Scripting and UI

Vijay Kalivarapu
Day 1 Review

- Game Engines
- Unity Interface
- Cameras, Lights, and Objects
- Scripting in C#
Enabling and Disabling Components

```
using UnityEngine;
using System.Collections;

public class LightScript : MonoBehaviour {
    private Light myLight;
    // Use this for initialization
    void Start ()
    {
        myLight = GetComponent<Light> ();
    }

    // Update is called once per frame
    void Update ()
    {
        if(Input.GetKeyUp(KeyCode.Space))
        {
            myLight.enabled = !myLight.enabled;
        }
    }
}
```
Activating Game Objects

• Making a GameObject inactive will disable every component and turn off any attached renderers, colliders, rigid bodies, scripts, etc...

• Any scripts that you have attached to the GameObject will no longer have Update() called
Activity

• Build a simple scene with a cube, a sphere, and two different colored lights. One pointed at each object.

• Use two different keyboard keys to turn on/off the lights by setting them active/inactive

• Keyboard input is of the form “Input.GetKeyUp(KeyCode.Space)”
Getting a Component

- GetComponent<Type>()
- Allows you access to any Component in the object
- You can access Parent and Children too
Calling Other Scripts

• Scripts are GameComponents, so you can use `GetComponent<Type>()` or `FindObjectOfType<Type>()` to obtain a reference to other scripts.

```csharp
using UnityEngine;
using System.Collections;

public class KeyboardInput : MonoBehaviour {
    private AnimationScript animationScript;

    void Start ()
    {
        animationScript = GetComponent<AnimationScript>();
    }

    void Update ()
    {
        if(Input.GetKeyUp(KeyCode.Space))
        {
            animationScript.animate();
        }
    }

    public void animate()
    {
        animating = !animating;
    }
}
```
Activity

- Use your previous scene
- Create a keyboard input script that looks for a button press
- Have the keyboard input script call to the scripts attached to your lights to control turning them on/off
Unity User Interfaces
UI Canvas

- Everything UI starts with the Canvas
- Canvas is a GameObject
- All UI elements must be children of a canvas
UI Text

• Whenever you need text
• Text properties can be set in the Inspector
• Can be changed during runtime through scripting
UI Image

- Can be used for almost anything, button, slider, etc.
- When importing an image, you must define what type of texture it is (Normal Map, Light Map, Sprite)
- For UI, we want a Sprite
UI Button

- Button is a GameObject that must be a child of a canvas
- Many different options for styling
On Click()

- You can hook up a button to an action through the Inspector
- Chose your GameObject
- Choose your Component
- Choose your Method
Activity

• Using the same scene

• Update your project to use UI buttons to control the lighting instead of the keyboard presses
Creating an Executable

• What if I want to create a standalone app?

• Let’s make an executable
Creating an Executable

• Add the desired scene
• Select your platform
• Build and Run!