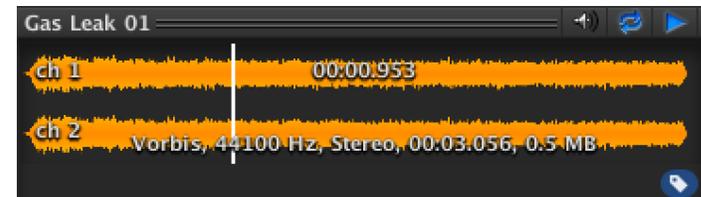
Ajouter une source sonore ponctuelle

2) Créer un son « fuite de gaz »



3) L'ajouter dans le dossier « Sounds » des Assets

Écoutez le en boucle... Quel est le problème ?

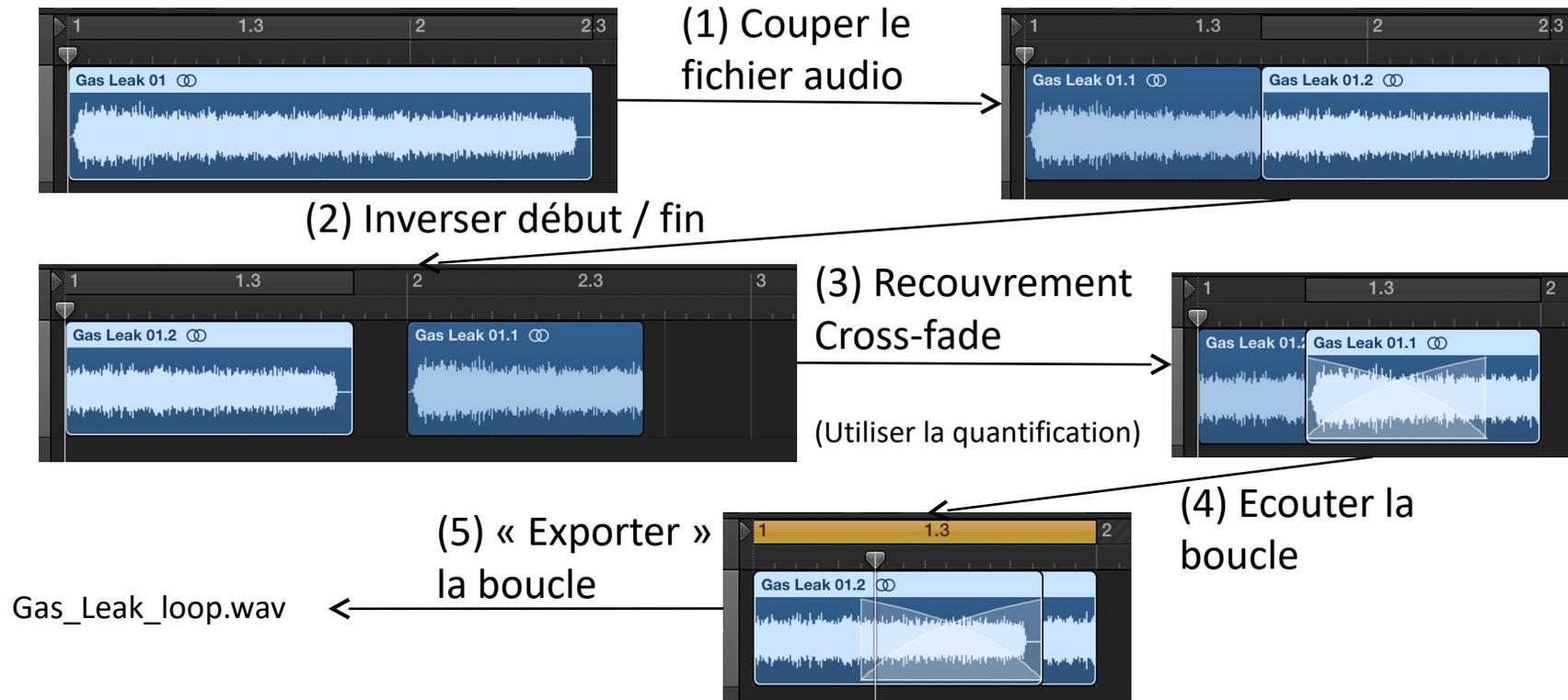


La création de contenu 3D à l'aide du moteur de jeu Unity 3D

Créer sa première scène

Ajouter une source sonore ponctuelle

3) Créer une boucle à partir du sfx (Exemple avec Logic Pro X)

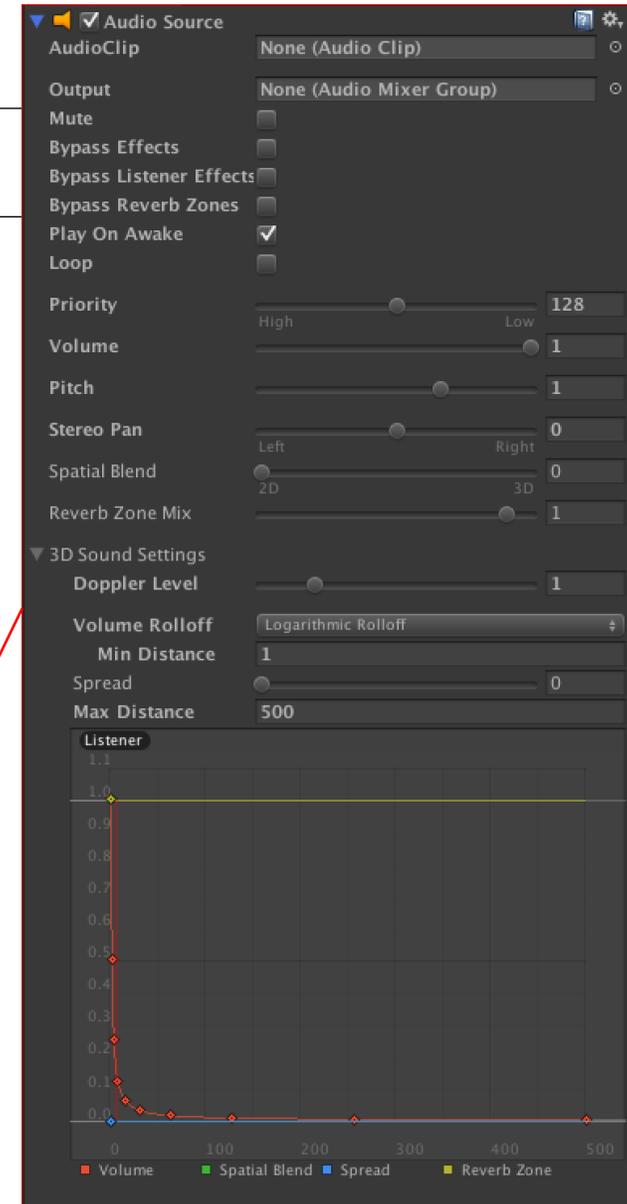
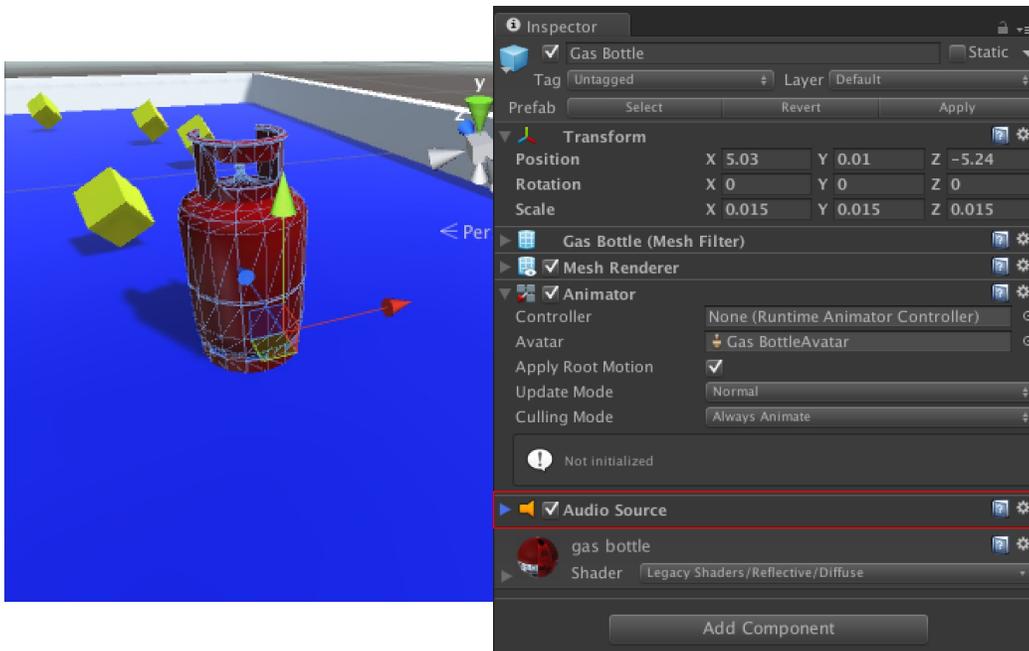


La création de contenu 3D à l'aide du moteur de jeu Unity 3D

Créer sa première scène

Ajouter une source sonore ponctuelle

4) Sélectionner la bouteille de gaz dans la scène.
Ajouter un « Component » Audio Source



La création de contenu 3D à l'aide du moteur de jeu Unity 3D

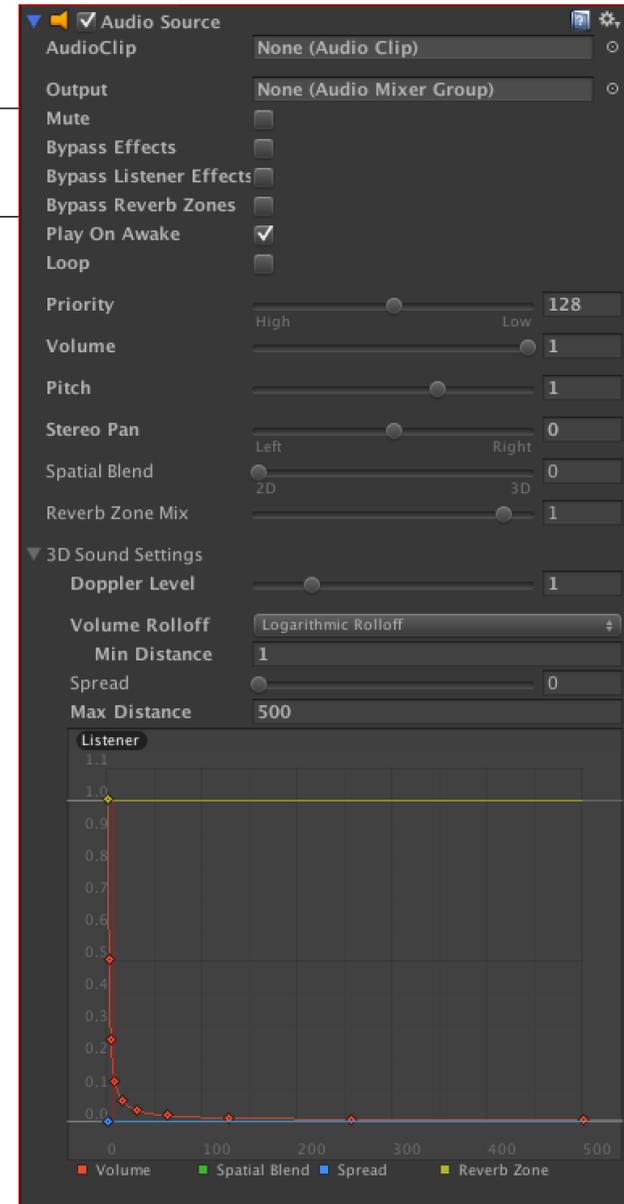
Créer sa première scène

Ajouter une source sonore ponctuelle

4) Sélectionner la bouteille de gaz dans la scène.
Ajouter un « Component » Audio Source

- Ajouter l'audio clip
- Cocher « Loop »

• Jouer la scène...
Qu'est-ce que vous entendez (ou n'entendez pas ?)



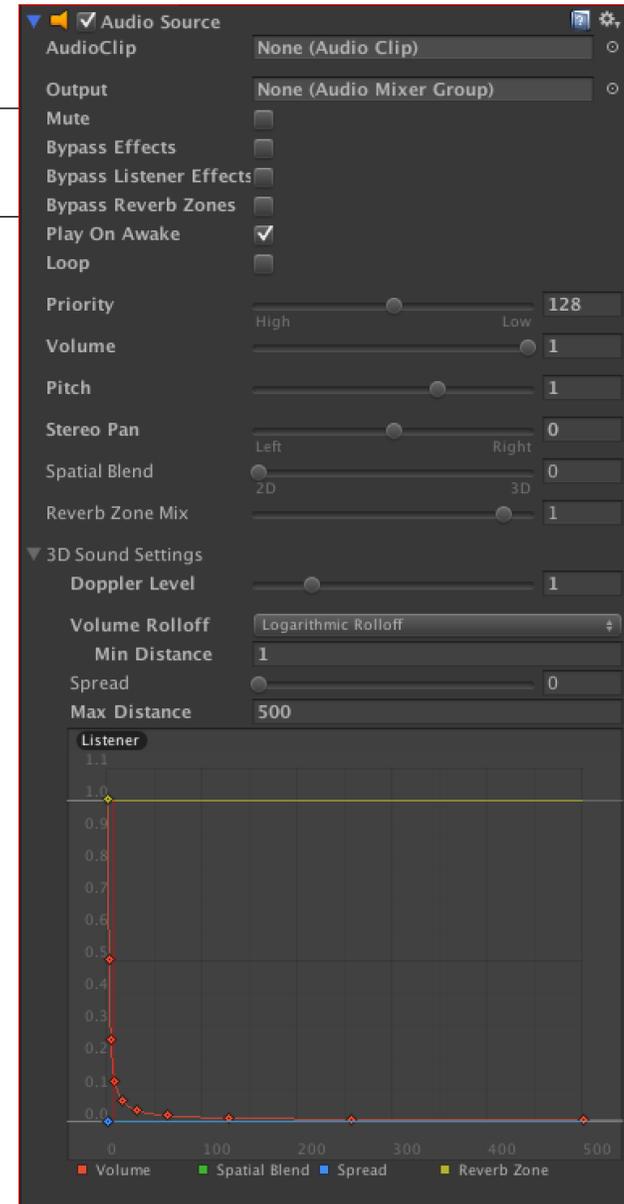
La création de contenu 3D à l'aide du moteur de jeu Unity 3D

Créer sa première scène

Ajouter une source sonore ponctuelle

4) Sélectionner la bouteille de gaz dans la scène.
Ajouter un « Component » Audio Source

- Spatial Blend : 2D -> 3D
- Régler les 3D Sound Setting
-> Min Distance / Max Distance
-> Modifier la courbe de Rolloff

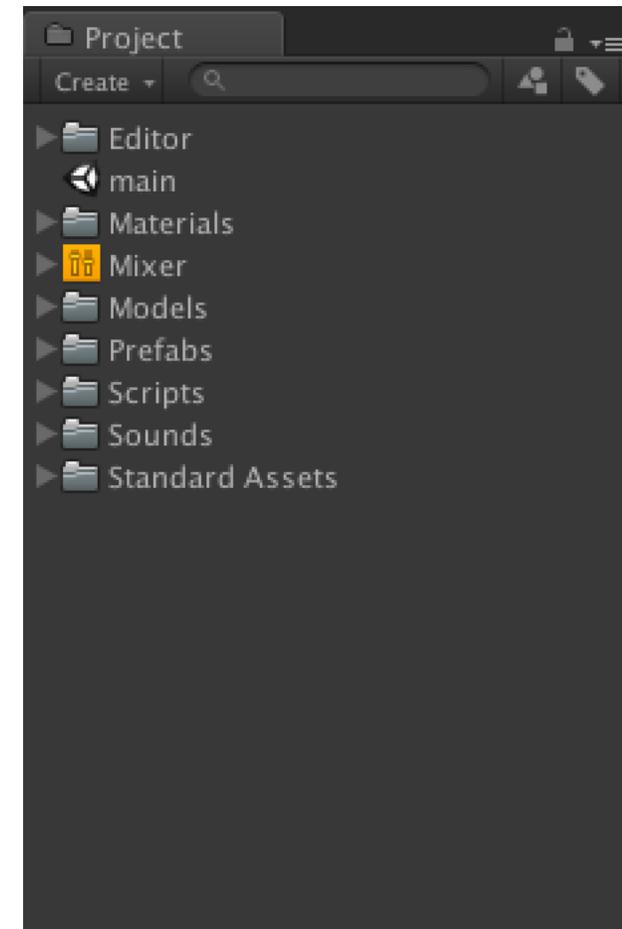
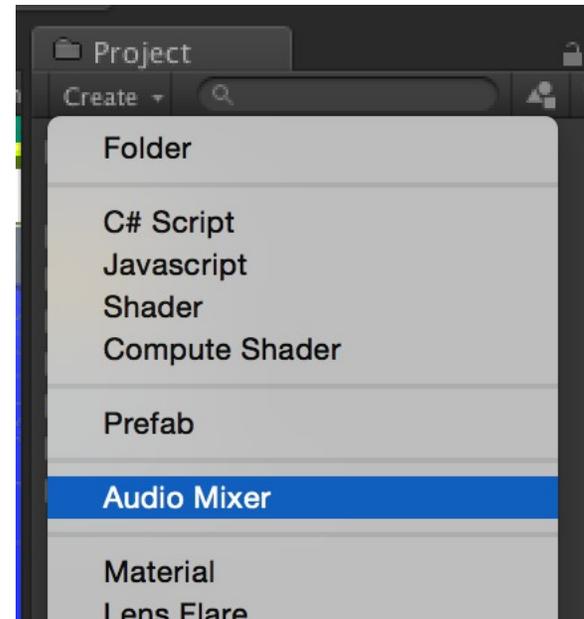


La création de contenu 3D à l'aide du moteur de jeu Unity 3D

Créer sa première scène

Ajouter un Mixer

1) Le Mixer est un asset... Créé dans la fenêtre de projet

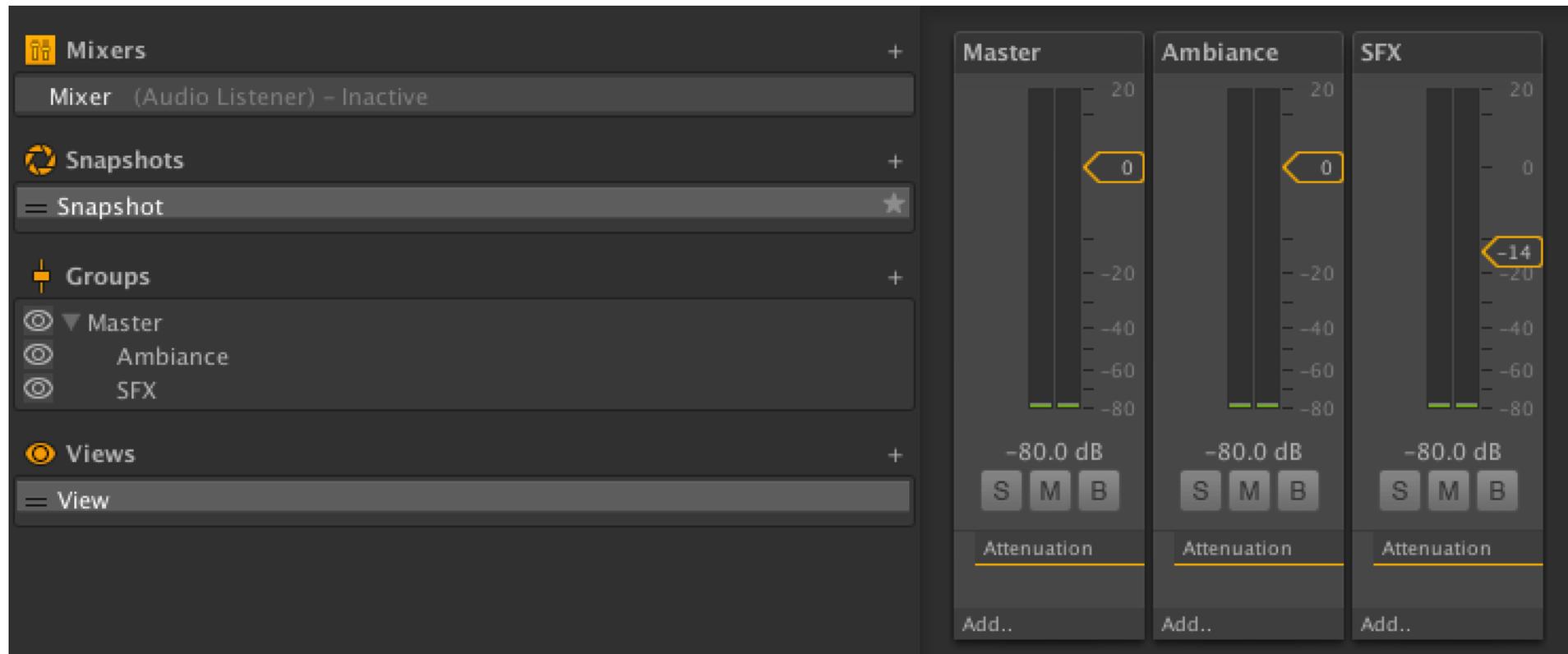


La création de contenu 3D à l'aide du moteur de jeu Unity 3D

Créer sa première scène

Ajouter un Mixer

2) On créé des Groups pour le mixer Master



The screenshot displays the Unity Audio Mixer interface. On the left, a sidebar shows the hierarchy: Mixers (Audio Listener) - Inactive, Snapshots (Snapshot), and Groups (Master, Ambiance, SFX). The main area shows three mixer groups, each with a vertical fader and a numerical value. The Master group has a value of 0, Ambiance has a value of 0, and SFX has a value of -14. Each group also has a -80.0 dB label and S, M, B buttons. The SFX group has a yellow box around the -14 value.

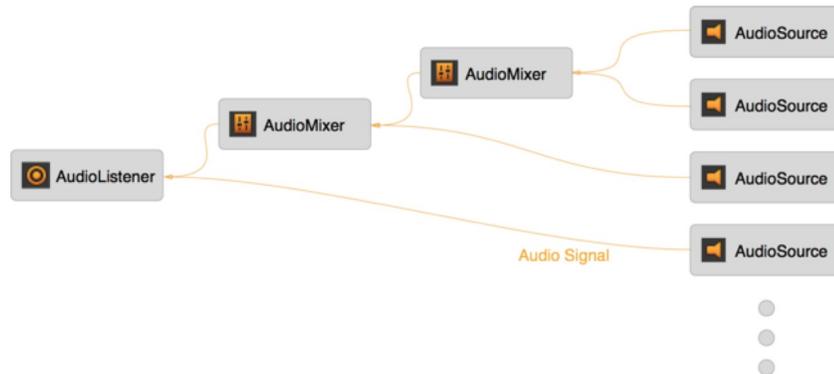
Group	Value
Master	0
Ambiance	0
SFX	-14

La création de contenu 3D à l'aide du moteur de jeu Unity 3D

Créer sa première scène

Ajouter un Mixer

2) On créé des Groups pour le mixer Master



On peut créer un « graphe » audio complexe. Ici on ne fait que des AudioMixer en parallèle :
-> Un pour les SFX, un pour les ambiances, etc.

Ensuite, on assigne une Audio Source à un groupe de mixer



La création de contenu 3D à l'aide du moteur de jeu Unity 3D

Créer sa première scène

A faire : déclencher un son lorsque l'on collecte un pick-up

La création de contenu 3D à l'aide du moteur de jeu Unity 3D

Créer sa première scène

A faire : créer un menu

La création de contenu 3D à l'aide du moteur de jeu Unity 3D

A VOIR – SCÉANCES N°2

- CRÉER UN MENU !

- LE MIXER AUDIO

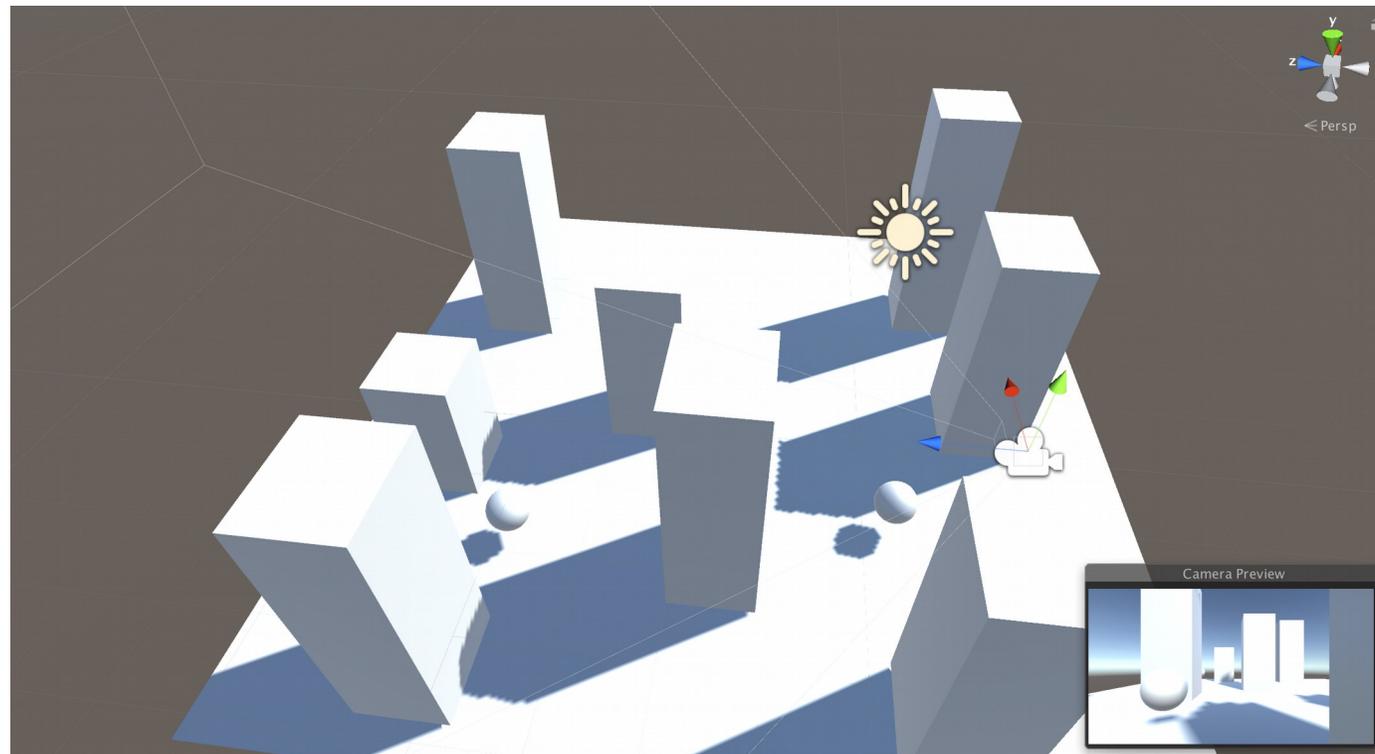
Introduction à la Réalité Virtuelle

Caméra en Survol dans une scène 3D



Introduction à la réalité virtuelle : caméra en survol dans une scène 3D

1) Construire un monde abstrait avec des Cubes des Sphères et un Plan

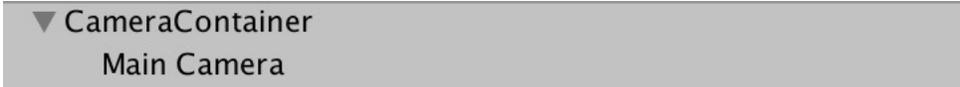


Introduction à la réalité virtuelle : caméra en survol dans une scène 3D

2) Ajouter une animation à la caméra

Méthode rapide :

- Créer un objet vide dans la scène nommée « CameraContainer »
- Glisser l'objet camera « dans » l'objet « CameraContainter » (qui devient l'objet Parent)



▼ CameraContainer
Main Camera

- Sélectionner l'objet « CameraContainer » dans la hiérarchie et le déplacer à la position de départ du survol.

NB : pour la VR, on ne déplace jamais directement la caméra

Introduction à la réalité virtuelle : caméra en survol dans une scène 3D

2) Ajouter une animation à la caméra

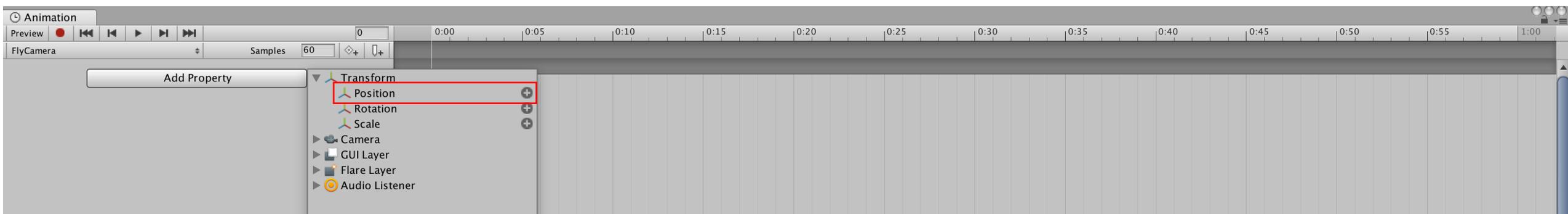
Méthode rapide :

- Ouvrir la fenêtre d'Animation (Window>Animation)
- Appuyer sur le bouton « Create » dans la fenêtre d'animation

To begin animating Main Camera, create an Animator and an Animation Clip.

Create

- Sauver l'animation sous le nom « FlyCamera » dans un dossier « Animation » du dossier « Assets » du projet Unity.
- Appuyer sur le bouton « Add Property » et sélectionner « Transform>Position »

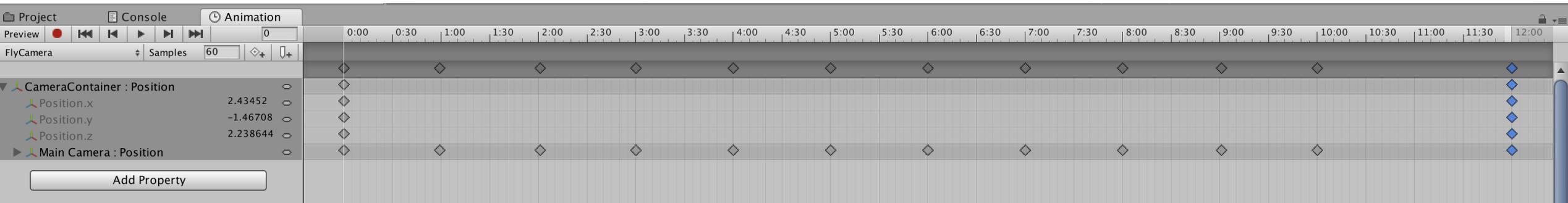


Introduction à la réalité virtuelle : caméra en survol dans une scène 3D

2) Ajouter une animation à la caméra

Méthode rapide :

- Appuyer sur le bouton  pour enregistrer l'animation
- Déplacer le curseur de lecture de l'animation à 1:00 et déplacer la caméra à une 1^{ère} étape de son déplacement
- Déplacer le curseur de lecture de l'animation à 1:00 et déplacer la caméra à une 2nde étape de son déplacement
- Etc..
- Jusqu'à retourner au point de départ à la 12^{ème} secondes

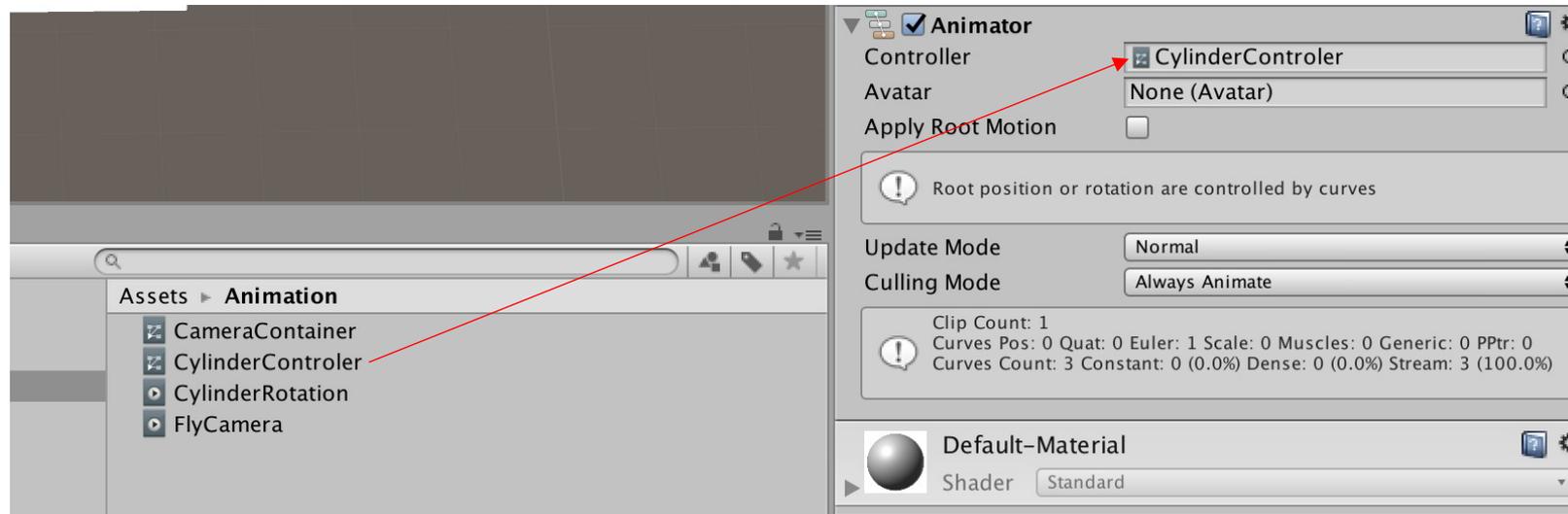


Introduction à la réalité virtuelle : caméra en survol dans une scène 3D

2) Ajouter un cylindre qui tourne sur lui-même

Méthode pas à pas :

- Créer un asset « Animator Controller », nommé « CylinderControler »
- Créer un asset « Animation », nommé « CylinderRotation »
- Ajouter un component « Animator » au cylindre et assigner la variable « Controler » à l'asset « CylinderControler »

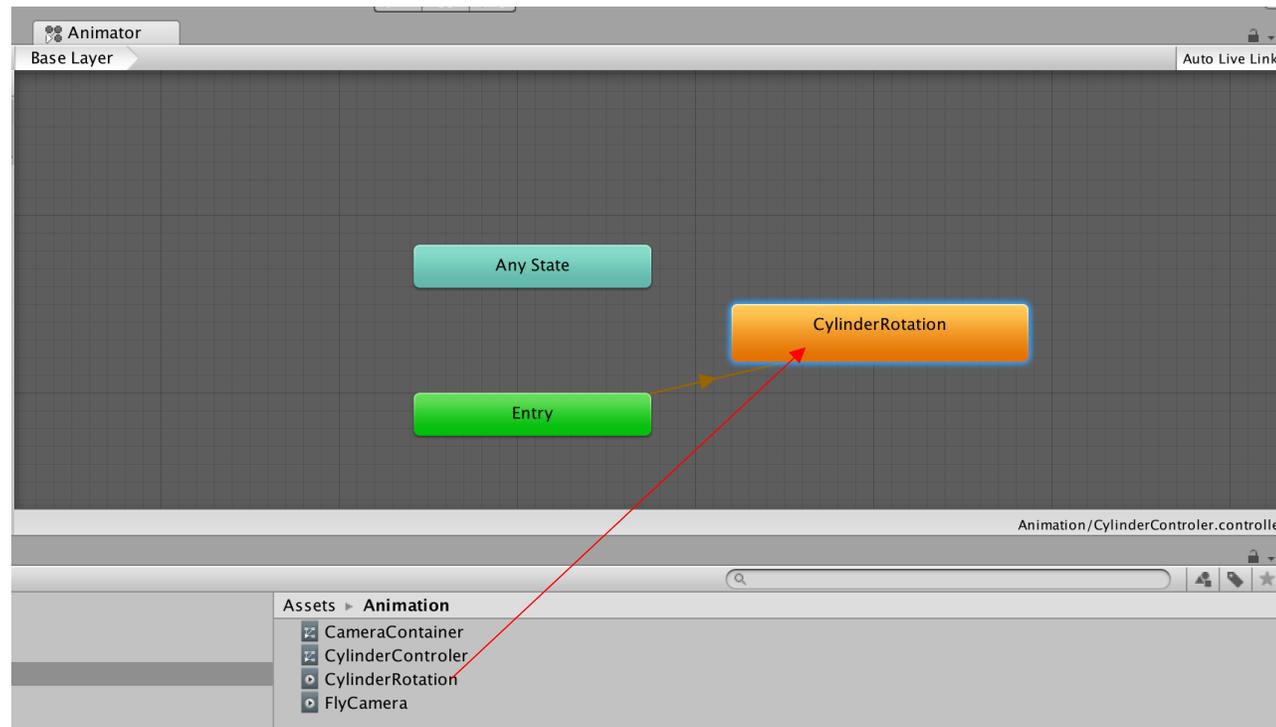


Introduction à la réalité virtuelle : caméra en survol dans une scène 3D

2) Ajouter un cylindre qui tourne sur lui-même

Méthode pas à pas :

- Ouvrir la fenêtre « Animator » et glisser l'animation « CylinderRotation »

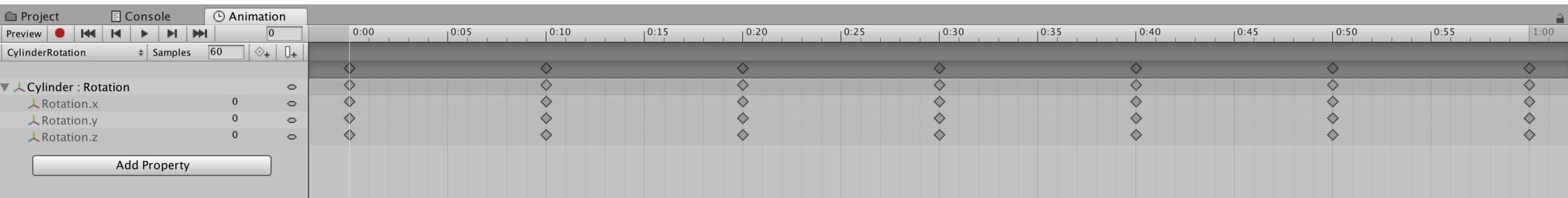


Introduction à la réalité virtuelle : caméra en survol dans une scène 3D

2) Ajouter un cylindre qui tourne sur lui-même

Méthode pas à pas :

- Sélectionner l'animation, ouvrir la fenêtre d'animation et procéder comme précédemment pour appliquer une rotation de 360° en une seconde



Introduction à la réalité virtuelle : caméra en survol dans une scène 3D

2) Ajouter un cylindre qui tourne sur lui-même

A faire...

- Déclencher l'animation de rotation avec un bouton en overlay

Introduction à la réalité virtuelle : caméra en survol dans une scène 3D

3) Animation d'un objet 3D, exporté de Blender

La création de contenu 3D à l'aide du moteur de jeu Unity 3D

A VOIR – SCÉANCES N°2

- L'ANIMATION DU DÉBUT À LA FIN AVEC UN MESH EXPORTÉ DE BLENDER
- DÉCLENCHEMENT DE L'ANIMATION AVEC L'INTERACTION VR...

La vidéo 360-3D

La vidéo 3D-360

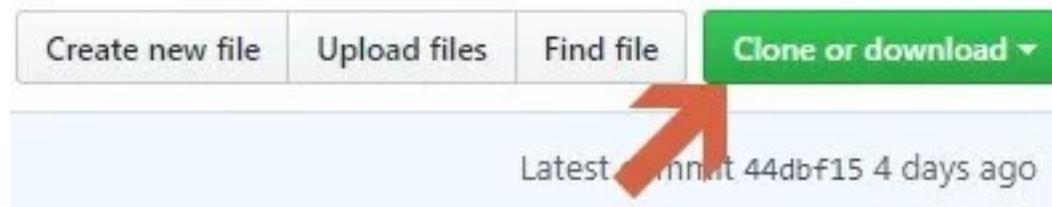


Introduction à la réalité virtuelle : caméra en survol dans une scène 3D

1) Intégrer le Shader « Skybox-PanoramicBeta » au projet

- Télécharger le shader sur « github »

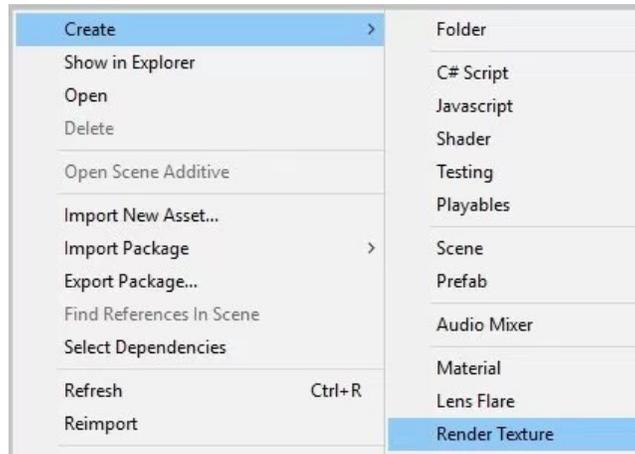
<https://github.com/Unity-Technologies/SkyboxPanoramicShader>



- Copier le fichier Skybox-PanoramicBeta.shader dans un dossier « Shaders » du dossier « Assets » du projet

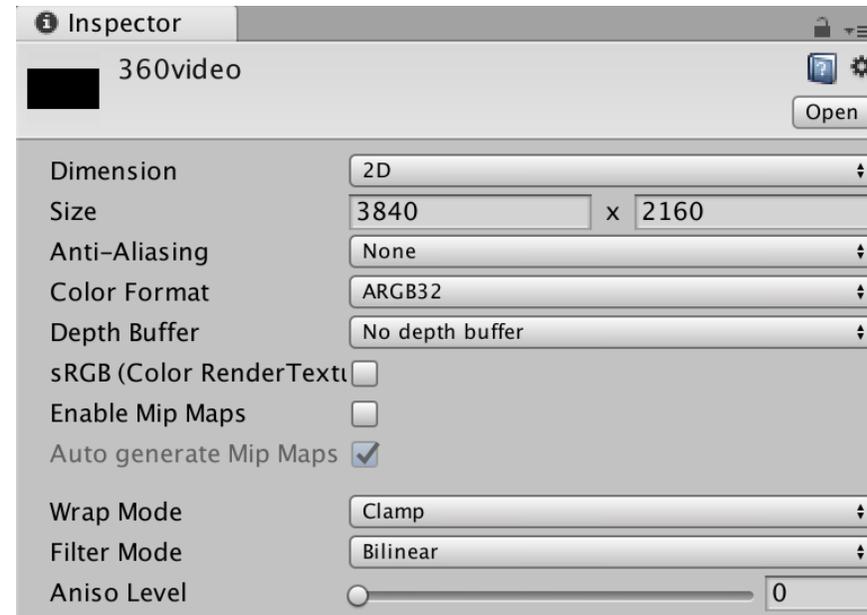
Introduction à la réalité virtuelle : caméra en survol dans une scène 3D

2) Créer un asset « RenderTexture » nommé « 360video »



Modifier la taille de la texture pour qu'elle corresponde à la taille de la vidéo

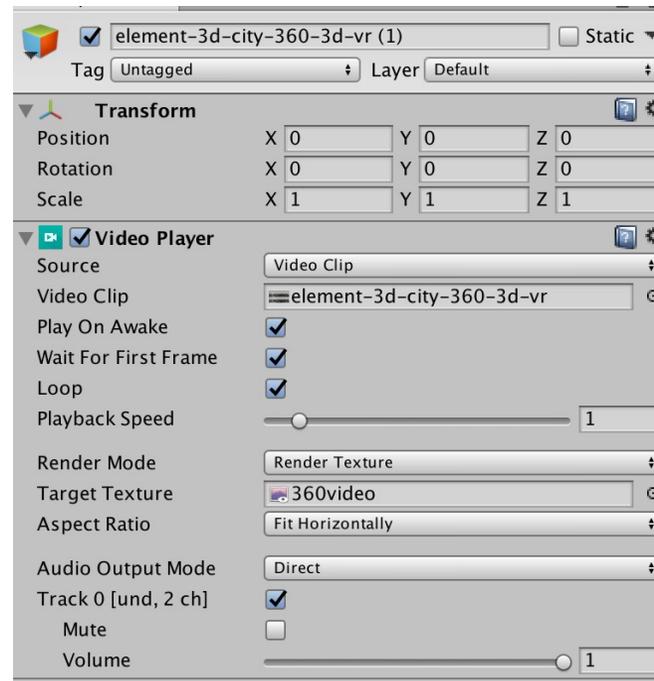
Changer le paramètre « depth buffer » à « No depth buffer »



Introduction à la réalité virtuelle : caméra en survol dans une scène 3D

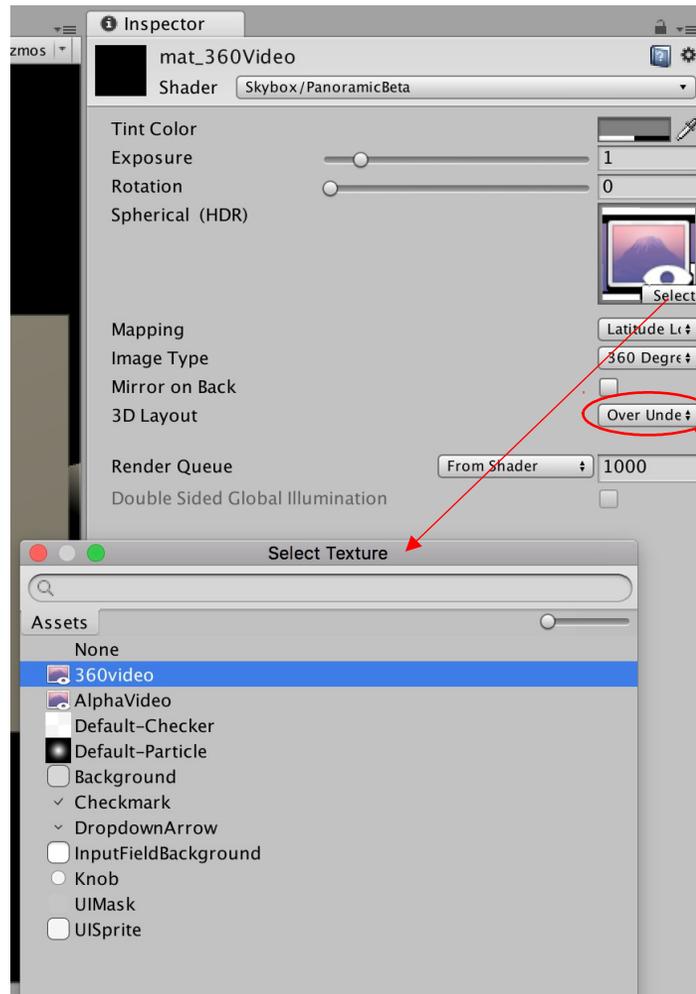
3) Intégrer la vidéo à la scène

- Ajouter la vidéo 3D-360 dans un dossier « Video » du dossier « Assets » du projet.
- Glisser le « video clip » dans la hiérarchie
- Changer le « Render Mode » du Component « Video Player » à « Render Texture »



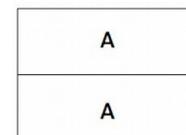
Introduction à la réalité virtuelle : caméra en survol dans une scène 3D

4) Utiliser le Shader dans un Material pour lire la vidéo dans le Skybox de la scène

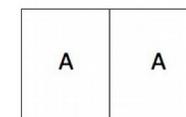


- Créer un asset « Material » appelé « mat_360Video »
- Sélectionner le shader « Skybox/Panoramic Beta » que nous avons ajouté à nos assets
- Sélectionner le « Render Texture » nommée « 360video » dans le champ « Spherical (HDR) »

/!\ Dans le cas d'une vidéo 360 stéréoscopique, choisi le type de 3D layout de votre video



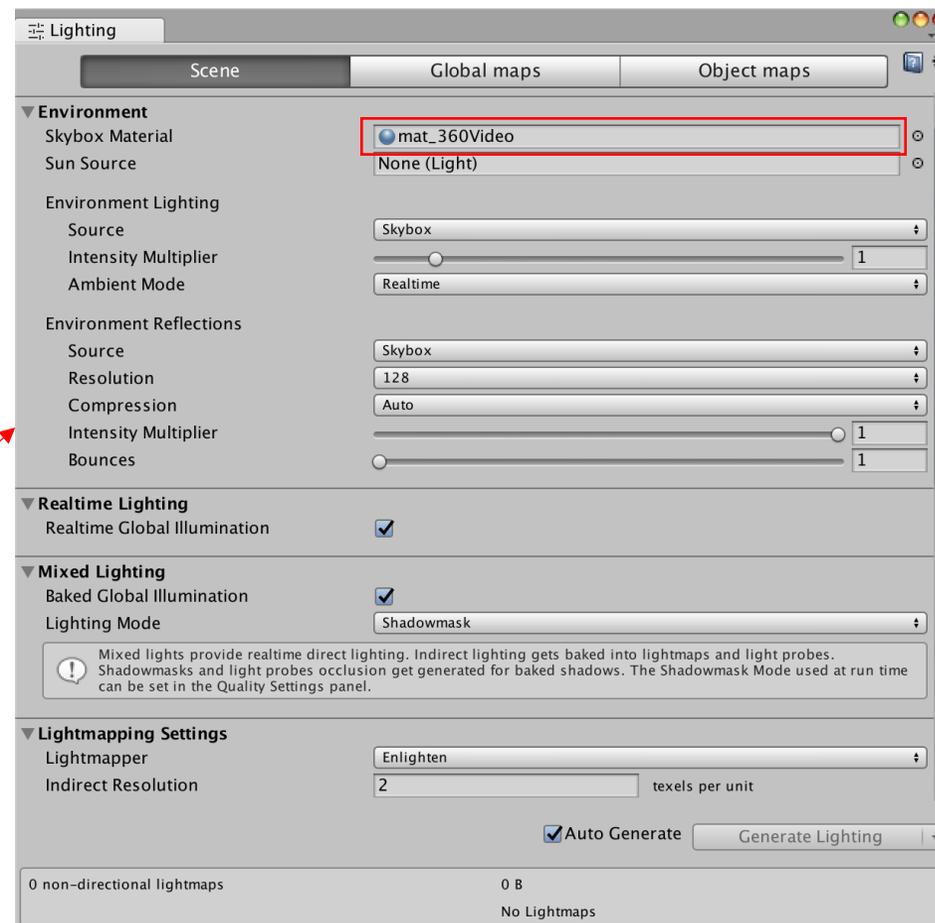
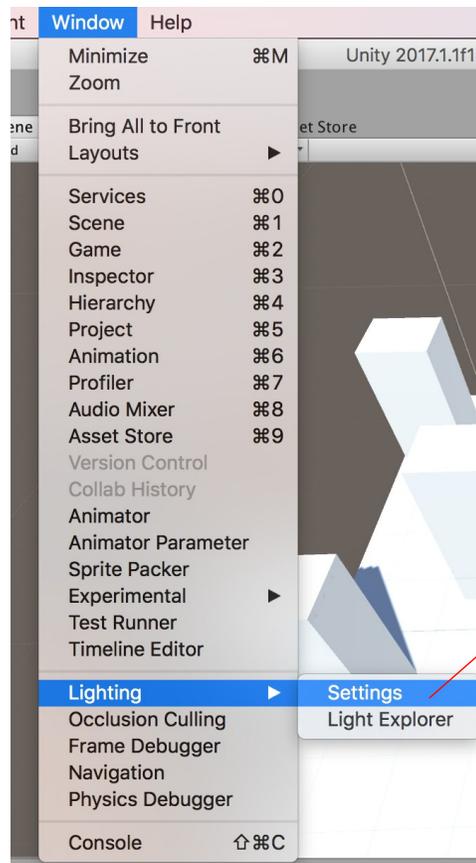
Over Under



Side by Side

Introduction à la réalité virtuelle : caméra en survol dans une scène 3D

5) Associer le « skybox » de Unity à notre Material « mat_360Video »



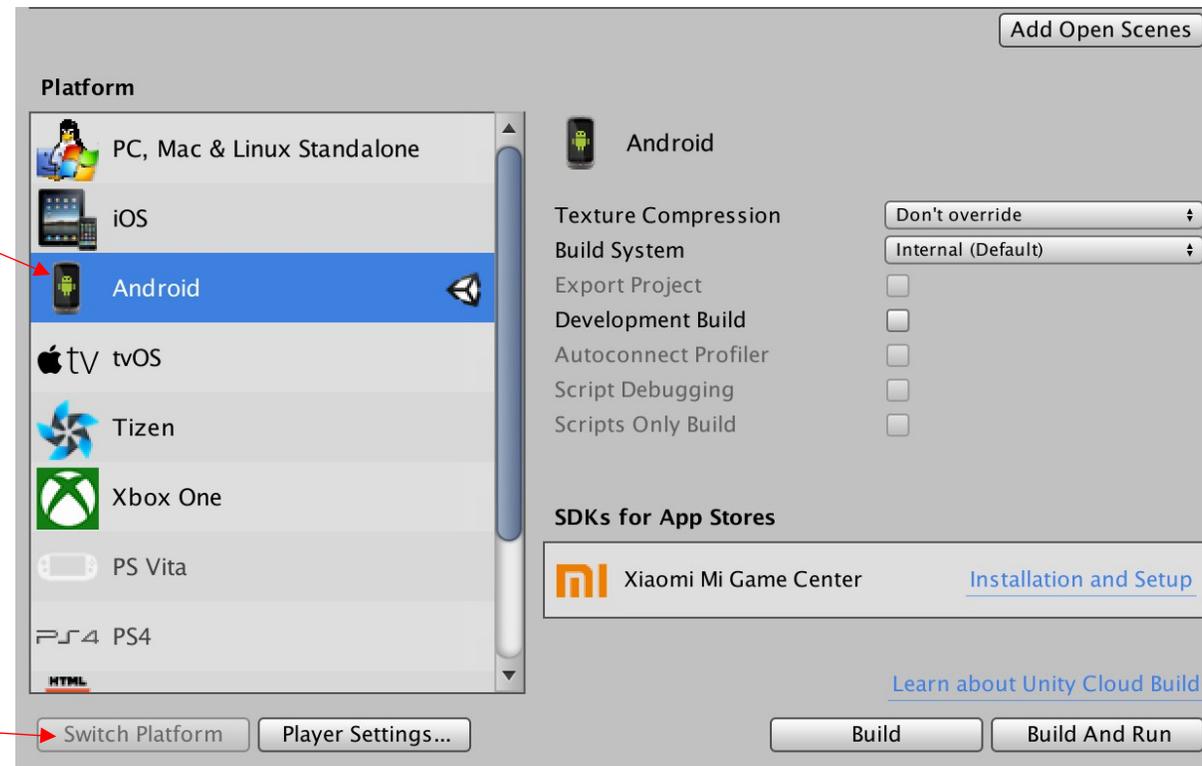
Introduction à la réalité virtuelle : caméra en survol dans une scène 3D

6) La compilation pour Android – Gear VR et Google Daydream – **avec Mac OS**

File > Build Settings

Sélectionner la
plateforme Android

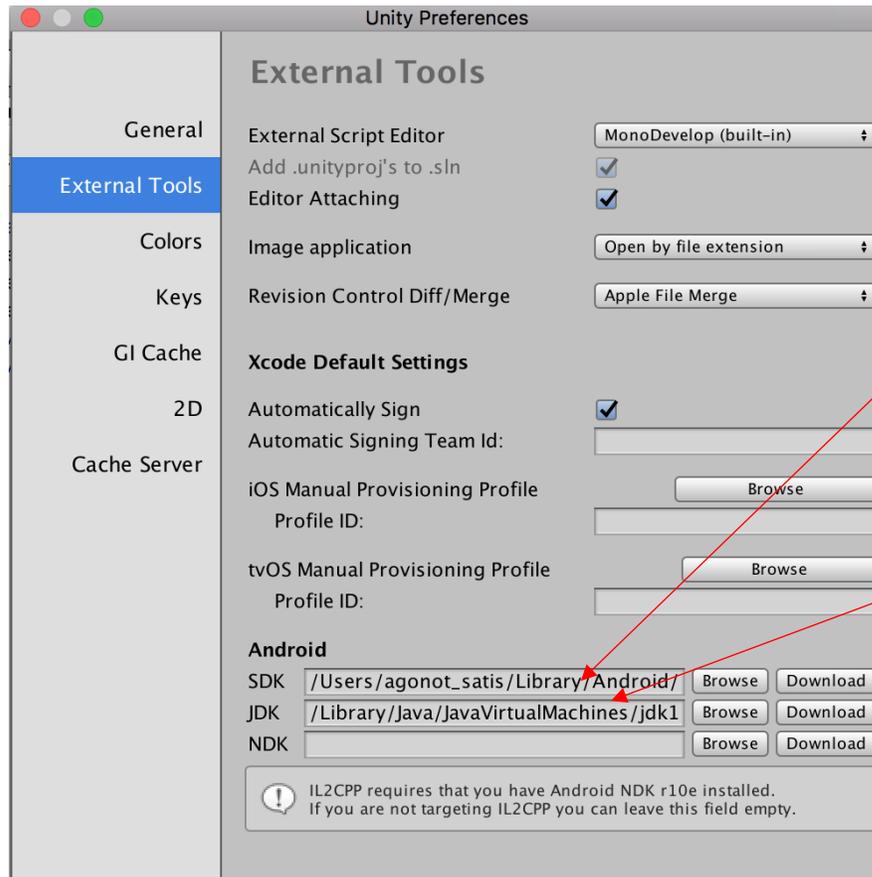
Switch Platform



Introduction à la réalité virtuelle : caméra en survol dans une scène 3D

6) La compilation pour Android – Gear VR et Google Daydream – avec Mac OS

Unity > Preferences



Indiquer le chemin du SDK Android

`/Users/$Home/Library/Android/sdk`

Indiquer le chemin du Java Development Kit

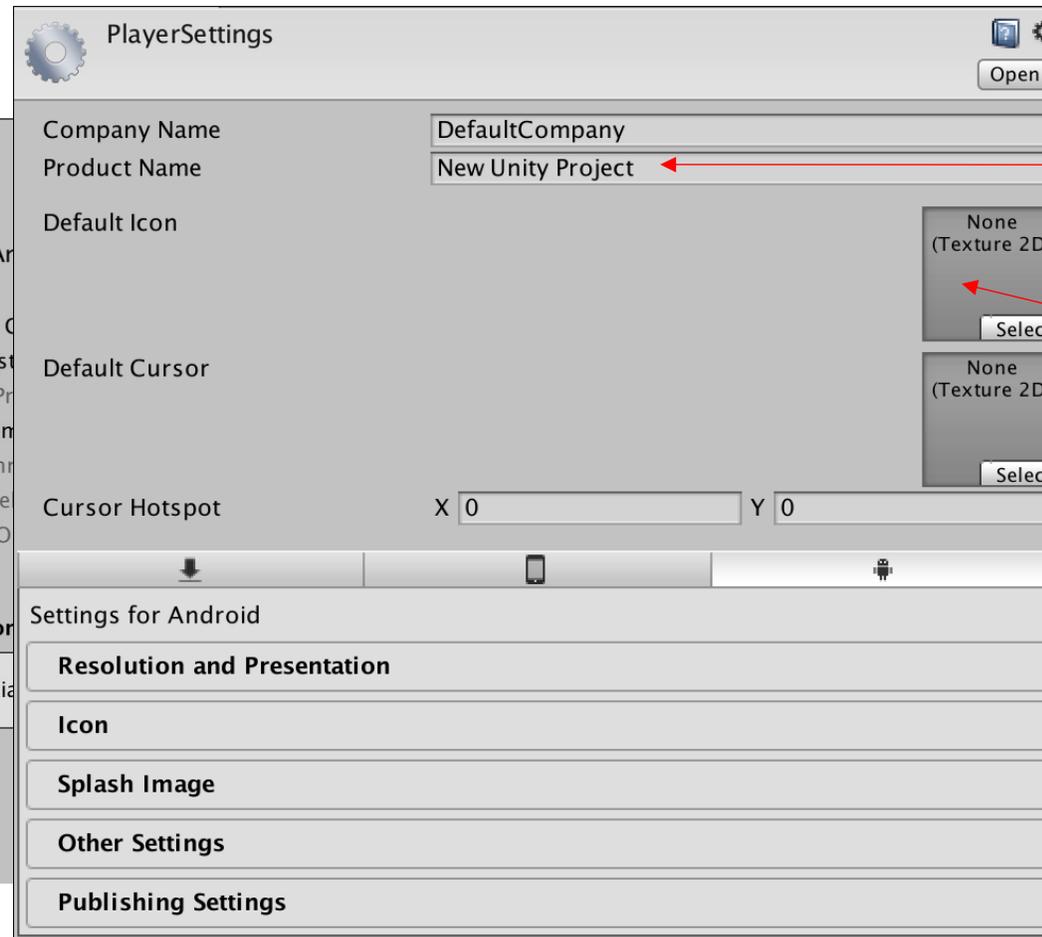
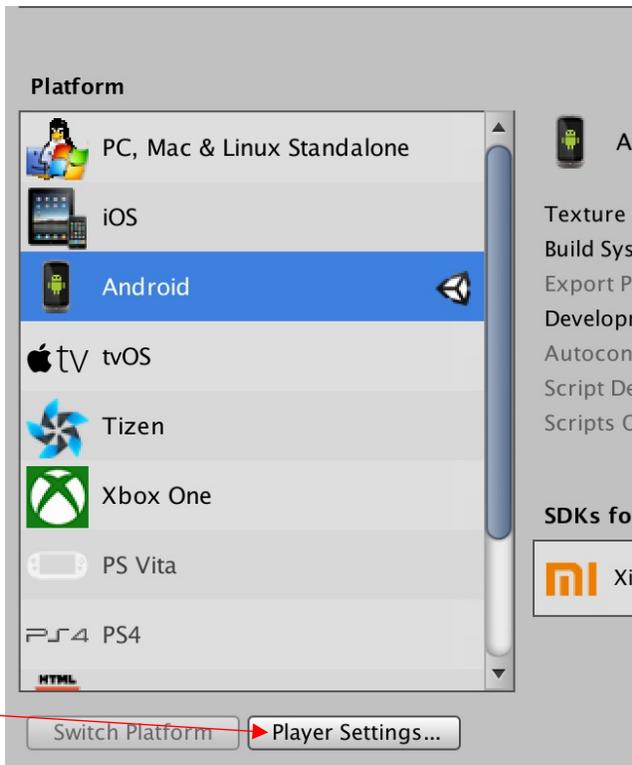
`/Library/Java/JavaVirtualMachines/jdk1.8.0_151.jdk/Contents/Home`

! IL2CPP requires that you have Android NDK r10e installed.
If you are not targeting IL2CPP you can leave this field empty.

Introduction à la réalité virtuelle : caméra en survol dans une scène 3D

6) La compilation pour Android – Gear VR et Google Daydream – avec Mac OS

File > Build Settings



Donner un nom à l'application
Ex. FlyingVR

Ajouter une icône

Switch Platform

Introduction à la réalité virtuelle : caméra en survol dans une scène 3D

6) La compilation pour Android – Gear VR et Google Daydream – avec Mac OS

File > Build Settings

Platform

- PC, Mac & Linux Standalone
- iOS
- Android**
- tvOS
- Tizen
- Xbox One
- PS Vita
- PS4
- HTML5

Other Settings

Rendering

- Color Space* Gamma
- Auto Graphics API
- Multithreaded Rendering*
- Static Batching
- Dynamic Batching
- GPU Skinning*
- Graphics Jobs (Experimental)*
- Virtual Reality Supported
- Protect Graphics Memory

Identification

- Package Name com.Company.ProductName
- Version* 1.0
- Bundle Version Code 1
- Minimum API Level Android 4.1 'Jelly Bean' (API level 16)
- Target API Level Automatic (highest installed)

Modifier le « Package Name »
Ex. com.Satis.FlyingVR

Sélectionner la version minimale d'Android

Switch Platform

Switch Platform Player Settings...

Introduction à la réalité virtuelle : caméra en survol dans une scène 3D

6) La compilation pour Android – Gear VR et Google Daydream – avec Mac OS

File > Build Settings

Platform

- PC, Mac & Linux Standalone
- iOS
- Android**
- tvOS
- Tizen
- Xbox One
- PS Vita
- PS4
- HTML5

Other Settings

Rendering

- Color Space* Gamma
- Auto Graphics API
- Multithreaded Rendering*
- Static Batching
- Dynamic Batching
- GPU Skinning*
- Graphics Jobs (Experimental)*
- Virtual Reality Supported**

Virtual Reality SDKs
List is Empty

Must add at least one Virtual Reality SDK

Stereo Rendering Method* Multi Pass

Protect Graphics Memory

Cocher l'option « Virtual Reality Supported »

Sélectionner le SDK selon le type de casque cible (Daydream ou Oculus)

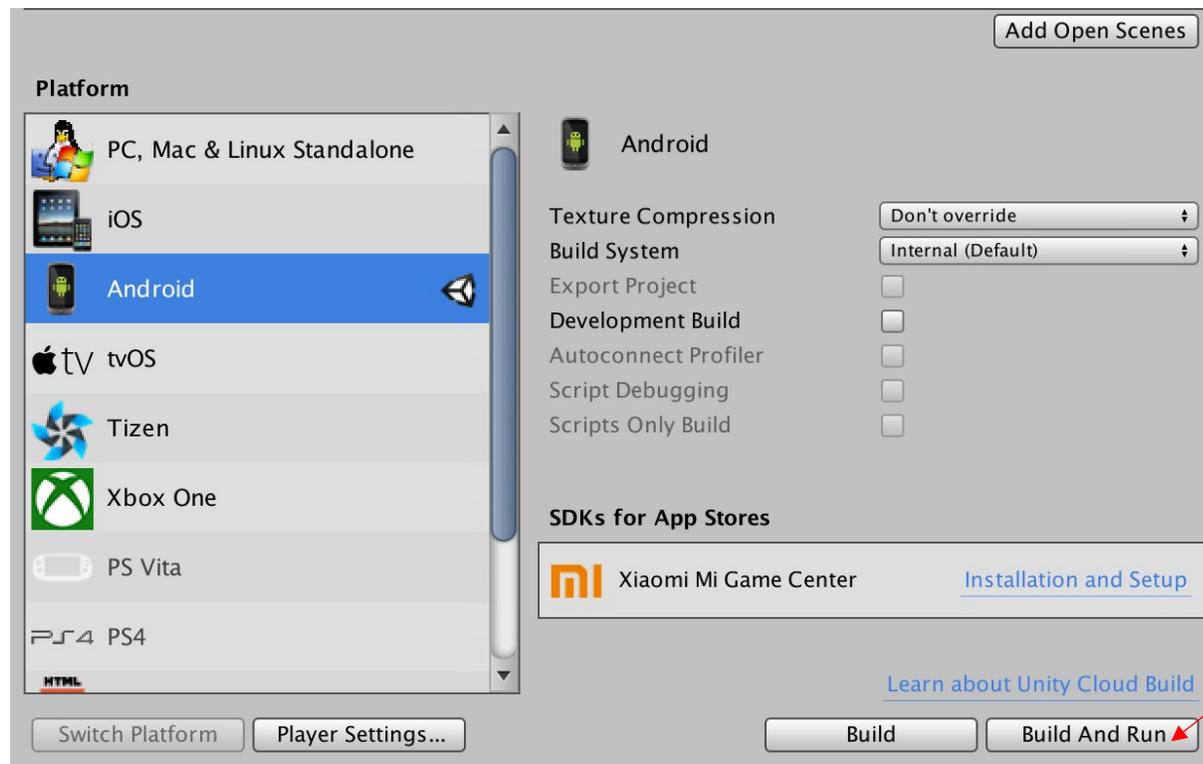
Cardboard
Daydream
None
Oculus
Split Stereo Display (non head-mounted)

Switch Platform

Introduction à la réalité virtuelle : caméra en survol dans une scène 3D

6) La compilation pour Android – Gear VR et Google Daydream – avec Mac OS

File > Build Settings



Connecter le smartphone sur le port USB

Lancer la compilation de l'exécutable (*.apk), son installation et son exécution

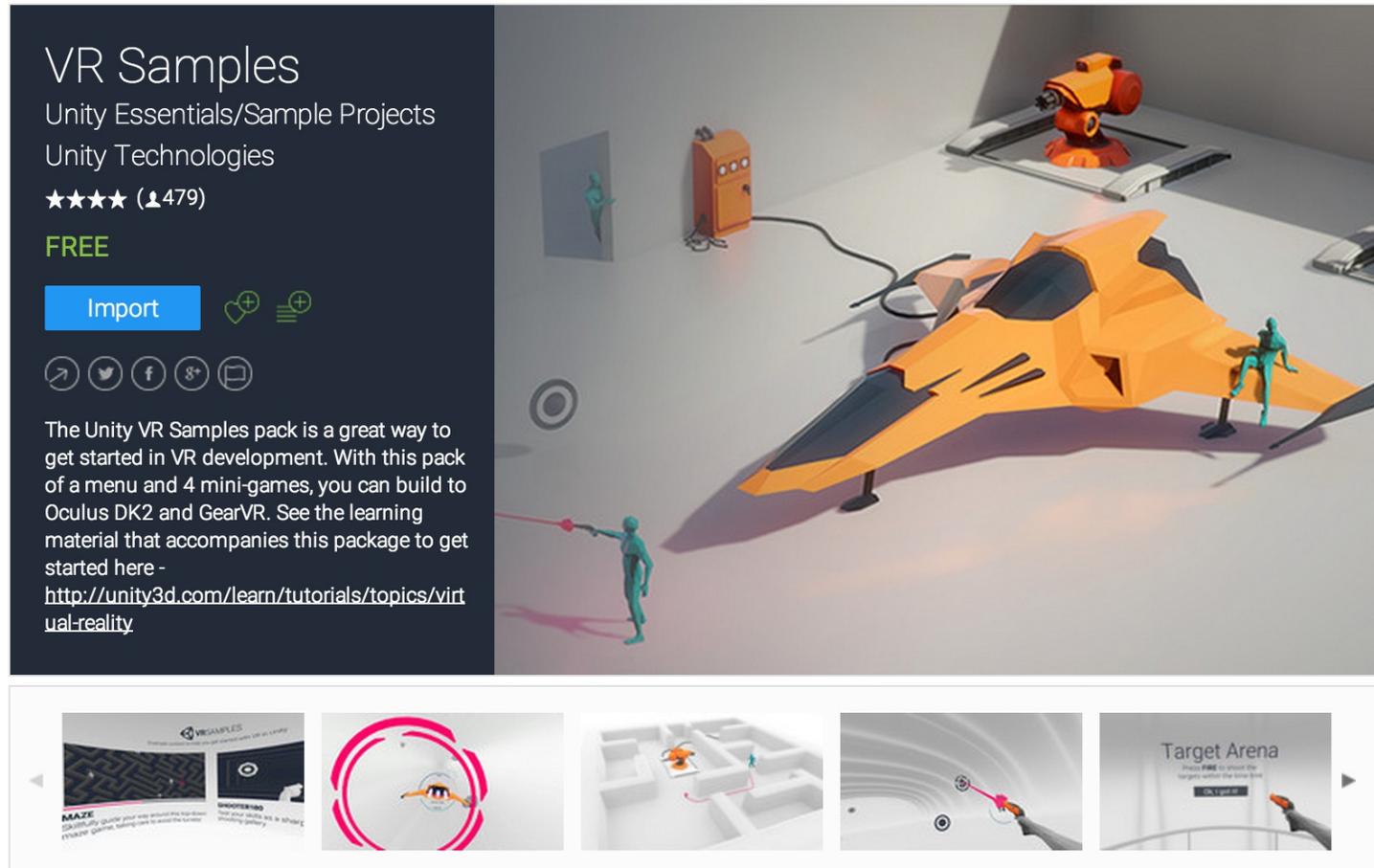
Interaction Utilisateur avec un casque de Réalité Virtuelle

« Gaze Interaction » et « Controler »



L'interaction Utilisateur avec un casque de réalité virtuelle

1) Télécharger « VR Samples » dans le Unity Store



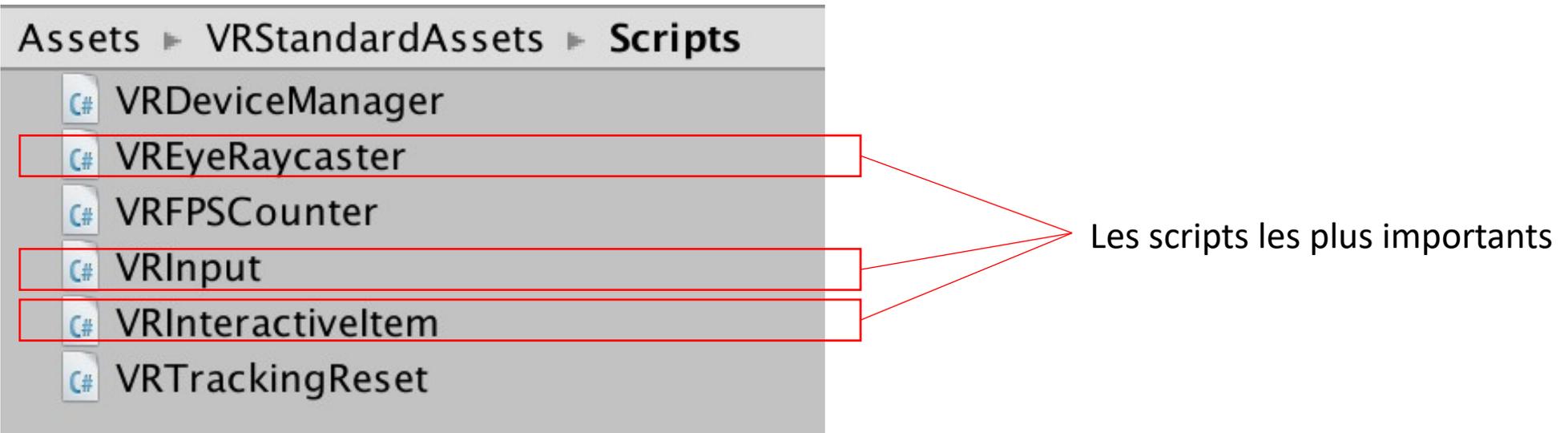
VR Samples
Unity Essentials/Sample Projects
Unity Technologies
★★★★ (1479)
FREE
Import

The Unity VR Samples pack is a great way to get started in VR development. With this pack of a menu and 4 mini-games, you can build to Oculus DK2 and GearVR. See the learning material that accompanies this package to get started here - <http://unity3d.com/learn/tutorials/topics/virtual-reality>

The main image shows a 3D scene with a large orange jet, a small orange robot, and a teal character. Below the main image is a carousel of thumbnails for the included mini-games: MAZE, SHOOTER180, a circular arena, and Target Arena.

L'interaction Utilisateur avec un casque de réalité virtuelle

1) Télécharger « VR Samples » dans le Unity Store



/!\ Les scripts ne permettent pas de prendre en compte les actions sur le « Controler » de Google (Uniquement celui du Gear VR d'Oculus)

-> Modification des scripts « VREyecaster.cs », « VRInput.cs » et « VRInteractiveltem.cs »

L'interaction Utilisateur avec un casque de réalité virtuelle

2) Télécharger [GoogleVRForUnity_1.130.1.unitypackage](#)

Pour la prise en charge (entre autres) de l'interaction avec le Contrôleur du Google Daydream

<https://github.com/googlevr/gvr-unity-sdk/releases>

GVR SDK for Unity v1.130.1

 rusmaxham released this 24 days ago

Assets

 GoogleVRForUnity_1.130.1.unitypackage	34.4 MB
 GoogleVRForUnity_1.130.1_source.zip	34.3 MB
 Source code (zip)	
 Source code (tar.gz)	

Bug fixes

* Fixed build error ([issue 877](#)) when Unity iOS player support is not installed.

L'interaction Utilisateur avec un casque de réalité virtuelle

3) Le contenu de « VR Sample »

Assets ▶ VRSampleScenes ▶ Scenes ▶ **Examples**

 Interactiveltem

 RenderScale

 Reticle

 Rotation

 Touchpad

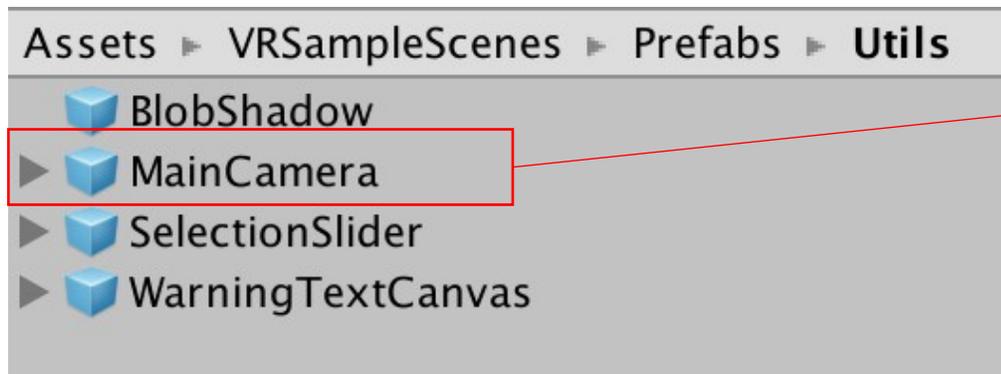
L'exemple qui nous intéresse

- Détecter l'objet dans l'axe du regards ET/OU
- Détecter l'appui sur le bouton du Controler

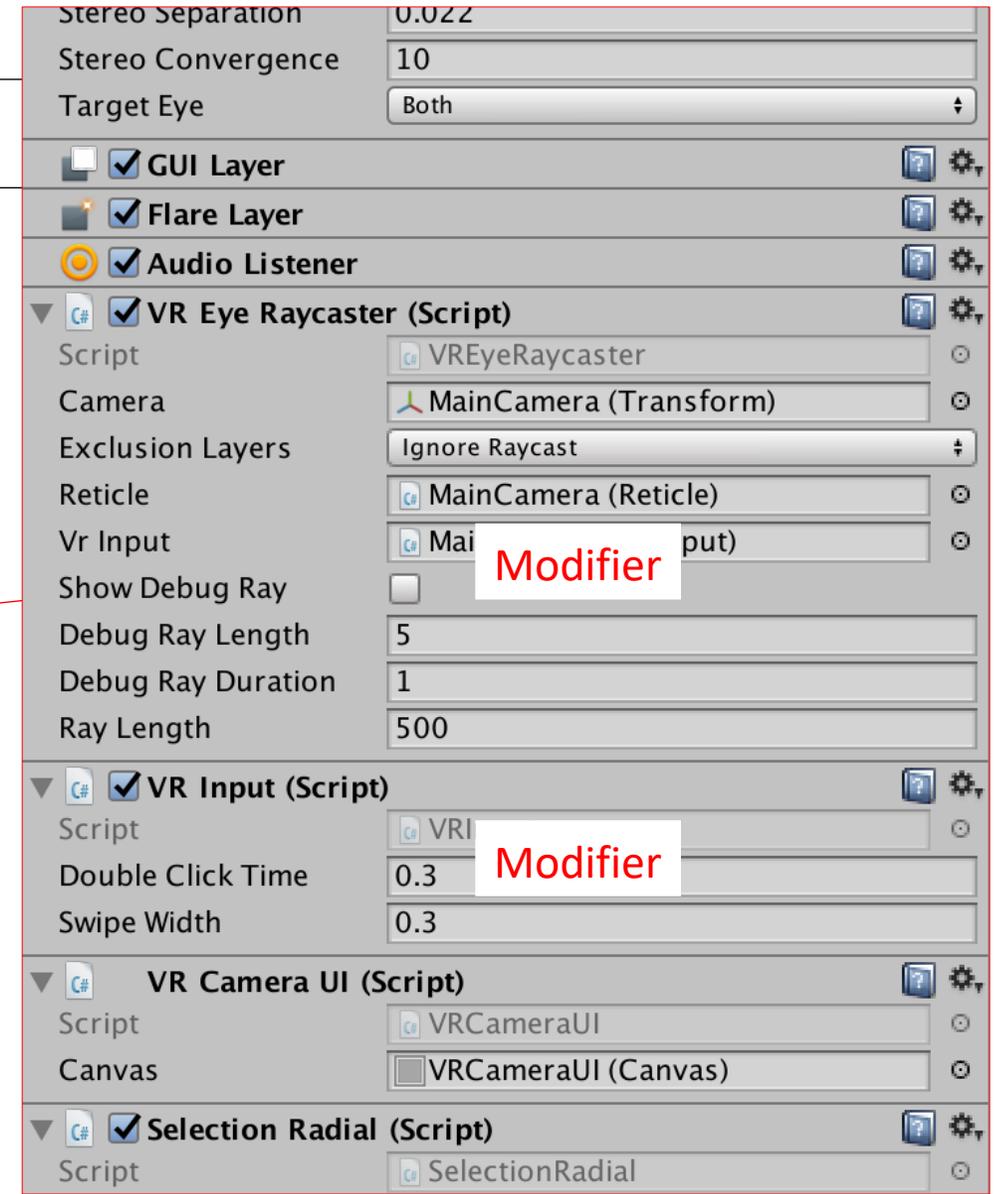
L'interaction Utilisateur avec un casque de réalité virtuelle

4) Interactive Item

- Créer une nouvelle scène « menu »
- Supprimer la camera par défaut et ajouter le Prefab « MainCamera » de « VR Samples »



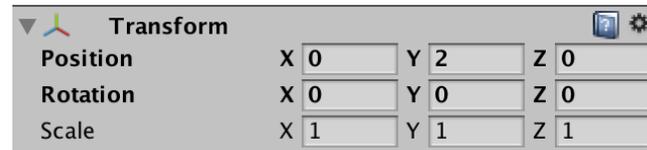
La camera contient les scripts « VREyecaster.cs » et « VRInput.cs » qu'il faut modifier



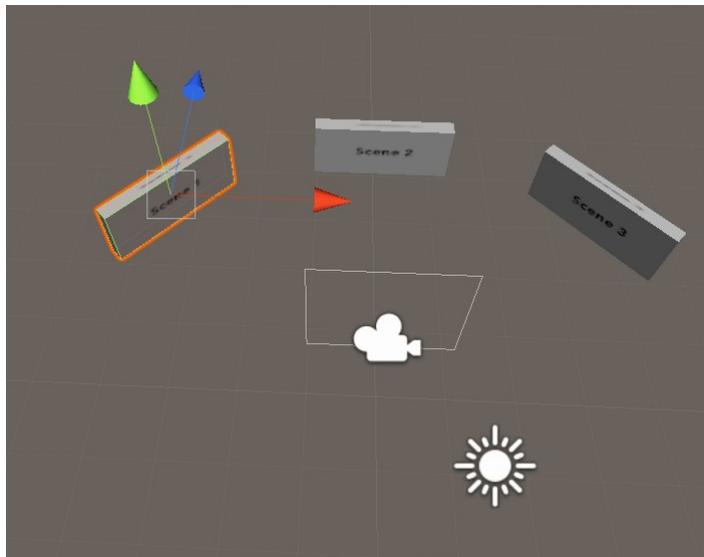
L'interaction Utilisateur avec un casque de réalité virtuelle

4) Interactive Item

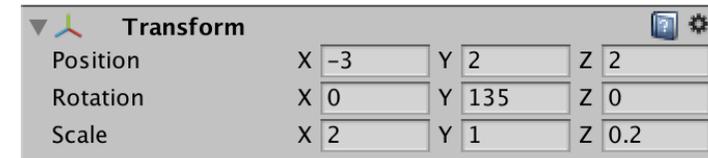
- Positionner la Camera à



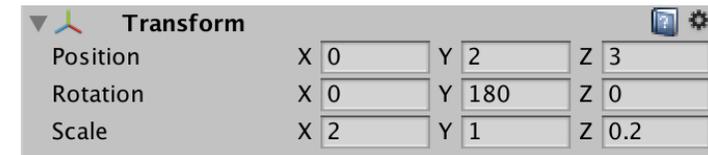
- A l'aide de 3 cubes, créer 3 objets nommés « Menulitem1 », « Menulitem2 », « Menulitem3 » et les placer autour de la caméra



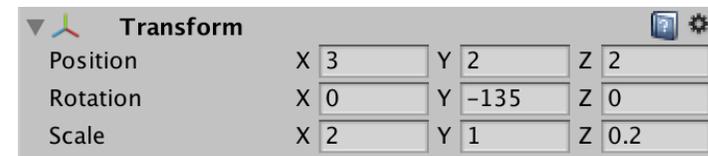
Menulitem1



Menulitem2



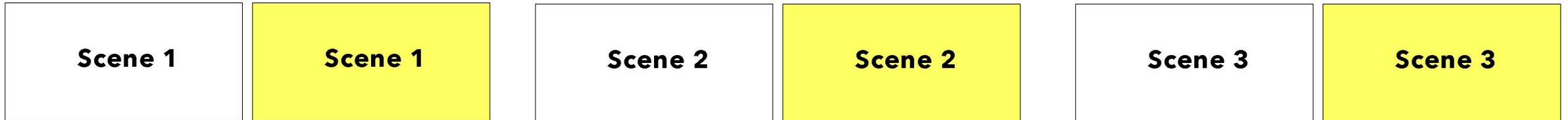
Menulitem3



L'interaction Utilisateur avec un casque de réalité virtuelle

4) Interactive Item

- Copier les 6 textures dans le dossier « Assets »

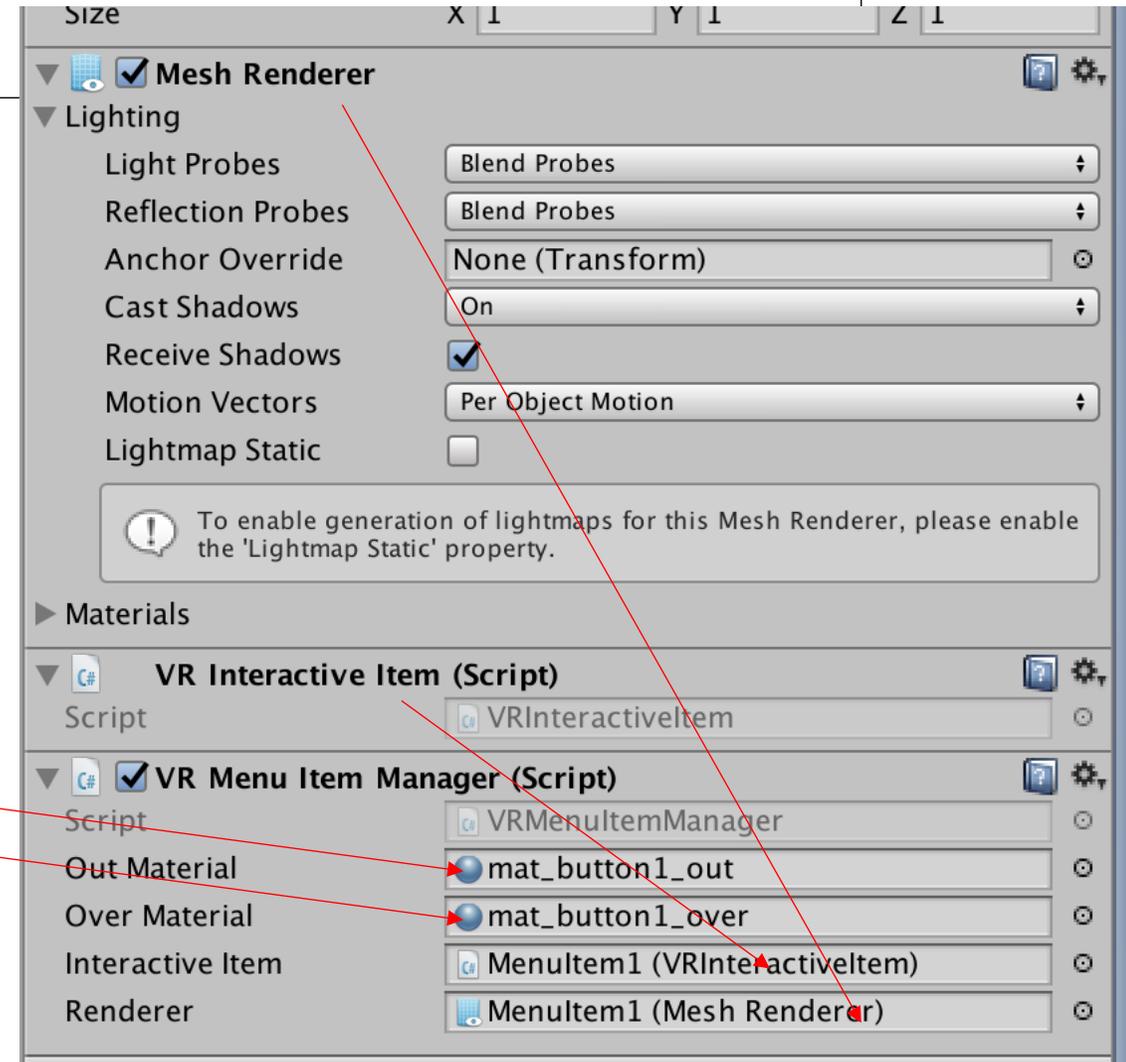
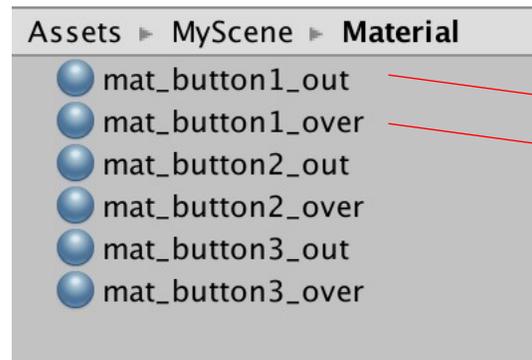


- Créer un Material pour chacune de ces textures
- Associer le Material « Off » pour chaque objet « MenuItem »

L'interaction Utilisateur avec un casque de réalité virtuelle

4) Interactive Item

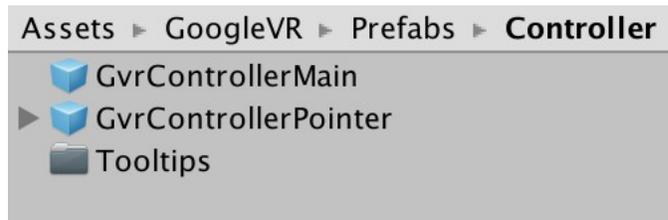
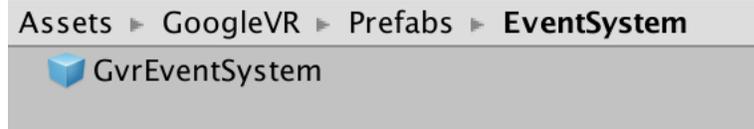
- Copier le script « VRMenuItemManager.cs » dans le dossier Assets
- Ajouter les scripts « VRInteractiveltem » et « VRMenuItemManager » à chaque objet « MenuItem »
- Instancier les variables public des Components « VRMenuItemManager » de chaque objet



L'interaction Utilisateur avec un casque de réalité virtuelle

5) Gestion du Controller Google Daydream

- Ajouter les prefabs GvrControllerMain et GvrEventSystem dans la scène



L'interaction Utilisateur avec un casque de réalité virtuelle

6) Compilation

- Une compilation avec le SDK – Daydream : un fichier *.apk
-> Installation
- Une seconde compilation avec le SDK – Oculus : un autre fichier *.apk
-> Installation