Lecture 3
Wrangling Unity

98-127: Game Creation for People Who Want to Make Games (S19)

Written by Adrian Biagioli

Instructors:
Adrian Biagioli (abiagiol@andrew.cmu.edu)
Carter Williams (ncwillia@andrew.cmu.edu)

1 Objectives

By the end of this lesson you will be able to:

These lecture notes were written for **Unity 2018.3.3f1**.

2 Basic Functions

BasicFunctions‣StartTester

```
using UnityEngine;

public class StartTester : MonoBehaviour
{
    // Start() is called exactly once when you launch the game
    private void Start()
    {
        // Use Debug.Log(...) to log to the Console view
        Debug.Log("Hello, World!");
    }
}
```

BasicFunctions‣StartTester

```
using UnityEngine;

public class StartTester : MonoBehaviour
{
    // Start() is called exactly once when you launch the game
    private void Start()
    {
        // Use Debug.Log(...) to log to the Console view
        Debug.Log("Hello, World!");
    }
```

```
}
```

BasicFunctions‣UpdateTester

```
using UnityEngine;

public class UpdateTester : MonoBehaviour
{
    // Update() is called once per frame
    void Update()
    {
        Debug.Log("Hello, World! (a lot)");
    }
}
```

BasicFunctions‣FieldTester

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class FieldTester : MonoBehaviour
{
    // Mark fields with [SerializeField] to allow them to be edited in the
        inspector
    [SerializeField]
    private string StringToPrint;

    private void Start()
    {
        Debug.Log(StringToPrint);
    }
}
```

BasicFunctions‣LotsOfFieldsTester

```
using UnityEngine;
using UnityEngine.Events;

public class LotsOfFieldsTester : MonoBehaviour
{
    [Header("Basic Fields")]
    [SerializeField]
    private int _IntField;
    [SerializeField]
    private float _FloatField = 5.0f;
    [SerializeField]
    private string _StringField = "Test Field";
    [SerializeField]
    private Vector3 _VectorField = new Vector3(42, 69, 1337);

    [Header("Object Fields")]
    [SerializeField]
    private GameObject _GameObjectField;
    [SerializeField]
    private Rigidbody _ComponentField1;
    [SerializeField]
    private MeshRenderer _ComponentField2;
    [SerializeField]
    private Material _AssetField1;
    [SerializeField]
    private Mesh _AssetField2;

    [Header("Basic Fields with Controls")]
    [SerializeField]
    [Range(-1.0f, 1.0f)]
    private float _RangeFloatField;
    [SerializeField]
    [TextArea]
    private string _LargeStringField;

    [Header("Weird/Advanced Fields")]
    [SerializeField]
    private UnityEvent _EventField;
    [SerializeField]
    private LayerMask _LayerMaskField;
}
```

BasicFunctions‣ClosedFormAnimation

```
using UnityEngine;

```

```
public class ClosedFormAnimation : MonoBehaviour
{
   [SerializeField]
   private float Radius = 1.0f;
   [SerializeField]
   private Vector3 Center = Vector3.zero; // Vector3.zero == new
      Vector3(0,0,0)
   [SerializeField]
   private float Speed = 1.0f;

   private void Update()
   {
      float s = Mathf.Sin(Time.time * Speed);
      float c = Mathf.Cos(Time.time * Speed);

      transform.position = Center + new Vector3(c * Radius, s * Radius, 0);
   }
}
```

BasicFunctions▸DeltaTimeAnimation

```
using UnityEngine;

public class DeltaTimeAnimation : MonoBehaviour
{
   // You don't need [SerializeField] for public variables
   public float Speed = 1.0f; // speed in m/s
   public Vector3 Direction = new Vector3(1,0,0);
   public bool localPosition = false;

   // Update is called once per frame
   private void Update()
   {
      Vector3 delta = Speed * Direction.normalized;

      // This doesn't work!! What is wrong?
      if (localPosition)
         transform.localPosition += delta;
      else
         transform.position += delta;
   }
}
```

BasicFunctions▸SimpleMovement

```
using UnityEngine;

public class SimpleMovement : MonoBehaviour
```

```
{
   [SerializeField]
   private string _HorizontalMovementAxis = "Horizontal";
   [SerializeField]
   private string _VerticalMovementAxis = "Vertical";
   [SerializeField]
   private float _MovementSpeed = 1.0f;

   private void Update()
   {
      float hoz = Input.GetAxis(_HorizontalMovementAxis);
      float vrt = Input.GetAxis(_VerticalMovementAxis);

      Vector3 mov = new Vector3(hoz, vrt, 0);

      if (mov.sqrMagnitude > 1.0f)
         mov.Normalize(); // make vector have length 1

      transform.position += mov * Time.deltaTime;
   }
}
```

BasicFunctions‣ChangeMaterialTester

```
using UnityEngine;

[RequireComponent(typeof(MeshRenderer))]
public class ChangeMaterialTester : MonoBehaviour
{
   [SerializeField]
   private Gradient _Gradient;
   [SerializeField]
   [Tooltip("Length in seconds to cycle through the gradient")]
   private float _CycleLength;

   // Note: This field isn't serialized, so we can't edit it in the inspector!
   private MeshRenderer _Renderer;

   private void Start()
   {
      // GetComponent<T> gets the component with type T attached to the
         current GameObject
      // If none exist, returns null
      _Renderer = GetComponent<MeshRenderer>();
   }

   private void Update()
   {
```

```
        float a = (Mathf.Cos(Time.time / _CycleLength * (2.0f * Mathf.PI)) +
            1.0f) / 2.0f;
        Color c = _Gradient.Evaluate(a);

        _Renderer.material.color = c;
    }
}
```

3 Physics Examples

FIRST: Make sure to talk about Physics materials and Colliders!

PhysicsExamples▸CollisionDetectionTester

```
using UnityEngine;

public class CollisionDetectionTester : MonoBehaviour
{
    private void OnTriggerEnter(Collider other)
    {
        Debug.Log("Just got triggered by GameObject called " +
            other.gameObject.name);
    }

    private void OnTriggerExit(Collider other)
    {
        Debug.Log("Just ended trigger by GameObject called " +
            other.gameObject.name);
    }

    private void OnCollisionEnter(Collision collision)
    {
        Debug.Log("Just collided with GameObject called " +
            collision.gameObject.name);
    }

    private void OnCollisionExit(Collision collision)
    {
        Debug.Log("Just ended collision with GameObject called " +
            collision.gameObject.name);
    }
}
```

PhysicsExamples▸CollisionDetectionTester2D

```
using UnityEngine;

```

```
public class CollisionDetectionTester2D : MonoBehaviour
{
    private void OnTriggerEnter2D(Collider2D other)
    {
        Debug.Log("Just got triggered by GameObject called " +
            other.gameObject.name);
    }

    private void OnTriggerExit2D(Collider2D other)
    {
        Debug.Log("Just ended trigger by GameObject called " +
            other.gameObject.name);
    }

    private void OnCollisionEnter2D(Collision2D collision)
    {
        Debug.Log("Just collided with GameObject called " +
            collision.gameObject.name);
    }

    private void OnCollisionExit2D(Collision2D collision)
    {
        Debug.Log("Just ended collision with GameObject called " +
            collision.gameObject.name);
    }
}
```

PhysicsExamples▸EventOnTrigger

```
using UnityEngine;
using UnityEngine.Events;

public class EventOnTrigger : MonoBehaviour
{
    [SerializeField]
    private UnityEvent _OnEnter;
    [SerializeField]
    private UnityEvent _OnExit;

    private void OnTriggerEnter(Collider other)
    {
        if(_OnEnter != null)
            _OnEnter.Invoke();
    }

    private void OnTriggerExit(Collider other)
    {
        if(_OnExit != null)
```

```
            _OnExit.Invoke();
        }
    }
```

PhysicsExamples▸EventOnLookAndPress

```
// Don't write this script in front of the class. But you can show it being used!
using UnityEngine;
using UnityEngine.Events;

public class EventOnLookAndPress : MonoBehaviour
{
    [SerializeField]
    private string _ButtonAxis = "Fire1"; // default to left click
    [SerializeField]
    private float _ReachDistance = 5;
    [SerializeField]
    private UnityEvent _Event;

    private void Update()
    {
        if (_Event == null)
            return;

        Camera cam = Camera.main;
        if (cam == null) // no main camera
            return;

        // GetButtonDown is ONLY true on the frame you pressed the button
        if(Input.GetButtonDown(_ButtonAxis))
        {
            RaycastHit hit;
            if(Physics.Raycast(cam.transform.position, cam.transform.forward,
                out hit, _ReachDistance))
            {
                if (hit.collider.gameObject == gameObject)
                    _Event.Invoke();
            }
        }
    }
}
```

4 Prefabs

FIRST: Introduce Prefabs using the Prefab example scene

PrefabExamples‣SpawnPrefabOnButton

```
using UnityEngine;

public class SpawnPrefabOnButton : MonoBehaviour
{
   [SerializeField]
   private string _ButtonToPress = "Fire1";
   [SerializeField]
   private GameObject _PrefabToSpawn;
   [SerializeField]
   private Transform _TransformToSpawnAt;

   private void Update()
   {
      if(Input.GetButtonDown(_ButtonToPress))
      {
         var go = Instantiate<GameObject>(_PrefabToSpawn);
         go.transform.position = _TransformToSpawnAt == null ?
            transform.position : _TransformToSpawnAt.position;
         go.transform.rotation = _TransformToSpawnAt == null ?
            transform.rotation : _TransformToSpawnAt.rotation;
      }
   }
}
```

PrefabExamples‣SpawnPrefabInterval

```
using System.Collections;
using UnityEngine;

public class SpawnPrefabInterval : MonoBehaviour
{
   [SerializeField]
   private float _SpawnIntervalSeconds = 1.0f;
   [SerializeField]
   private GameObject _PrefabToSpawn;
   [SerializeField]
   private Transform _TransformToSpawnAt;

   private void Start()
   {
      StartCoroutine(SpawnPrefabCoroutine());
   }

   private IEnumerator SpawnPrefabCoroutine()
   {
      while(true)
```

```
21          {
22              var go = Instantiate<GameObject>(_PrefabToSpawn);
23              go.transform.position = _TransformToSpawnAt == null ?
                    transform.position : _TransformToSpawnAt.position;
24              go.transform.rotation = _TransformToSpawnAt == null ?
                    transform.rotation : _TransformToSpawnAt.rotation;
25
26              yield return new WaitForSeconds(_SpawnIntervalSeconds);
27          }
28      }
29  }
```