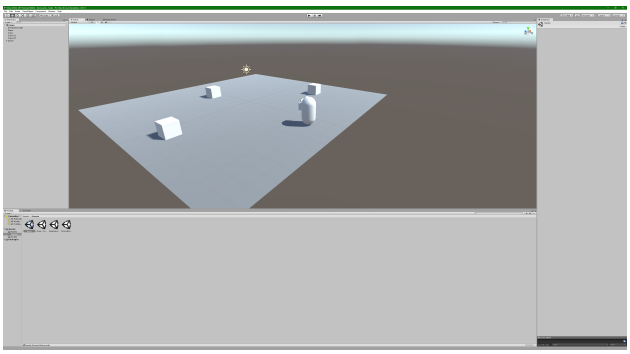
Tutorial 1 – FPS Game – Camera Controller

This tutorial will show how to create a First-Person Camera Controller to allow basic movement.



1. Create a New Scene

Start by creating a new Scene called FPSController.

Create a 3D Plane to allow the player perspective of movement when testing the script.

Create a 3D Capsule and name it “Player”, removing the Capsule Collider in the Inspector tab and adding a Character Controller via Add Component.

Rename the Main Camera to “Eyes”, and parent this to the Player capsule, setting all position values to 0.

2. Creating the Script

Create a new script called FPSController.

We’re going to create movement using the W, A, S, and D keys via the Inputs Axis. To make this, we need to add some variables, such as `speed`, `moveLR`, and `moveFB`. The `speed` can be adjusted later, but for now let’s just set it to 2f. We will be using `moveLR` and `moveFB` to equate to the value of the virtual axis in the Input.

We also need to set up `rotX`, `rotY`, and `sensitivity` variables to ensure the character rotates based on the mouse position.

Also, create a `CharacterController` named player, which allows us to store a Character Controller as a variable.

Creating a public `GameObject` for the camera allows us to lock the rotation to the camera alone. We'll call this "`eyes`".

```
public class FPSController : MonoBehaviour {

	public float speed = 2f;
	public float sensitivity = 2f;
	CharacterController player;

	public GameObject eyes;

	float moveLR;
	float moveFB;

	float rotX;
	float rotY;
```

Before creating movement, we need to store the `CharacterController` as a variable, allowing us to change and manipulate that component. Using `GetComponent` will only work if the component you are trying to get is on the same object that the script is on.

```
void Start () {

	player = GetComponent<CharacterController> ();

}
```

Now moving into `Update`, we will assign the floats we made to the correct Axis. If the W key is pressed the value would equal 1, so we multiply the value by the `speed` to create the movement.

Now that we are monitoring the horizontal and vertical axis, we now want to move the player along that. Creating a `new Vector3` allows us to allocate the X and Z values to the `moveLR` and `moveFB` variables we created.

The player's rotation will direct what is forward and backward, so to set this we multiply the rotation by the `new Vector3` movement we just created.

Using the Character Controller variable of `player` that we created earlier, we now want to set the movement of the Player using `Time.deltaTime`.

Using the same technique as we did earlier, we will now use Input.GetAxis to allow rotation with the mouse. To get this to happen, we will get the `rotX` to take place on the Y axis. However, to get the `rotY` to happen it must only be placed on the camera which can be done by utilizing the `GameObject` we created for the `eyes` earlier.

```
void Update () {

	moveLR = Input.GetAxis("Horizontal") * speed;
	moveFB = Input.GetAxis("Vertical") * speed;

	rotX = Input.GetAxis("Mouse X") * sensitivity;
	rotY = Input.GetAxis("Mouse Y") * sensitivity;

	Vector3 movement = new Vector3 (moveLR, 0, moveFB);
	transform.Rotate (0, rotX, 0);
	eyes.transform.Rotate (-rotY, 0, 0);
	movement = transform.rotation * movement;
	player.Move (movement * Time.deltaTime);

}
```

3. Back to Unity

Switching back to the Unity Editor, drag the FPSController Script onto the Player in the Hierarchy. Also drag and drop the camera "Eyes" in the Hierarchy into the GameObject "Eyes" in the FPSController Script in the Inspector tab.

Now press play and see your FPSController in action!