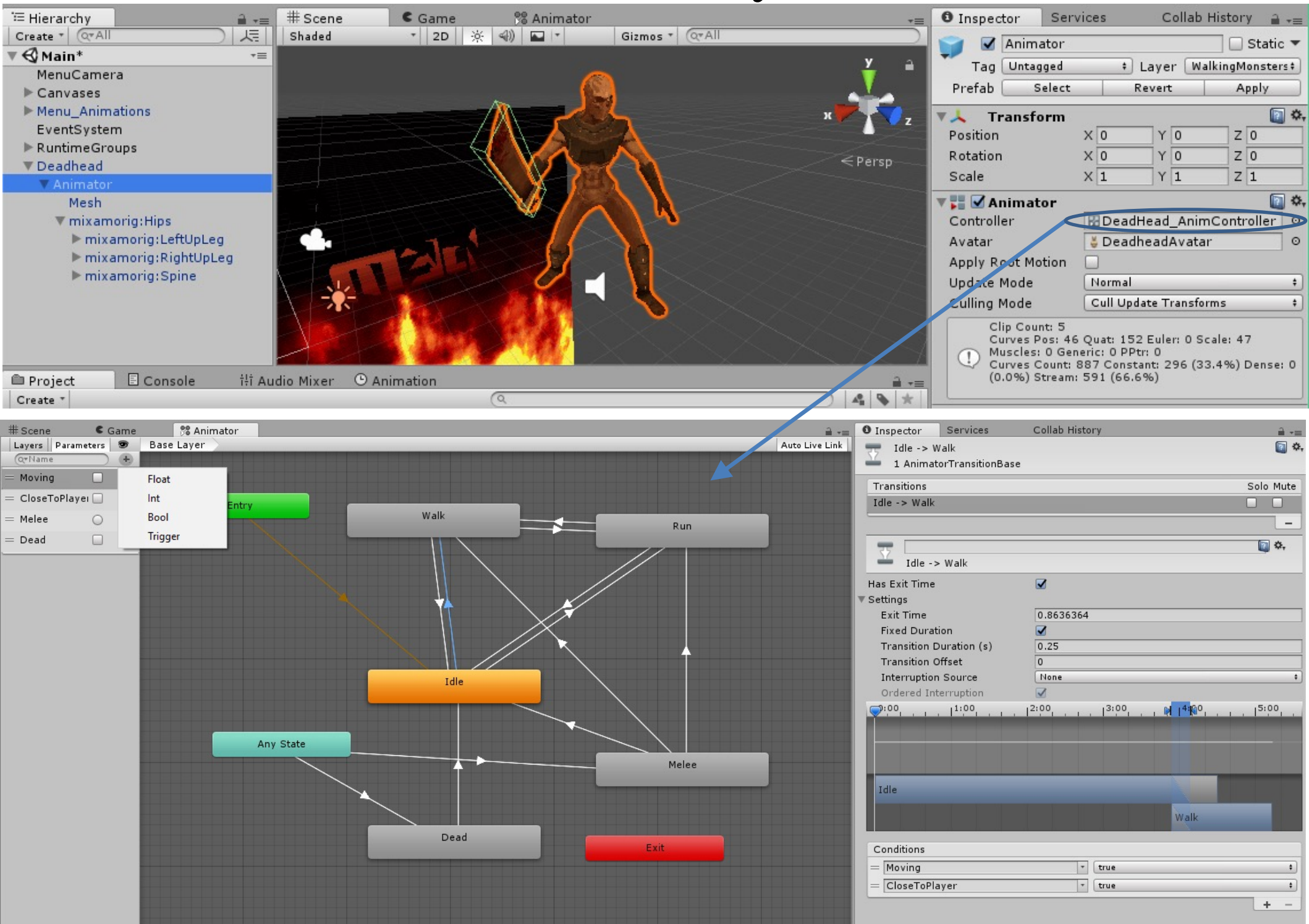
CT3536
(Games Programming using Unity3D)

Animators
Materials
Lights

Unity Animators

- https://docs.unity3d.com/Manual/class-Animator.html
- The Animator component is used to assign animation to a GameObject in your scene
- The Animator component requires a reference to an Animator Controller, which defines which animation clips to use, and controls when and how to blend and transition between them.
- The Animator Controller operates as a finite state machine

The 'DeadHead' enemy in DemonPit

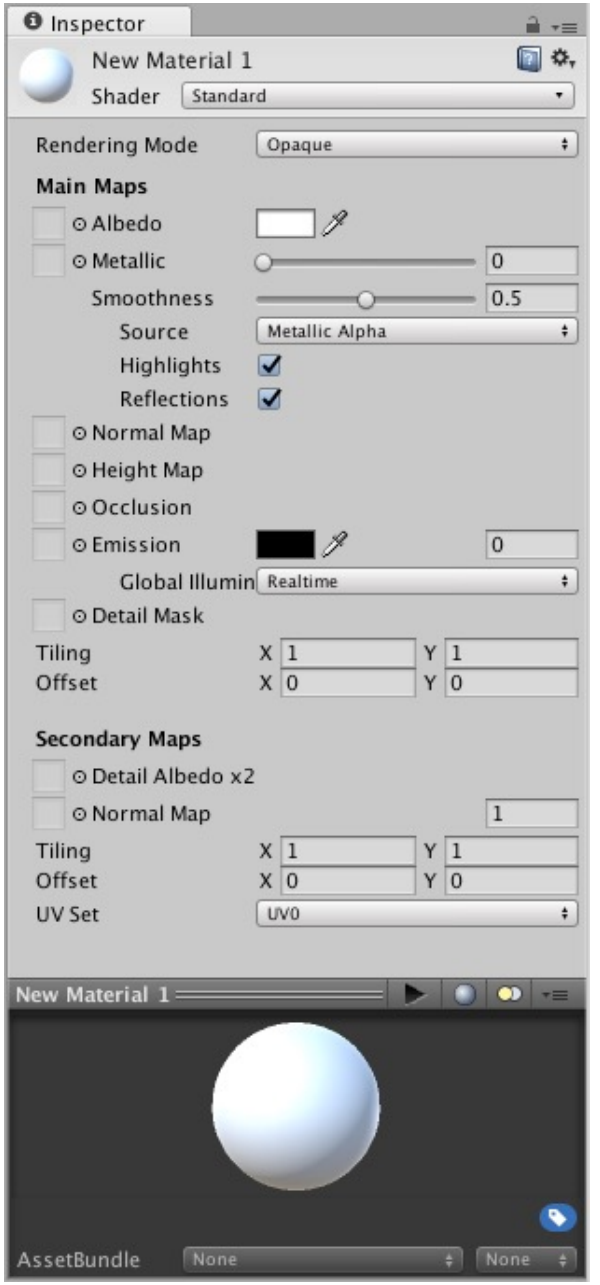


Materials

- https://docs.unity3d.com/Manual/Materials.html
- Note that by default in Unity all prefabs using a particular material will share a single instance of that material.
- Therefore if any of a material's settings need to change at runtime, you should make separate instances of the material for each prefab
- E.g. put this in a Start( ) method:

```
Renderer r = GetComponent<Renderer>();
r.material = Instantiate(r.material);
```

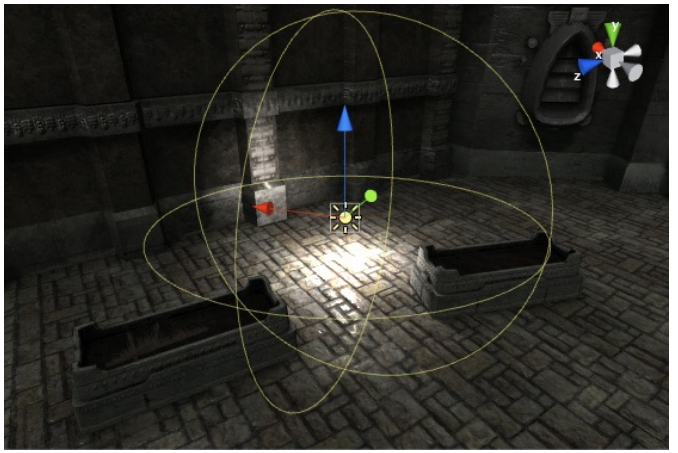
- The material shown here uses the Standard shader, which has a lot of settings

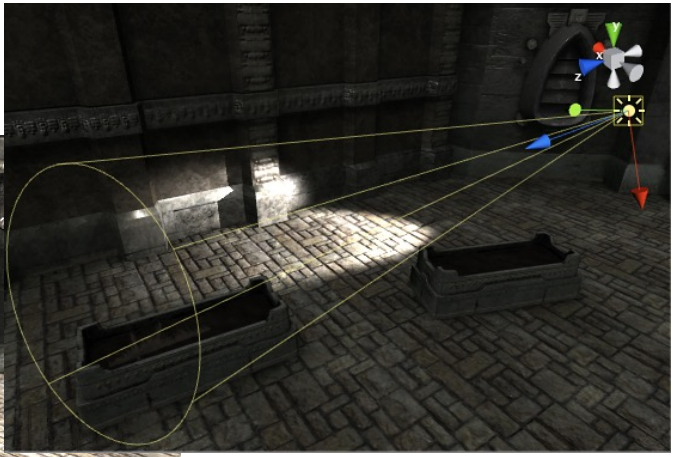


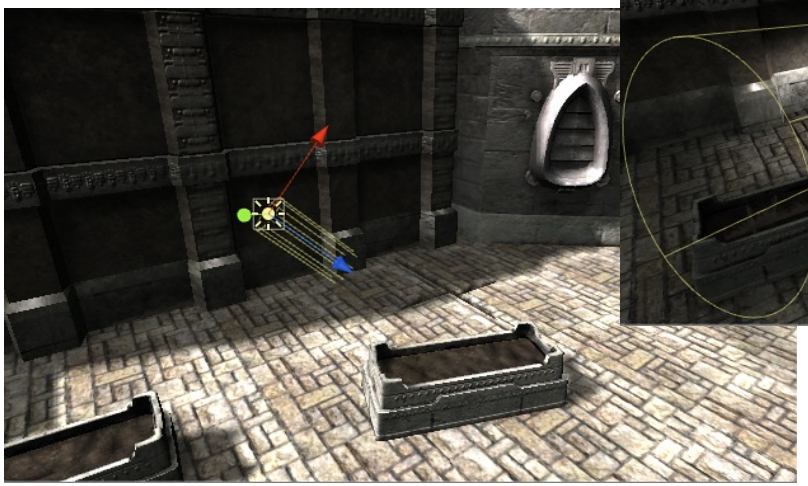
Lights

- https://docs.unity3d.com/Manual/Lighting.html
- https://docs.unity3d.com/Manual/UsingLights.html

- A point light is located at a point in space and sends light out in all directions. The intensity diminishes with distance from the light, reaching zero at a specified range.
- Spot lights are generally used for artificial light sources such as flashlights, car headlights and searchlights
- Directional lights represent large, distant sources that come from a position outside the range of the game world







Week4LectureExamples
edit: attach the Point Light object

```
public class ChainRoot : MonoBehaviour {

    // inspector settings
    public GameObject chainLinkPrefab;

    void Start () {
        // create a bunch of connected chain links
        Vector3 pos = transform.position;
        Rigidbody previous = this.GetComponent<Rigidbody>();
        Vector3 anchorOffset = new Vector3(0f, 1.5f, 0f);

        for (int i=0; i<10; i++) {
            GameObject go = Instantiate(chainLinkPrefab);
            pos.y -= 1f;
            go.transform.position = pos;
            SpringJoint sj = go.GetComponent<SpringJoint>();
            sj.connectedBody = previous;
            sj.connectedAnchor = anchorOffset;

            if (i==9) {
                // make the last link bigger and heavier
                go.GetComponent<Rigidbody>().mass *= 5f;
                go.transform.localScale = new Vector3(2f,2f,2f);
                // attach the scene's "Point Light" to the last link, just below it
                GameObject pl = GameObject.Find("Point Light"); // find the named object in the scene
                pl.transform.SetParent(go.transform);
                pl.transform.localPosition = new Vector3(0f,-0.5f,0f);
            }

            previous = go.GetComponent<Rigidbody>();
        }

    }
}
```

Week4LectureExamples
A new script on the Point Light GameObject

```
public class LightAnimator : MonoBehaviour {

    private Light light;
    private float intensityChangePerSec = -1f;

    void Start () {
        light = GetComponent<Light>();
        StartCoroutine( AnimateColor() );
   }

    void Update() {
        light.intensity += intensityChangePerSec * Time.deltaTime;
        if (light.intensity<=0f) {
            light.intensity = 0f;
            intensityChangePerSec = Mathf.Abs(intensityChangePerSec);
        }
        else if (light.intensity>=3f) {
            light.intensity = 3f;
            intensityChangePerSec = -Mathf.Abs(intensityChangePerSec);
        }
    }

    private IEnumerator AnimateColor() {
        WaitForSeconds delayPerColor = new WaitForSeconds(2f);
        Color red = new Color(1,0,0);
        Color yellow = new Color(1,1,0);
        Color green = new Color(0,1,0);

        while (true) {
            light.color = red;
            yield return delayPerColor;
            light.color = yellow;
            yield return delayPerColor;
            light.color = green;
            yield return delayPerColor;
        }
    }
}
```