Colliders, Triggers and RigidBodies

Learning Objectives

- By the end of this session you should:
    - Know the function of colliders, triggers and ridgidbodies
    - Know the difference between colliders and triggers
    - Understand when to use a collider and when to use a trigger
    - Be able to do a basic implementation of them in Unity.

This project has two simple scenes which you can use to explore the basic functions of rigibodies, colliders and triggers.

https://github.com/tommakesgames/GS_Demo

Colliders and Rigidbodies

- Open up Scene1 (a cube and a sphere positioned over a tilted plane)
- Play the scene.

1. What happens?

A.

2. What component does the Cube have, which the Sphere lacks?

A.

- Add this component to the sphere.
- Play the scene.
    - *Both objects should now fall and hit the sloped floor.*
- Disable the 'Mesh Collider' on the Plane.

3. How does this change the object interaction?

A.

- Re-enable the 'Mesh Collider' on the Plane.
- Disable the 'Sphere Collider' on the Sphere.

4. How does this change things?

A.

5. What do you need for a physics collision to occur?

A.

6. What is the role of the collider, and what is the role of the rigidbody?

A.

Extension

- Play with the values of the rigidbody component
  - How does 'Mass', 'Drag' and 'Angular Drag' affect the objects? (this will be easier to see on the Cube).
- What happens when you toggle the 'Constraints' values (this menu is folded by default near the bottom - unfold it)?
- What are the 'Interpolate' options for? (hint: Use the in-built help shortcut to find out).

Colliders and Triggers

- Open up Scene2 (a floor, a wall and a ballLauncher are in the scene).
- Play the scene, and click with the left-mouse button, press space or press A/X on a controller.
- Note the debug messages in the console.

7. What happens at the moment?

A.

- Open up 'ballCollision.cs'. (This script is attached to the 'ball' prefab in our example)
- Select the wall. In the 'Box Collider' component, make sure 'Is Trigger' is ticked/true.
- Now play the scene.

8. What has changed?

A.

- In 'ballCollision.cs', in the `OnCollisionEnter` function, uncomment the line `Destroy(collision.gameObject);`.
- Make sure 'Is Trigger' for the Box Collider of the wall is now UNticked/false.
- Play the scene.

9. How does the ball react to the wall, even though it's destroyed?

A.

- In 'ballCollision.cs' in the `OnTriggerEnter` function, uncomment the line `Destroy(other.gameObject);`.
- Play the scene.

10. How does the ball react to the wall (trigger) being destroyed? How is it different from when the wall was a collider?

A.

11. Can you explain the difference between a trigger and a collider?

A.

Extension

Other ways to investigate and experiment with colliders and triggers:

- Click on the button next to 'Edit Collider'...
  - You can directly edit the collider by dragging faces (for a Box) and length and radius (for a sphere/capsule). When do you think this would be useful in a game?
- Create a Physics Material (right click in 'Project' > Create > Physics Material), drag and drop it onto the 'Material' field in the collider. Play with the values of the Physics Material - it's fairly self-explanatory! See how the ball interacts with the wall.
- Look at the 'ballMovement.cs' script (this is attached the to ball prefab in the Project).
  - Focus on the line `rb.AddForce(forceSpeed * transform.forward, ForceMode.Impulse);`
  - Investigate the possible values for `ForceMode`, and see how changing it from `Impulse` affects the ball's behaviour. There are four possible values.

Summary

When would you use a collider?

A.

When would you use a trigger?

A.

When do you need a rigidbody?

A.

What do you need to implement a collider (including scripting)?

A.

What do you need to include a trigger in your game (including scripting)?

A.