# Intro to Unity

Introduction to Scripting

# Creating a New Script

- To create a new script in the project view or create a new script via. the add component menu.
- Scripts maintain their own state when added to multiple objects.
- Scripts can provide publicly accesible properties that changed in the inspector.

## GameObject Events

- Scripts can be run in each of the GameObject lifecycle methods.
- Inside of the events, you can access of the components and modify their values.
- Scripts be accessed from other GameObjects.



#### **Custom Events**

- Use events to broadcast changes in your object states.
- Objects can create their own events.
- Any object can subscribe to these events and then respond to them.
- UnityScript does not support this event model.

# Custom Events (cont'd)

```
public delegate void BallEventHandler(GameObject ball, GameObject target);
```

public static event BallEventHandler onBallCollideWithGround;

```
if (onBallCollideWithGround != null)
{
    onBallCollideWithGround(transform.gameObject, collision.gameObject);
}
```

```
BallScript.onBallCollideWithGround+= this.LoseBall;
```

```
public void LoseBall(GameObject ball, GameObject ground) { //code }
```

#### Coroutines

- These are functions are called during intervals.
- Coroutines are not threads nor are they asynchronous.
- Coroutines partially execute a function until they reach a yield statement.
- Coroutines will run indefinitely until they are manually stopped or the attached object is destroyed.

# Coroutines (cont'd)

```
private IEnumerator enableColumns()
{
   for (int i = 0; i < columns.Length; i++)
   {
      GameObject column = columns[i];
      column.SetActive(true);
      yield return new WaitForSeconds(.1f);
   }
}</pre>
```

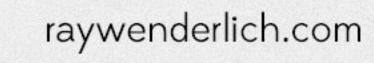
```
StartCoroutine("enableColumns");
```

```
StopCoroutine("enableColumns");
```

## Memory Management

- Memory is handled by automatic garbage collection.
- Nulled objects are collected by the garbage collector.
- Excessive garbage collection can affect the game's frame rate.
- Aim to pool and reuse objects as opposed to create and deallocate objects.

### Demo



## Challenge

```
public float secondsPerColor = 5.0f;
private Color endColor;
private Color startColor;
private Light pointLight;
private float currentTimeDuration = 0;
void Start()
    pointLight = GetComponent<Light>();
    startColor = pointLight.color;
```