Détection de collisions et déclenchement d'événements simples

La détection de collisions et le déclenchement d'événements sont des éléments essentiels dans le développement de jeux vidéo. Unity fournit plusieurs mécanismes pour gérer les collisions entre objets, ainsi que des moyens d'exécuter des actions spécifiques lorsqu'une collision ou un déclencheur se produit. Ce cours aborde ces concepts de base dans Unity, avec un accent particulier sur l'utilisation des **colliders**, des **triggers**, et des **événements** pour créer des interactions simples mais efficaces dans vos jeux.

---

1. Concepts de Base sur les Collisions dans Unity

Les collisions dans Unity sont gérées par deux composants principaux : **Collider** et **Rigidbody**.

- **Collider** : Définit la forme géométrique qui permet de détecter les collisions avec d'autres objets.
- **Rigidbody** : Permet à un objet d'être affecté par la physique (gravitée, forces, etc.). Un Rigidbody interagit avec les Colliders pour répondre aux événements physiques comme les collisions.

---

2. Types de Colliders

Il existe plusieurs types de **Colliders** que vous pouvez utiliser pour la détection de collisions :

- **BoxCollider** : Un **Collider** en forme de boîte, parfait pour des objets rectangulaires.
- **SphereCollider** : Un **Collider** sphérique, souvent utilisé pour des objets ronds.

- **CapsuleCollider** : Un **Collider** en forme de capsule (utilisé notamment pour les personnages).
- **MeshCollider** : Un **Collider** qui correspond à la forme de la géométrie de l'objet (utile pour les objets complexes).
- **TerrainCollider** : Utilisé pour les terrains dans Unity.

3. Activation du Détecteur de Collisions

Les **Colliders** peuvent être utilisés de manière physique ou comme **Triggers** :

- **Collider normal** : L'objet réagit physiquement à la collision (ex : il rebondit, est stoppé, etc.).
- **Trigger** : Un **Collider** qui ne génère pas de réaction physique mais qui sert à détecter les entrées et sorties d'autres objets. Un **Trigger** permet de déclencher des événements sans que l'objet ne réagisse physiquement.

4. Gestion des Événements de Collision

Unity offre des méthodes spécifiques pour détecter les collisions et déclencher des événements lorsque ces collisions se produisent.

Méthodes pour gérer les collisions :

1. **OnCollisionEnter** : Cette méthode est appelée lorsqu'une collision commence entre deux objets avec des **Colliders** non-triggers.
2. **OnCollisionStay** : Cette méthode est appelée à chaque frame où les deux objets restent en collision.
3. **OnCollisionExit** : Cette méthode est appelée lorsque la collision prend fin (lorsque les objets ne sont plus en contact).

Méthodes pour gérer les Triggers :

1. **OnTriggerEnter** : Cette méthode est appelée lorsqu'un objet entre dans une zone **Trigger**.
2. **OnTriggerStay** : Cette méthode est appelée à chaque frame où un objet reste dans une zone **Trigger**.
3. **OnTriggerExit** : Cette méthode est appelée lorsque l'objet quitte la zone **Trigger**.

---

5. Exemple d'Utilisation de la Détection de Collisions

Voyons un exemple simple d’utilisation des événements de collision pour gérer des interactions dans un jeu Unity. L'idée est de détecter lorsque le joueur touche un objet et de déclencher un événement (par exemple, afficher un message ou détruire l'objet).

Étapes à suivre :

1. Créez un cube dans la scène.
2. Ajoutez un **BoxCollider** et un **Rigidbody** à ce cube.
3. Cochez l'option **Is Trigger** dans le **BoxCollider** si vous souhaitez détecter la présence sans appliquer une réaction physique.
4. Créez un script pour détecter la collision avec le cube.

Script pour détecter une collision avec un objet Trigger :

```
using UnityEngine;

public class CollisionDetection : MonoBehaviour
{
    void OnTriggerEnter(Collider other)
    {
        // Vérifie si l'objet qui entre en collision a le t
ag "Player"
        if (other.CompareTag("Player"))
        {
            Debug.Log("Le joueur a touché l'objet !");
            // Par exemple, détruire l'objet
            Destroy(gameObject);
        }
    }
}
```

Dans cet exemple, le script détecte si un objet avec le tag "Player" entre en collision avec le cube. Si c’est le cas, le message "Le joueur a touché l'objet !" est affiché, et l'objet est détruit.

Exemple avec des collisions physiques (sans Trigger) :

Si vous ne souhaitez pas utiliser des triggers et préférez une réaction physique, vous pouvez utiliser **OnCollisionEnter**. Voici un exemple :

```
using UnityEngine;

public class CollisionDetection : MonoBehaviour
{
    void OnCollisionEnter(Collision collision)
    {
        // Vérifie si l'objet qui entre en collision a le t
ag "Player"
        if (collision.gameObject.CompareTag("Player"))
        {
            Debug.Log("Collision physique avec le joueur
!");
            // Par exemple, appliquer un effet ou une force
à l'objet
            collision.gameObject.GetComponent<Rigidbody>().
AddForce(Vector3.up * 500);
        }
    }
}
```

Dans ce cas, l’objet ne sera pas détruit, mais une force est appliquée au joueur (l’objet avec le tag "Player") en réponse à la collision.

---

6. Utilisation de Colliders et Triggers pour Déclencher des Événements

Les **Trigers** sont très utiles dans les situations où vous ne voulez pas de réactions physiques mais simplement détecter la présence ou l'interaction d'un objet. Voici quelques exemples :

- **Portes ou Entrées** : Vous pouvez utiliser un **Trigger** pour détecter quand le joueur passe devant ou entre dans une zone spécifique, et ouvrir une porte ou déclencher un événement (comme changer de scène).
- **Collecter des Objets** : Vous pouvez utiliser un **Trigger** pour collecter un objet quand le joueur entre dans la zone de celui-ci.

- **Zones de dégâts** : Si le joueur entre dans une zone de piège ou de dégâts, vous pouvez appliquer des effets de dommages.

---

7. Exemples de Déclenchement d'Événements

Voici un exemple simple où le joueur peut entrer dans une zone (détectée par un **Trigger**) pour récupérer un objet.

Script pour collecter un objet dans une zone de Trigger :

```
using UnityEngine;

public class CollectItem : MonoBehaviour
{
    void OnTriggerEnter(Collider other)
    {
        // Si le joueur entre dans la zone de l'objet (avec
le tag "Player")
        if (other.CompareTag("Player"))
        {
            Debug.Log("Objet collecté !");
            // Détruire l'objet collecté
            Destroy(gameObject);
        }
    }
}
```

Dans cet exemple, dès que le joueur entre dans la zone de **Trigger** de l’objet, celui-ci est collecté (il disparaît) et un message est affiché.

---

8. Conclusion

La détection de collisions et le déclenchement d’événements simples sont des éléments cruciaux dans la conception de jeux dans Unity. Grâce aux **Colliders** et aux **Triggers**, vous pouvez créer des interactions intéressantes et variées dans vos jeux sans avoir besoin de systèmes de physique complexes. En comprenant bien les différents types de **collisions** et **triggers**, vous serez capable de créer des jeux plus dynamiques et réactifs.