Projet : Implémenter un système de score et ajouter un niveau supplémentaire

Objectif :

Dans cet exercice, vous allez implémenter un système de score dans votre jeu et ajouter un niveau supplémentaire. Vous allez permettre au joueur de passer d'un niveau à l'autre tout en conservant le score, en utilisant la gestion de scènes et de données via Unity.

Étapes à suivre :

1. Implémenter le Système de Score

1. **Créer un script pour gérer le score :**
   - Créez un script C# nommé `ScoreManager.cs`.
   - Déclarez une variable `score` pour stocker le score du joueur et initialisez-la à 0.
   - Ajoutez des méthodes pour ajouter des points au score (par exemple, chaque fois que le joueur collecte un objet ou atteint un objectif).
   - Ajoutez des méthodes pour afficher le score à l'écran à l'aide de l'interface utilisateur (UI).

Exemple de script ScoreManager.cs :

```
using UnityEngine;
using UnityEngine.UI;

public class ScoreManager : MonoBehaviour
{
    public int score = 0;  // Variable pour stocker le scor
e
    public Text scoreText;  // Référence au texte affichant
le score
```

```
    // Méthode pour ajouter des points au score
    public void AddScore(int points)
    {
        score += points;
        UpdateScoreText();
    }

    // Méthode pour mettre à jour l'affichage du score
    void UpdateScoreText()
    {
        scoreText.text = "Score: " + score;
    }

    // Méthode pour réinitialiser le score (si nécessaire)
    public void ResetScore()
    {
        score = 0;
        UpdateScoreText();
    }
}
```

1. **Ajouter un élément UI pour afficher le score :**
   - Dans Unity, créez un texte dans votre scène pour afficher le score.
   - Associez ce texte à la variable `scoreText` dans le script `ScoreManager`.
2. **Testez le système de score :**
   - Ajoutez des objets dans le jeu que le joueur peut collecter (par exemple, des pièces ou des étoiles).
   - Modifiez les objets pour qu'ils appellent `AddScore()` lorsqu'ils sont collectés.

---

2. Ajouter un Niveau Supplémentaire

1. **Créer une nouvelle scène :**
   - Créez une nouvelle scène dans Unity pour représenter le niveau supplémentaire (par exemple, un niveau 2).

- Conception du niveau : ajoutez des plateformes, des ennemis, des obstacles, ou même des objectifs supplémentaires pour rendre le niveau intéressant.

2. **Ajouter la scène à la liste des scènes de construction :**
   - Allez dans **File > Build Settings**.
   - Cliquez sur **Add Open Scenes** pour ajouter la nouvelle scène à la liste.
   - Assurez-vous que votre première scène est également ajoutée à la liste des scènes.
3. **Créer une méthode pour passer d'un niveau à l'autre :**
   - Dans le script `ScoreManager.cs` ou dans un autre script de gestion des niveaux, implémentez une méthode pour charger la scène suivante lorsque le joueur atteint la fin du niveau.

Exemple de méthode pour changer de scène :

```
using UnityEngine;
using UnityEngine.SceneManagement;

public class LevelManager : MonoBehaviour
{
    // Méthode pour passer à la scène suivante
    public void LoadNextLevel()
    {
        int currentSceneIndex = SceneManager.GetActiveScene
().buildIndex;
        SceneManager.LoadScene(currentSceneIndex + 1);
    }
}
```

1. **Ajouter un point de passage pour passer au niveau suivant :**
   - Créez un objet de type **Collider** pour signaler la fin du niveau (par exemple, une porte ou un portail).
   - Assurez-vous que l'objet a un **Collider** et que le script `LevelManager` est attaché à un objet approprié dans la scène.
   - Utilisez la méthode `LoadNextLevel()` lorsque le joueur atteint cet objet.

Exemple d'interaction avec la fin du niveau :

```
using UnityEngine;

public class LevelEnd : MonoBehaviour
{
    public LevelManager levelManager;  // Référence au gest
ionnaire de niveau

    void OnTriggerEnter(Collider other)
    {
        if (other.CompareTag("Player"))  // Si l'objet entr
ant est le joueur
        {
            levelManager.LoadNextLevel();  // Charge le niv
eau suivant
        }
    }
}
```

1. **Ajouter des conditions de victoire :**
    - Dans votre niveau, vous pouvez définir des conditions de victoire (par exemple, atteindre un certain score, collecter tous les objets ou atteindre la fin du niveau).
    - Ajoutez des éléments visuels ou des messages pour informer le joueur qu'il a réussi à terminer un niveau.

---

3. Testez le jeu :

- Testez le système de score pour vous assurer qu'il est mis à jour correctement et affiché dans l'UI.
- Vérifiez que le changement de niveau fonctionne correctement, que le score est correctement conservé (si nécessaire), et que le joueur peut avancer vers le niveau suivant.
- Assurez-vous que la transition entre les niveaux est fluide, et que les objets collectés (si applicable) sont correctement pris en compte.

---

4. Bonus (Facultatif)

- **Persistance du score entre les niveaux :**
  Si vous voulez que le score du joueur persiste entre les niveaux (par exemple, que le score du niveau 1 soit ajouté à celui du niveau 2), vous pouvez utiliser
  `PlayerPrefs` pour sauvegarder et récupérer le score d'un niveau à l'autre.

Exemple d'utilisation de `PlayerPrefs` :

```
// Sauvegarder le score
PlayerPrefs.SetInt("Score", score);
PlayerPrefs.Save();

// Récupérer le score lors du chargement du niveau suivant
score = PlayerPrefs.GetInt("Score", 0);  // 0 est la valeur par défaut
```