Exercice : Ajouter un Contrôle de Base pour le Personnage Joueur

L'objectif de cet exercice est de vous familiariser avec la création d'un contrôle de base pour un personnage joueur dans Unity, en utilisant un script en C# pour gérer les entrées clavier et déplacer le personnage dans l'environnement du jeu.

---

1. Préparation de l'Environnement

Avant de commencer à écrire le script, vous devez créer la scène et préparer les éléments nécessaires :

Étape 1 : Créer une nouvelle scène

1. Ouvrez **Unity** et créez un nouveau projet ou utilisez un projet existant.
2. Dans la **hiérarchie** de la scène, faites un clic droit et créez un nouvel objet **Cube** (ce sera votre personnage joueur). Vous pouvez aussi importer un modèle 3D de personnage si vous le souhaitez.

Étape 2 : Ajouter un sol

1. Créez un **plan** (`Plane`) pour servir de sol sous le personnage.
2. Assurez-vous que le personnage est bien positionné au-dessus du sol.

Étape 3 : Ajouter un Rigidbody

1. Sélectionnez le **Cube** dans la hiérarchie.
2. Dans l'**Inspector**, cliquez sur **Add Component** et cherchez `Rigidbody`. Cela permettra de gérer les collisions et la physique de l'objet.

---

2. Créer un Script de Contrôle

Étape 1 : Créer le script

1. Dans l'**Asset Window**, faites un clic droit et créez un **C# Script** que vous appellerez `PlayerController`.
2. Double-cliquez sur le script pour l'ouvrir dans l'éditeur de code (Visual Studio, par exemple).

Étape 2 : Écrire le code pour le contrôle du joueur

Voici le code pour déplacer un personnage en utilisant les touches **WASD** ou les flèches directionnelles.

```
using UnityEngine;

public class PlayerController : MonoBehaviour
{
    // Vitesse de déplacement du joueur
    public float moveSpeed = 5f;

    // Composant Rigidbody du joueur
    private Rigidbody rb;

    // Initialisation
    void Start()
    {
        // Récupérer le composant Rigidbody attaché à l'obj
et
        rb = GetComponent<Rigidbody>();
    }

    // Update est appelé une fois par frame
    void Update()
    {
        // Récupérer les entrées du joueur pour les axes ho
rizontal et vertical
        float horizontal = Input.GetAxis("Horizontal"); //
A/D ou flèches gauche/droite
        float vertical = Input.GetAxis("Vertical");     //
W/S ou flèches haut/bas

        // Créer un vecteur de déplacement
```

```
        Vector3 movement = new Vector3(horizontal, 0, verti
cal) * moveSpeed * Time.deltaTime;

        // Appliquer le mouvement au Rigidbody
        rb.MovePosition(transform.position + movement);
    }
}
```

Explication du code :

- `public float moveSpeed = 5f;` : Déclare une variable publique pour définir la vitesse du joueur, ce qui permet de la modifier facilement depuis l'éditeur Unity.
- `private Rigidbody rb;` : Déclare une variable pour stocker la référence du Rigidbody du personnage.
- `void Start()` : La méthode `Start()` est appelée au début. Ici, on récupère le composant **Rigidbody** attaché au personnage.
- `float horizontal = Input.GetAxis("Horizontal");` : Récupère l'entrée horizontale (les touches fléchées ou `A` et `D`).
- `float vertical = Input.GetAxis("Vertical");` : Récupère l'entrée verticale (les touches fléchées ou `W` et `S`).
- `Vector3 movement = new Vector3(horizontal, 0, vertical) * moveSpeed * Time.deltaTime;` : Crée un vecteur de mouvement basé sur les entrées du joueur. Le `Time.deltaTime` est utilisé pour rendre le déplacement indépendant du taux de frames par seconde.
- `rb.MovePosition(transform.position + movement);` : Déplace l'objet en fonction du vecteur de mouvement. On utilise `MovePosition` pour déplacer le Rigidbody de manière fluide, tout en gérant les collisions.

---

3. Attacher le Script à l'Objet

1. Sélectionnez votre objet **Cube** dans la hiérarchie.
2. Faites glisser le script **PlayerController** dans l'**Inspector** de l'objet, ou utilisez le bouton **Add Component** pour ajouter le script.
3. Si vous avez modifié la vitesse dans le script, vous pouvez ajuster la valeur de **moveSpeed** directement depuis l'Inspector pour tester différents

comportements de déplacement.

---

4. Tester le Contrôle

1. Cliquez sur le bouton **Play** dans Unity pour entrer en mode de jeu.
2. Utilisez les touches **WASD** ou les flèches directionnelles pour déplacer votre personnage dans la scène.
3. Le personnage devrait se déplacer de manière fluide dans la scène, selon les entrées de l'utilisateur.

---

5. Améliorations possibles

Pour améliorer le contrôle du joueur, vous pouvez :

- **Ajouter une rotation** pour faire face à la direction du mouvement. Par exemple :

```
// Faire tourner le joueur dans la direction du mouvemen
t
if (movement.magnitude > 0)
{
    Quaternion toRotation = Quaternion.LookRotation(move
ment, Vector3.up);
    transform.rotation = Quaternion.RotateTowards(transf
orm.rotation, toRotation, 720 * Time.deltaTime);
}
```

- **Limiter la vitesse de déplacement** : Vous pouvez ajouter une logique pour limiter la vitesse du personnage, afin d'éviter des mouvements trop rapides ou trop lents.
- **Ajouter des animations** : Pour les jeux plus avancés, vous pourriez intégrer des animations de marche ou de course en fonction de la vitesse du joueur.

---

6. Conclusion

Cet exercice vous a permis d'apprendre à créer un contrôle de base pour un personnage dans Unity. Vous avez vu comment récupérer les entrées du joueur et appliquer un déplacement à l'aide de `Rigidbody` et `Transform`. Ce type de contrôle est essentiel pour la création de nombreux types de jeux, et il peut être

étendu avec des fonctionnalités supplémentaires telles que la gestion de la caméra, les sauts, ou les animations pour créer une expérience de jeu plus immersive.