



Introduction à Unity3D

Damien Marchal



C:\whoami

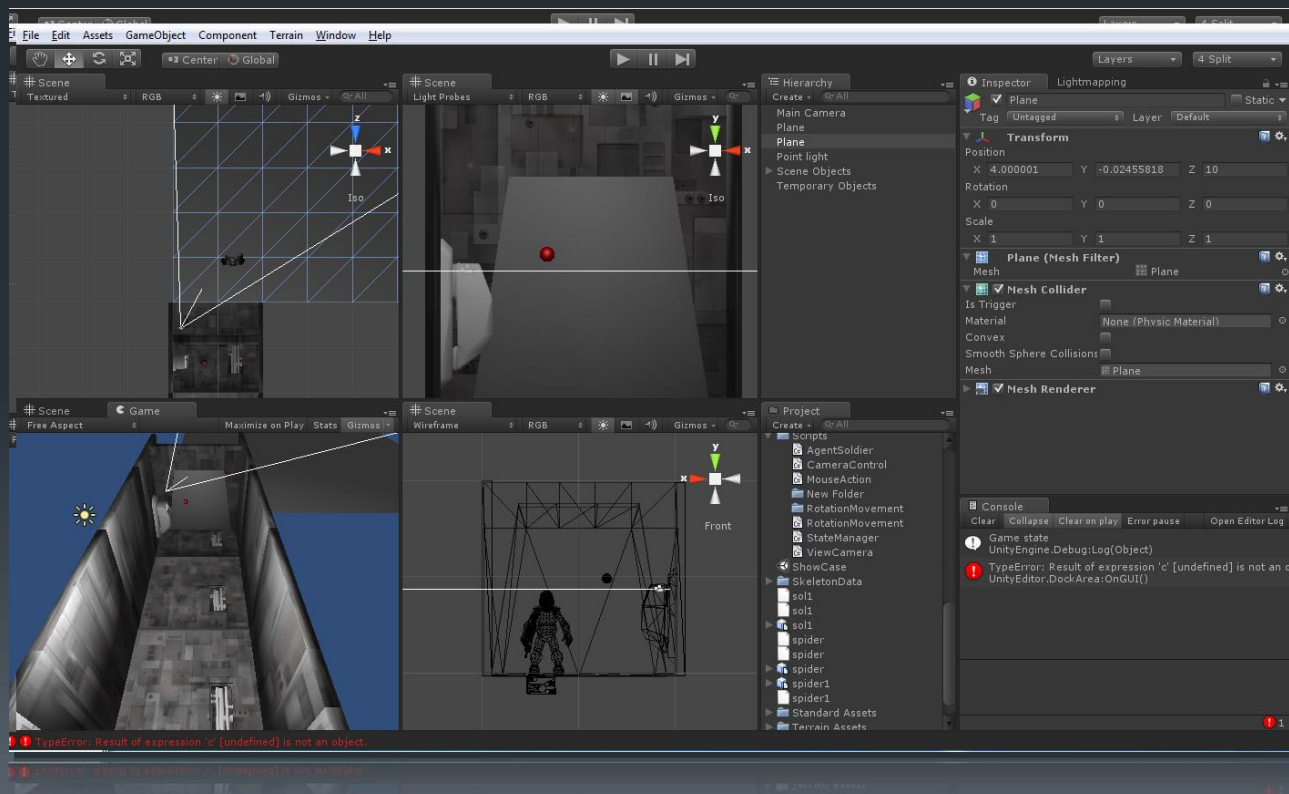
- Damien Marchal: damien.marchal@lifl.fr
- Ingénieur au CNRS dans le domaine de l'interaction homme-machine et de la simulation temps-réel (équipes MINT et Shacra).
- Je fait des prototypes d'applications de RV et d'interaction 3D en utilisant Unity;
- J'interviens depuis 2 ans dans le Master IVI.
 - <http://master-ivi.univ-lille1.fr/>
 - <http://www.lifl.fr/~marchal/rvi/tp.html>



Nomenclature

- Les chemins dans les menus seront notés de la manière suivante: [windows:inspector].
- Les interactions à la souris seront notées de la manière suivante:
 - [LMB]: left mouse button,
 - [RMB]: right mouse button,
 - [LMB+V]: indique un clic avec le bouton gauche de la souris et un appui sur la touche V.

Unity





Unity

- Unity est un logiciel auteur d'application interactive 3D comprenant:
 - Un éditeur de scène (MacOS et Windows)
 - Un moteur de jeu multiplateformes (Windows, MacOS, Android, iOS, XBox360, Wii, PS3, Web browser)
 - Des outils annexes:
 - Unity Asset Server: serveur centralisé des données.
 - Unity Store: pour downloader des assets et du code.
 - Une communauté forte



Unity - engine

- Rendu 3D temps-réel avec :
 - Shaders écrits en Cg ou GLSL, Ombres douces,
 - SSAO, HDR, Surface Shader, Post-processing,
 - Lightmaps, Allegorithmic Substance.
 - Import: *Maya, 3DS Max, Cheeta, Cinema 4D, Blender, Modo, Collada, Carrara, Lightwave, XSI, Sketchup, Wings 3D,*
- Pour l'animation il y a un support des:
 - Images clés,
 - Squelettes,
 - Système Particules et simulation physique.



Unity - engine

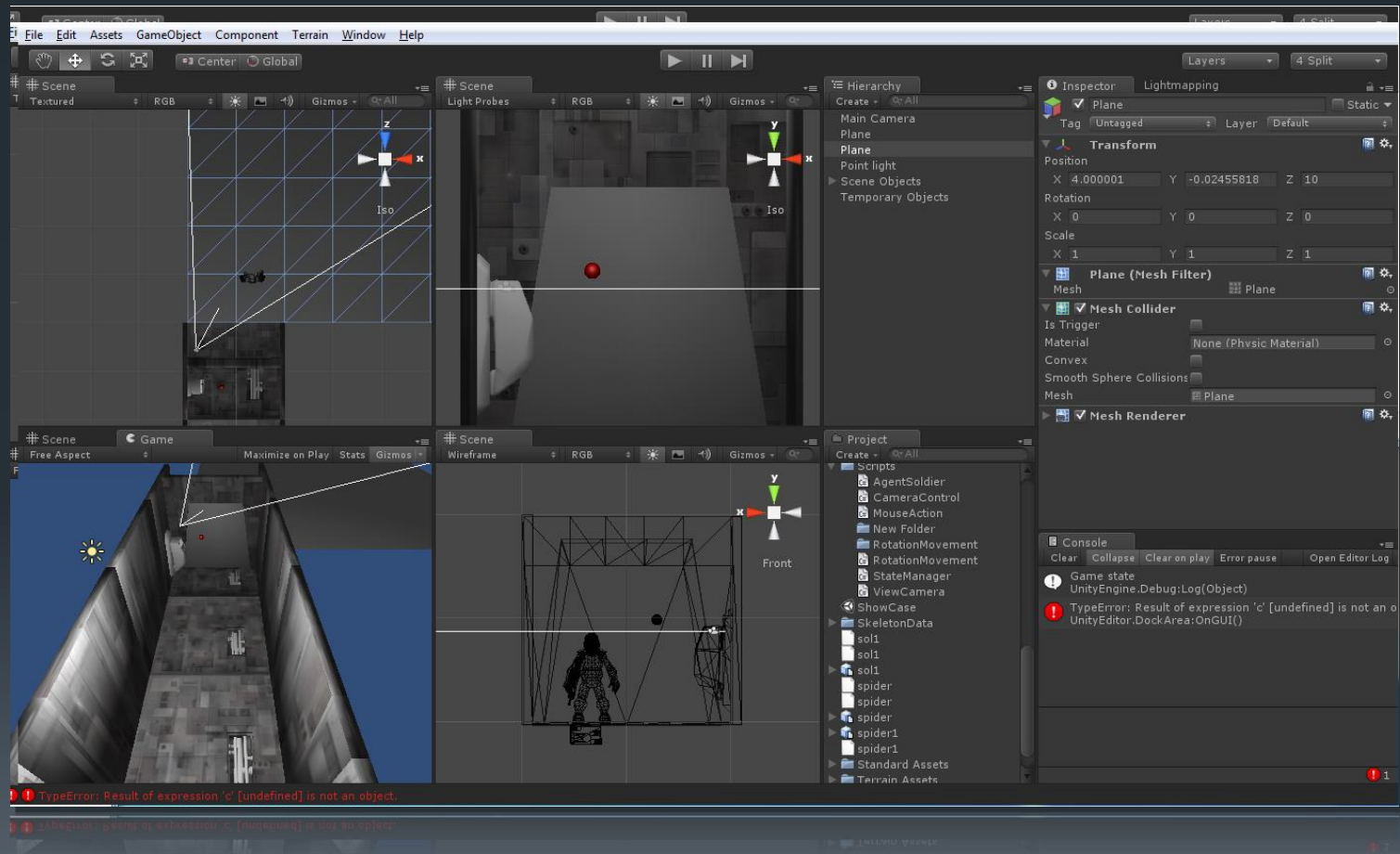
- Plusieurs langages de programmation :
 - C#,
 - UnityScript, un dérivé de Javascript (avec typage),
 - Boo, un dérivé de python (avec typage),
 - Utilisation possible de bibliothèques écrites en C/C++ (support Kinect, table tactiles, etc...),
 - Éditeur de code fourni et profileur intégré.



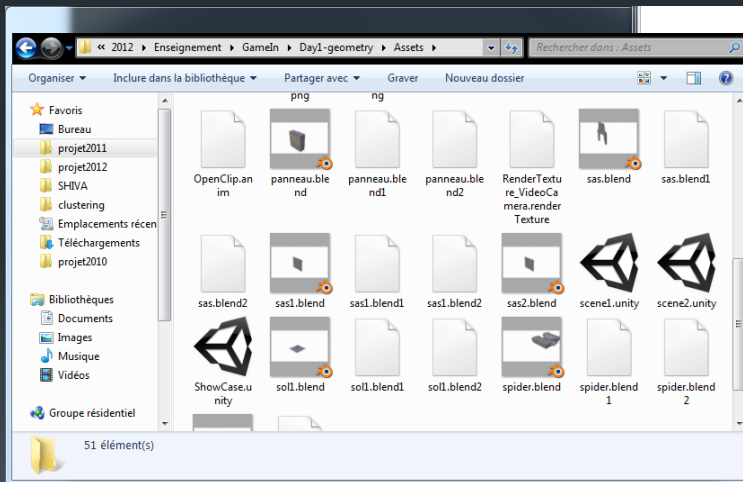
Unity - engine

- Autres aspects :
 - Editeur de terrains,
 - Système de path-finding à base de maillage de navigation,
 - Gestion réseau, synchronisation des objets, RPC, socket,
 - Intégration dans un browser web via un plugin,
 - Son spatialisé.
- L'asset-store permet d'acheter de nombreux modules supplémentaires et parfois « *indispensables* »: iTween,

Unity - éditeur

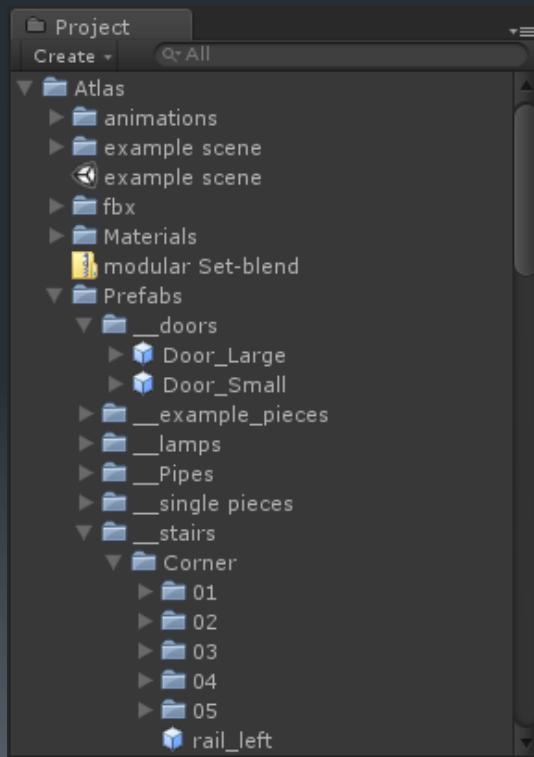


Projet == un répertoire



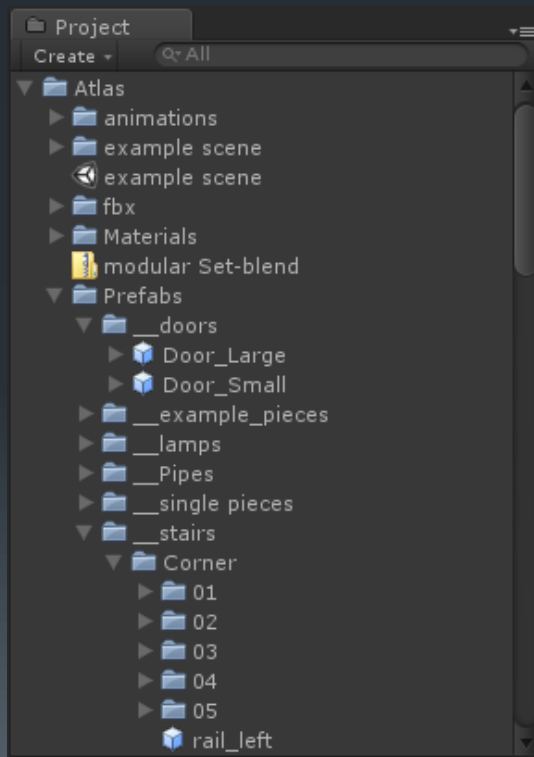
- L'arborescence du répertoire sert à organiser les ressources (assets).
- Tout ce qui est copié dans ce répertoire est importé automatiquement dans Unity.

Projet == un répertoire



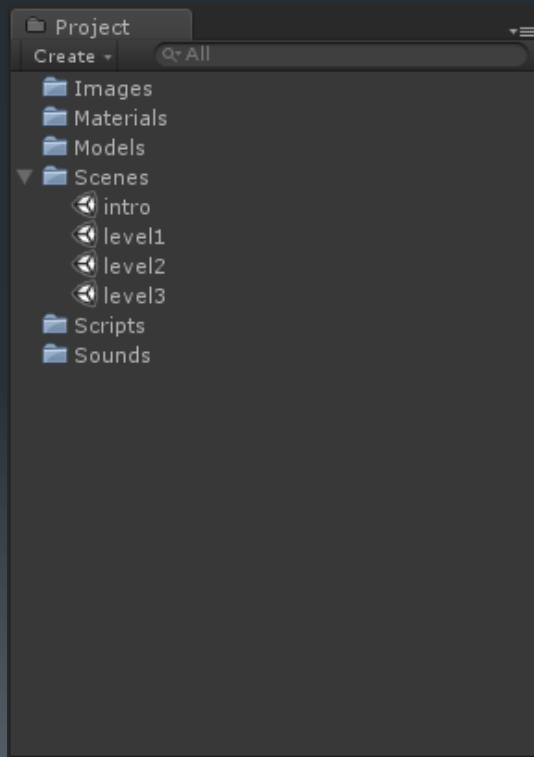
- Le contenu du répertoire est visible dans la fenêtre Project de Unity.
- C'est en combinant les ressources qu'on construit une scène.
- Il faut utiliser ce panneau pour déplacer ou renommer les ressources (et pas l'explorateur de fichier).
- [LMB], [RMB]

Projet == un répertoire



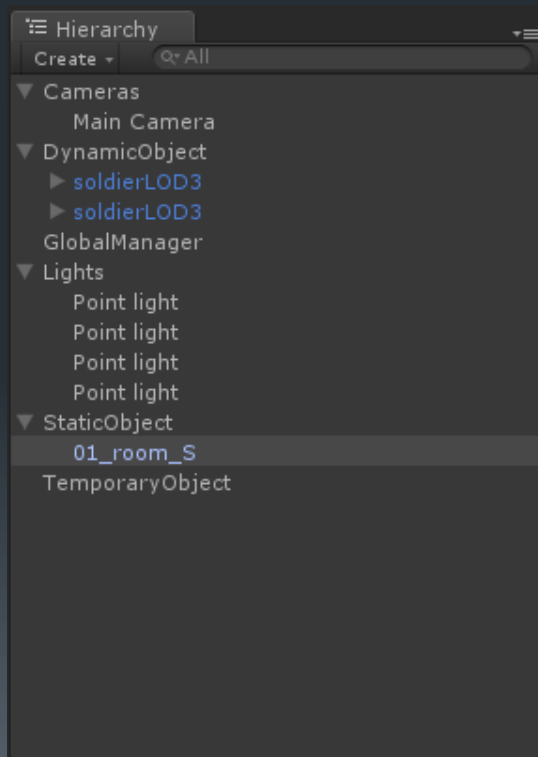
- Drag'n Drop des données dans la scène.
- Les données deviennent alors des `GameObject`.

Une scène



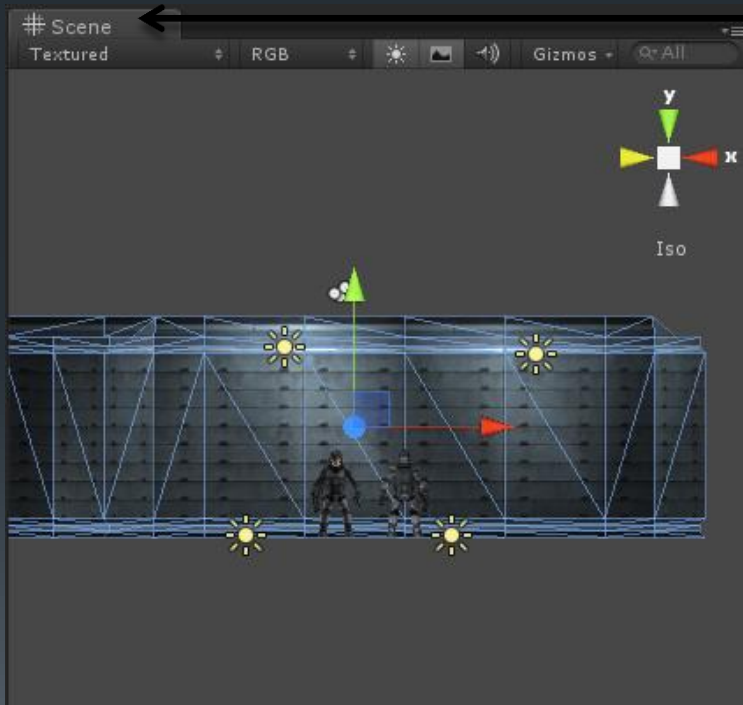
- C'est un assemblage donné des ressources.
- Découpage en niveaux ou en en phases (menus, jeu, cinématiques).
- Par script on passe d'une scène à l'autre.
- Pour créer une scène [File:Save scene as...].

Une scène – vue Hierarchy



- Hiérarchie d'objets.
- Chaque nœud correspond à un niveau de transformation 3D.
- Des nœuds vides (empty) sont utilisés pour structurer les données.

Une scène – vue 3D



- Vues 3D.
- Manipulation directe pour l'édition de scène.
- Drag&Drop à partir du panneau Project pour ajouter des objets.

Une scène – vue Game



- Prévisualisation du jeu (vue par la caméra).
- Affiche la scène vue par la caméra principale.
- Pas d'édition.



Une Scène – plusieurs objets

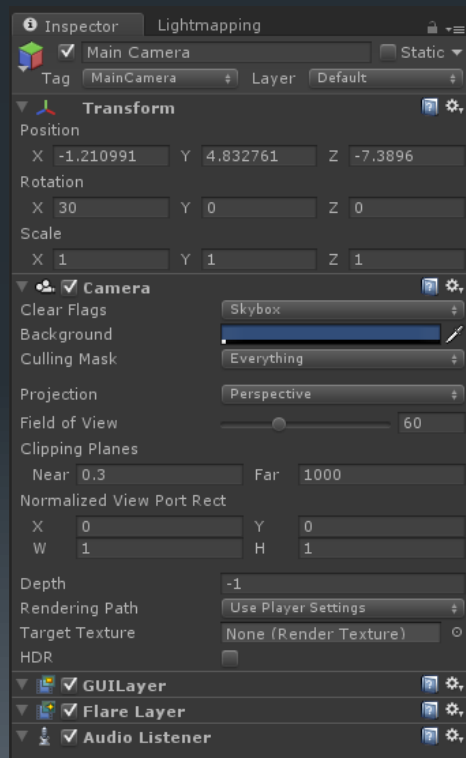
- Dans unity les objets d'une scène sont appelés GameObject.
- Pour instancier de nouveaux GameObjects dans une scène:
 - Menu [GameObject:Create Empty],
 - Menu [GameObject:Create Other:...],
 - Drag&Drop des assets du panneau Project,
 - Copié-collé d'objets existants.



GameObject et composants

- Par défaut un GameObject ne fait pas grand-chose.
- Il faut lui rajouter des composants. Exemples :
 - MeshFilter + MeshRenderer
 - Scripts, Collider,
 - GUILayer, Flare Layer, AudioListener,
 - Camera, Light.
- Voir menu [Component].

Panneau Inspector



- Affichage contextuel qui donne accès aux éléments d'un objet sélectionné.
- Qui donne accès aux éléments d'une ressource.
- Les différents éléments et paramètres peuvent être modifiés:
 - par une fenêtre de sélection
 - par glisser-déposer

Panneau Inspector

Active/Désactive

Spécifie un mot
clés associé
à l'objet.

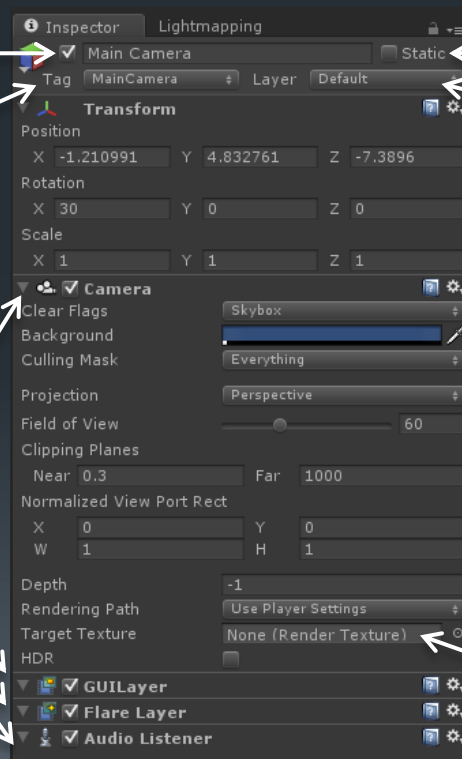
Composants

Indique si l'objet est statique.

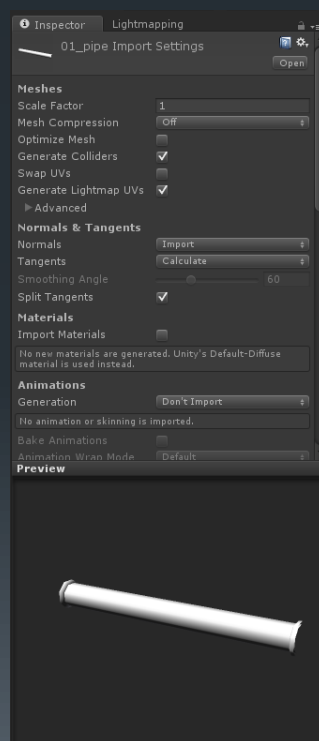
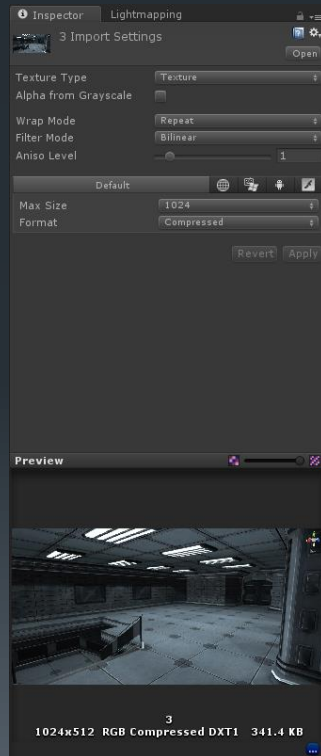
Spécifie si l'objet appartient à un
calque particulier. Les calques sont utilisés
pour le rendu, l'éclairage et la collision.

Ouvre une fenêtre de sélection.

Glisser-déposer d'un objet de la scène ou
d'une ressource.



Panneau Inspector



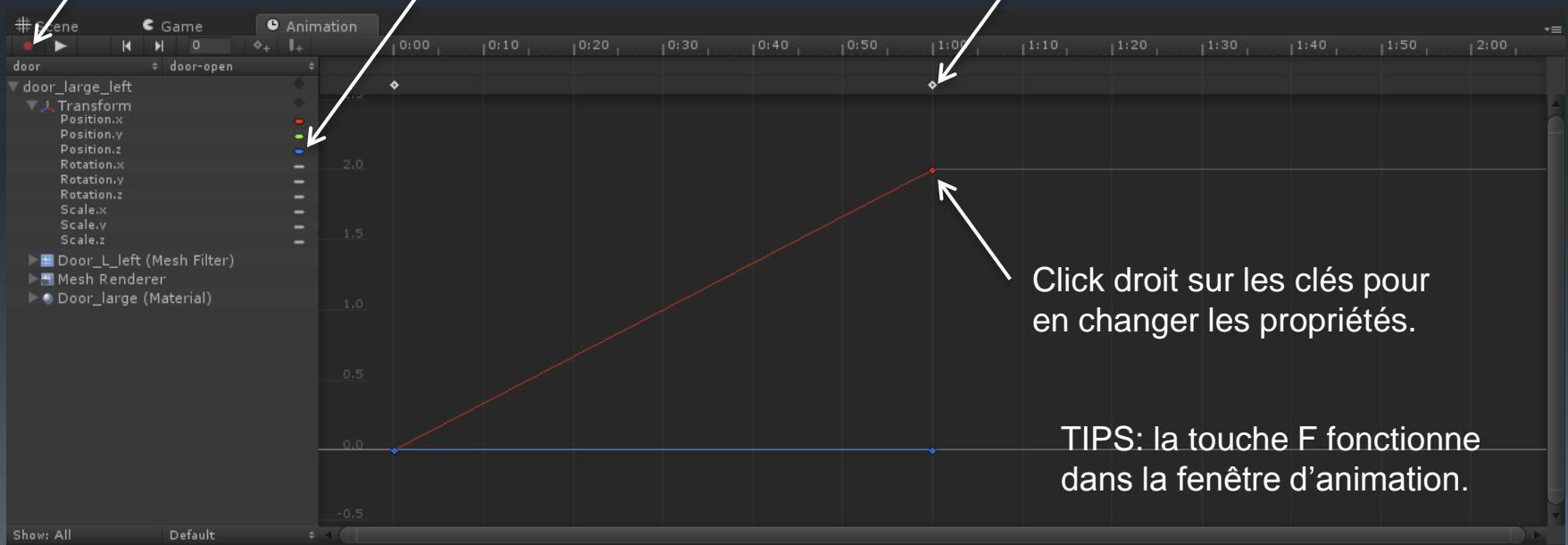
- Lorsque une ressource est sélectionnée, le panneau inspecteur permet d'accéder aux paramètres d'import.

Panneau Animation

Enregistre les propriétés de l'objet.

KeyFrame

Click droit pour ajouter une courbe.



Edition WYSIWYP

- Mode test: —————→
 - « What you see is what you play.
- Tips: On peut toujours changer les paramètres des objets dans les panneaux de l'éditeur...
- mais les modifications sont perdues lorsqu'on quitte le mode de test.





Scripts

Plusieurs langages de programmation :

- C#,
- UnityScript, un dérivé de Javascript (avec typage),
- Boo, un dérivé de python (avec typage),
- Utilisation possible de bibliothèques écrites en C/C++ (support Kinect, table tactiles, etc...),
- Éditeur fourni et profileur intégré.

Scripts

On crée un script dans le panneau Assets.
On le lie à un objet par glissé-déposé.

Un script hérite généralement de la class *MonoBehavior*.

```
using UnityEngine;
using System.Collections;

public class NewBehaviourScript : MonoBehaviour {

    // Use this for initialization
    void Start () {

    }

    // Update is called once per frame
    void Update () {

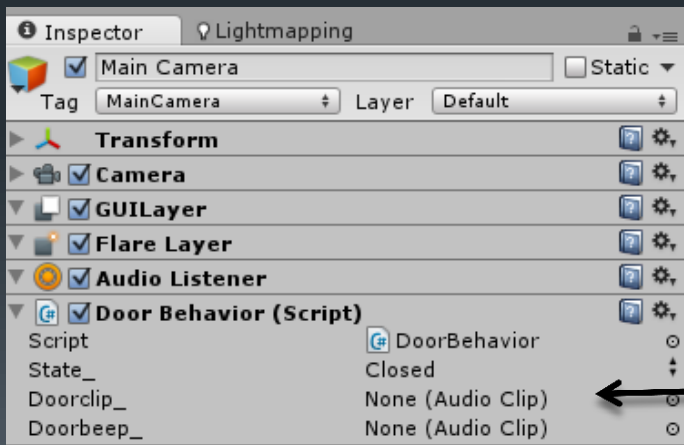
    }

}
```

<http://docs.unity3d.com/Documentation/ScriptReference/MonoBehaviour.html>

Scripts et attributs publiques

Mettre les attributs en public les rend accessibles dans l'Inspecteur,

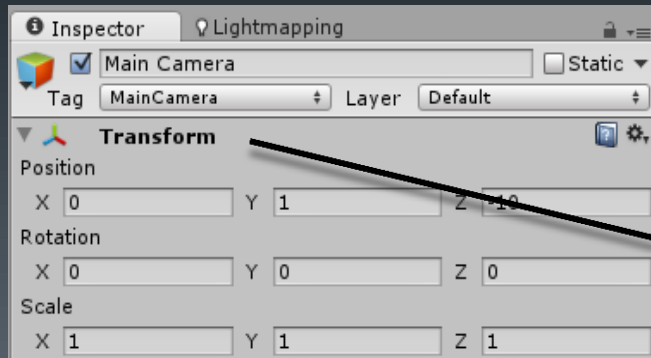


```
1 using UnityEngine;
2 using System.Collections;
3
4 public class DoorBehavior : MonoBehaviour {
5     public enum DoorState
6     {
7         Open,
8         Closed,
9         Closing,
10        Opening
11    };
12
13    public DoorState state_ = DoorState.Closed;
14    public AudioClip doorclip_;
15    public AudioClip doorbeep_;
16 }
```

Scripts et composants

On peut récupérer les différents composants:

- sous la forme d'attributs,
- à l'aide la fonction



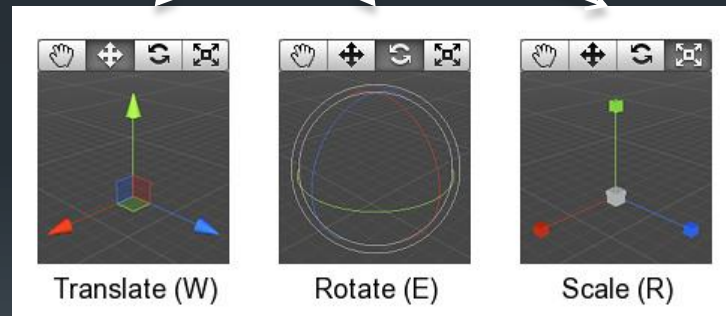
```
1 using UnityEngine;
2 using System.Collections;
3
4 public class GoTo : MonoBehaviour {
5     public GameObject target_;
6     float startTime_;
7     bool started_;
8     Transform tr;
9
10    // Use this for initialization
11    void Start () {
12        Transform tr=GetComponent<Transform>();
13    }
14 }
```

Modélisation

Pour déplacer les GameObjects:



[LMB] sur les widgets de positionnements:



Tips: Les raccourcis clavier correspondant aux transformation suivent l'ordre QWERTY.

Alternative 1: les changer pour AZERTY.

Alternative 2: utiliser les mêmes raccourcis que le logiciel de modélisation.

Navigation

Pour déplacer le point de vue:



Appuyez sur [Q] pour passer en mode Navigation puis:

[DragLBM] pour faire une translation,

[DragLBM+ALT] pour faire tourner la caméra,

[DragLM+CLTR] pour faire un Zoom.

[+MAJ] accélère la vitesse du mouvement de translation et de zoom.

Plus de détails sur:

<http://docs.unity3d.com/Documentation/Manual/SceneViewNavigation.html>



Réutilisation, les préfabs

- Les GameObjects, tel que nous venons de créer, peuvent être convertis en élément préfabriqués (Prefab). On peut ainsi les réutiliser dans les différentes parties du jeu sans avoir à reconstruire l'assemblage à chaque fois.
- Pour créer un prefab à partir d'un GameObject il suffit de glisser celui-ci dans le panneau **Project**.
- Pour mettre à jour le prefab/les objets on passe par le panneau inspecteur.

Plus de détails sur:

<http://docs.unity3d.com/Documentation/Manual/Prefabs.html>

Build

- Quand les scènes sont terminées, le jeu est « compilé » pour une architecture cible. [Files:Build settings...]

