

Les bases d'Unity



1. Préambule : (Important)

La partie A de ce TP n'est pas prioritaire. Elle sert à prendre en main Unity et à illustrer les mécanismes classiques et "sans code" d'animations avec les moteurs de jeu vidéo. Cette partie doit occuper le 1er TP d'1h30. La partie importante est le TP sur la "Cinématique inverse".

2. Unity : les bases

- [Téléchargez et installez Unity](#)
- [La documentation d'Unity est ici](#)

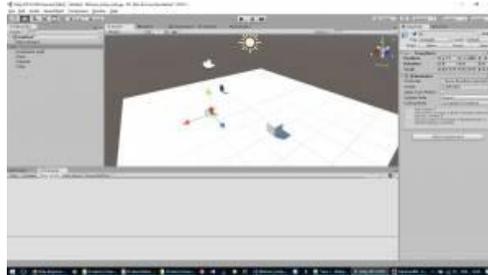
Créez un nouveau projet et une nouvelle scène. Sauvez bien la scène et le projet.

- Ajoutez un plan qui servira de terrain : menu "GameObject", "3D", "Plane"
- Bougez la caméra avec la souris + alt (rotation) ou ctrl+alt (Translation)
- Sélectionnez un objet (ou créez-en un), les touches pour changer de mode
 - Q Pan
 - W Move (Translation de l'objet)
 - E Rotate (Rotation de l'objet)
 - R Scale (Changement d'échelle de l'objet)



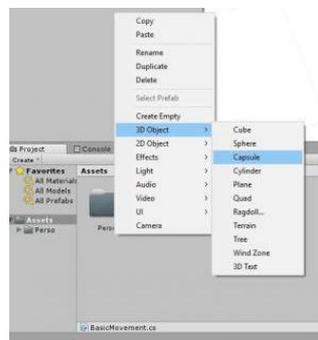
3. Contrôleur de déplacement d'une capsule au clavier/souris

L'objectif de cette partie est de piloter un objet au clavier/souris pour comprendre la notion de *GameObject* et de *Transform*.



[Une vidéo montrant les manipulations \(faites des pauses\) est ici.](#)

Faire bouger une capsule



- Créez un objet simple que nous allons piloter, par exemple une capsule. Menu "GameObject" puis "3D Object" puis "Capsule".
- Pour que l'objet fasse quelque chose, nous allons devoir écrire du code pour lui dire quoi faire. Dans le panneau *Project*, sélectionnez *Create* en haut (ou cliquez avec le bouton droit de la souris n'importe où dans le panneau) et sélectionnez *C# script*. Nommez-le *CharacterBasicControler*.
- Double-cliquez sur votre script pour l'ouvrir (Visual Studio ou Visual Code se lance). Chaque fois que vous créez un nouveau script, Unity ajoute deux méthodes par défaut, *Start()* et *Update()*. *Start* est appelé une fois, lorsque l'objet entre dans le monde du jeu. La mise à jour avec *Update* est appelée à chaque image/rendu.
- Copiez et collez la ligne suivante dans la méthode de mise à jour (*Update*).

```
transform.position += Vector3.forward * Time.deltaTime;
```

- Sélectionnez votre capsule en cliquant dessus. A droite dans le panel "*Inspector*", Ajoutez un composant "Add component", puis trouvez votre script de mouvement dans la petite barre de recherche (ou section "script") et ajoutez-le à votre capsule.
- Appuyez sur le bouton de lecture en haut de l'éditeur.
- Vous pouvez ajouter une variable de class "speed"

```
public float speed = 10.0f;
```

- Et modifiez le script. Notez que la variable "speed" est accessible aussi dans le panel "*Inspector*" car elle est *public*.

```
transform.position += Vector3.forward * speed * Time.deltaTime;
```

Contrôler la direction

- Renommez votre capsule en "Character", ce sera plus pertinent.
- Modifiez la fonction update

```
Vector3 forward_world = transform.TransformDirection(Vector3.forward);  
  
transform.position += forward_world * speed * Time.deltaTime;
```

- Décochez le "Capsule Collider" dans le panel "Inspector" à droite.
- Ajoutez un "Character Controller" avec le menu "Add component"
- Ouvrez votre script "CharacterBasicControler" et changez ceci

```
gameObject.GetComponent<CharacterController>().Move(forward_world * speed * Time.deltaTime);
```

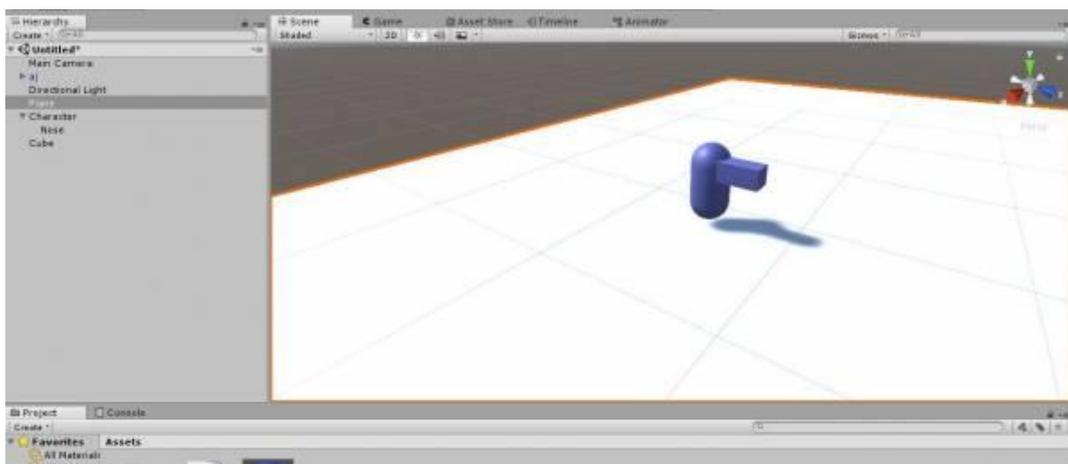
- Vous devriez garder le même comportement qu'avant mais maintenant c'est le Character Controller qui pilote votre Character.
- Ajoutez un cube qui servira de nez à la capsule pour savoir dans quelle direction il avance. Ce cube doit être un fils de "Character", faites glissez "nose" dans "Character" à gauche dans la hiérarchie de scène.
- Dans le script, ajoutez ces lignes qui récupèrent la souris et font tourner le Character (vous pouvez mettre speed à 0.1 dans l'interface ou dans la fonction "start" pour ralentir le mouvement)

```
Vector2 mouseInput = new Vector2(Input.GetAxis("Mouse X"), Input.GetAxis("Mouse Y"));  
  
transform.Rotate(Vector3.up, mouseInput.x * rotationSpeed);
```

- Modifiez la ligne du move par celle-ci pour utiliser le champ input

```
gameObject.GetComponent<CharacterController>().Move(transform.TransformDirection(input * speed * Time.deltaTime));
```

- Déplacez la caméra principale pour qu'elle devienne un enfant du Character (ou du nose). De cette façon, la caméra suivra le Character.



A faire sans guide

Vous pouvez enrichir votre projet.

- Tourner seulement si la touche "contrôle" est enfoncée
- Accélérer/freiner/turner avec les touches du clavier
- Ajoutez une rotation verticale pour pouvoir regarder en haut
- Activez les collisions entre votre capsule/perso et des cubes posés dans la scène
- Ajoutez des possibilités de tir de sphères droit devant

[Le tuto qui a inspiré cette partie du TP est ici.](#)