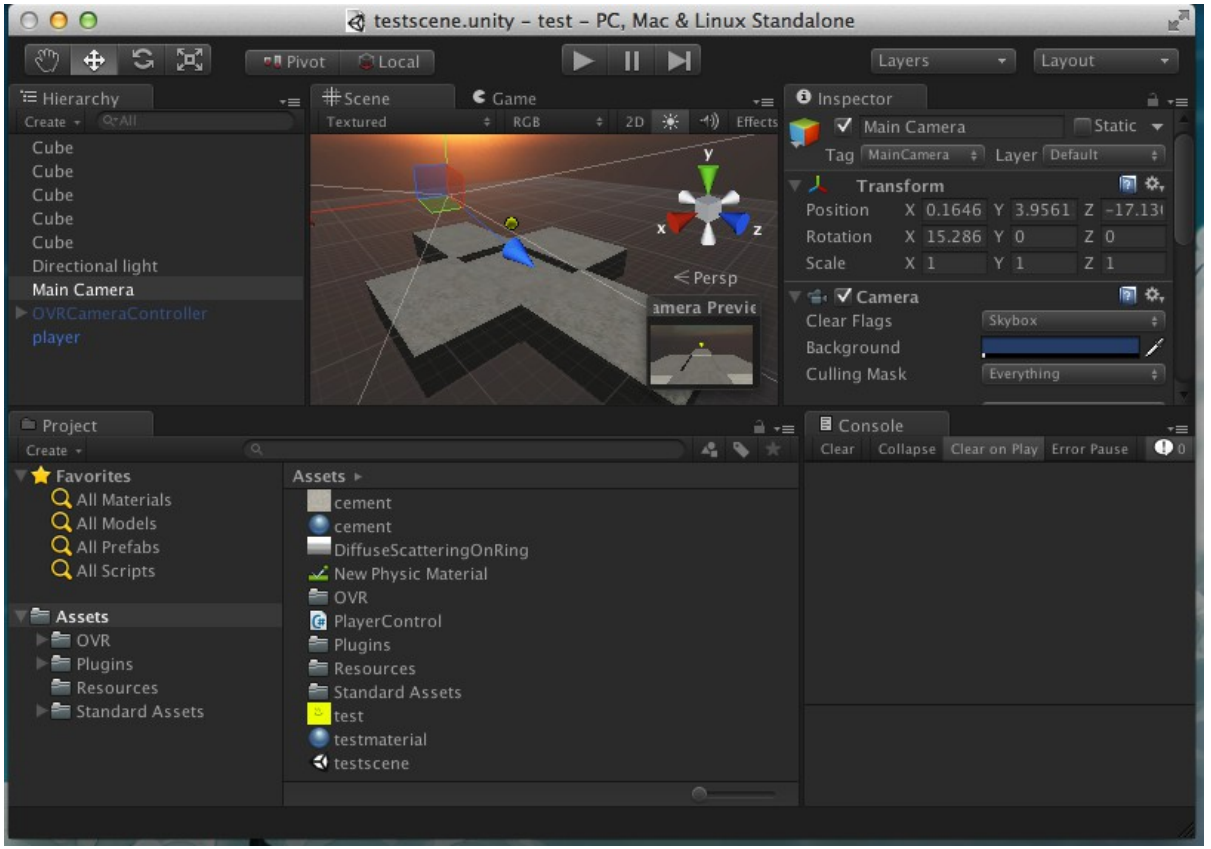
UW GAME DEV CLUB

https://unity3d.com/unity/download



UW GAME DEV CLUB

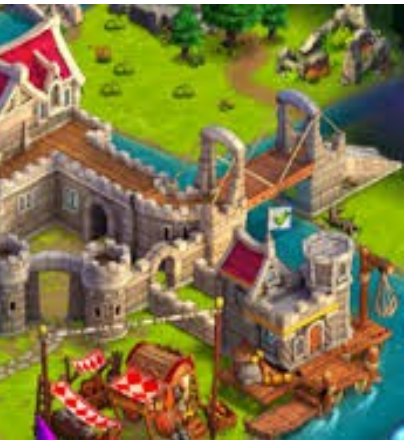
Unity Overview

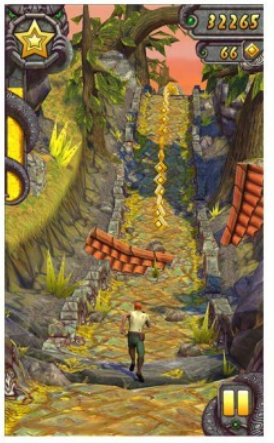
Cons:

-It's a huge download
-It takes a lot of memory and sometimes crashes
-It's not completely free
-Hard to do some things that are easy on other platforms (Ex: vector graphics)
-C# kinda sucks in its own way

Pros:

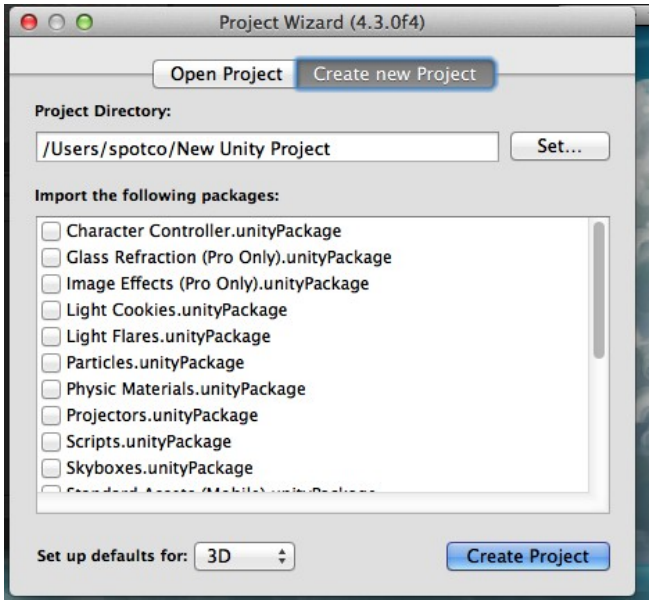
-Exports to almost any platform (PC, Web, iOS, Android)
-Incredibly easy to create 3d games (no 3d graphics programming required)
-Powerful GUI Editor
-Built in support for internal physics engine
-Easy integration for Oculus Rift





UW GAME DEV CLUB

Getting started...

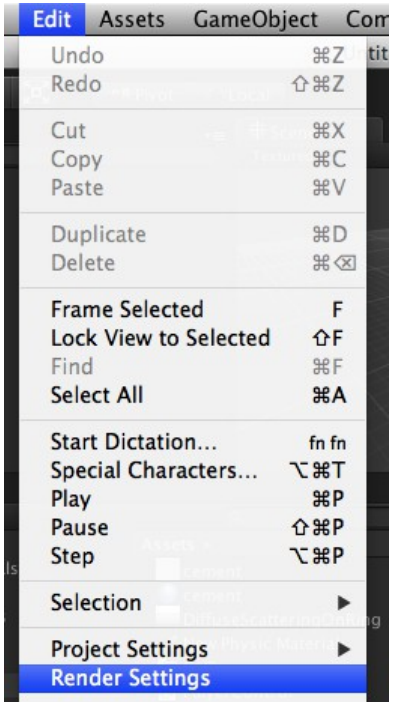


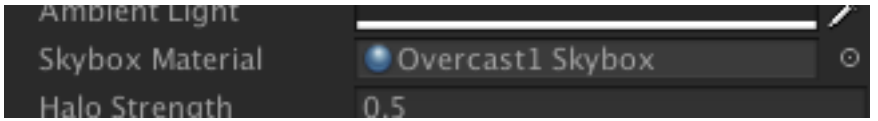
Create new project

Select any that sound interesting

(Check out Skyboxes and Toonshader)

(you may need to download Toonshader from the asset store)

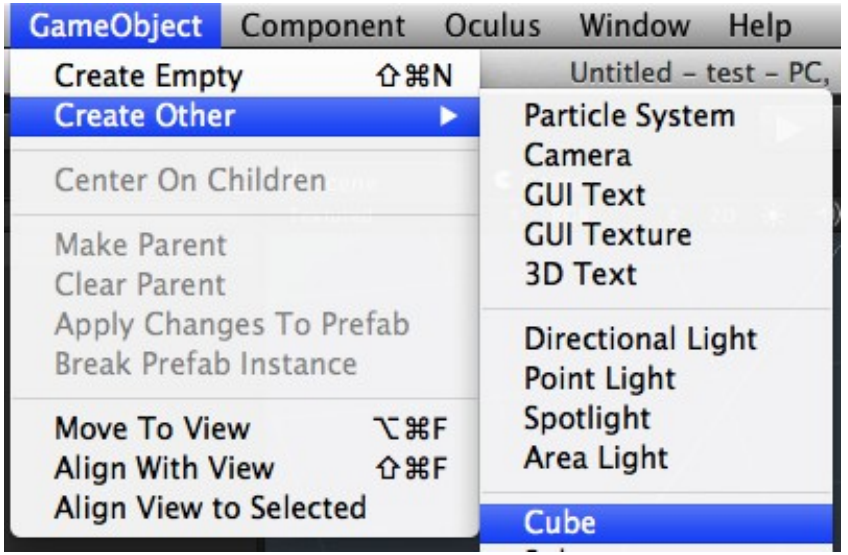




Add skybox:
Edit->Render Settings->Skybox Material

UW GAME DEV CLUB

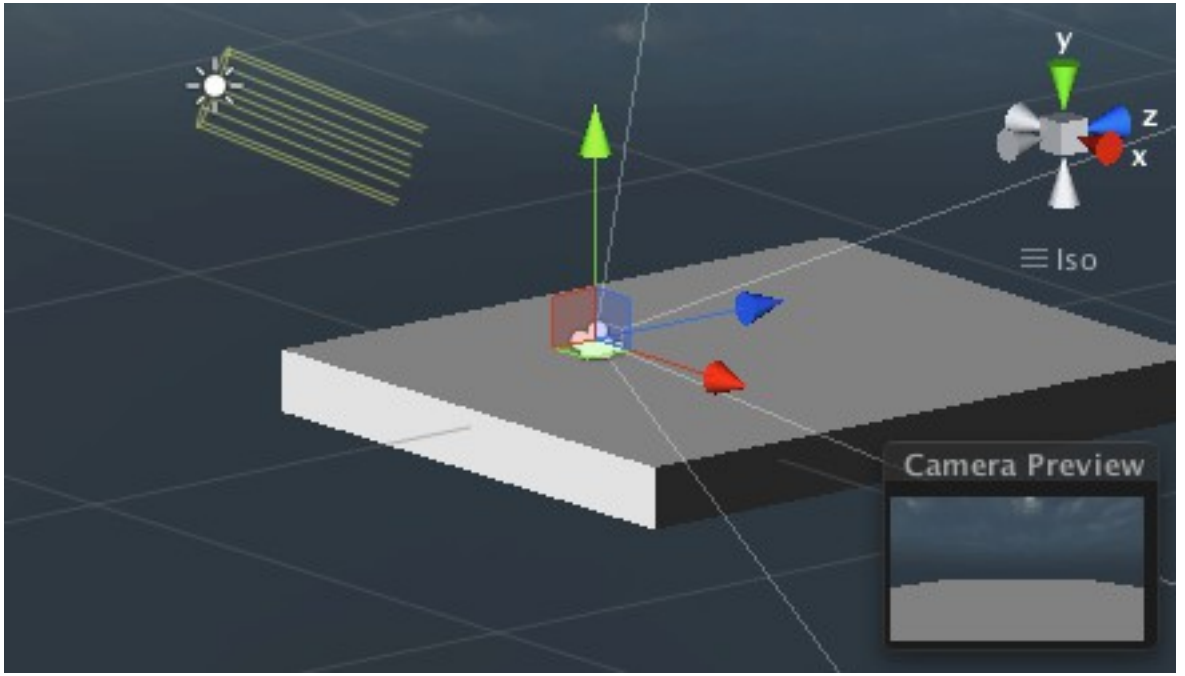
Creating stuff



Add stuff to the game.

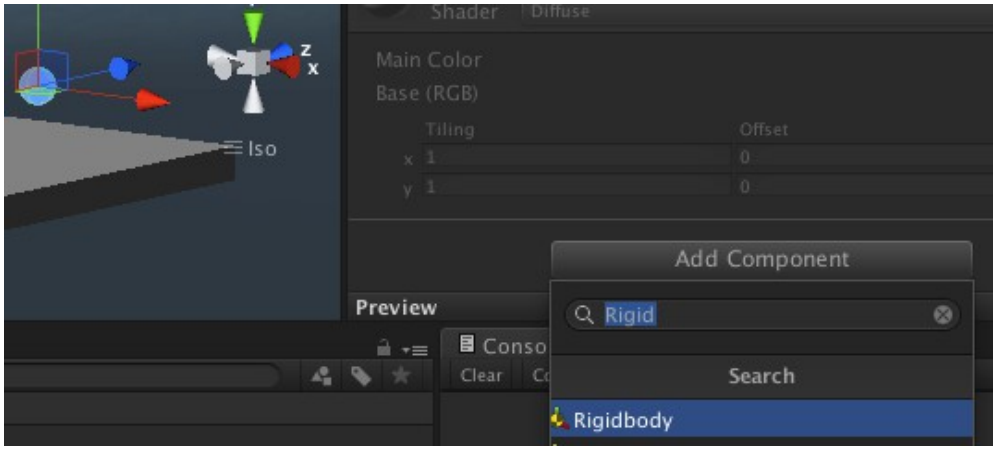
Add a cube and a directional light

And make sure they're both in the camera



UW GAME DEV CLUB

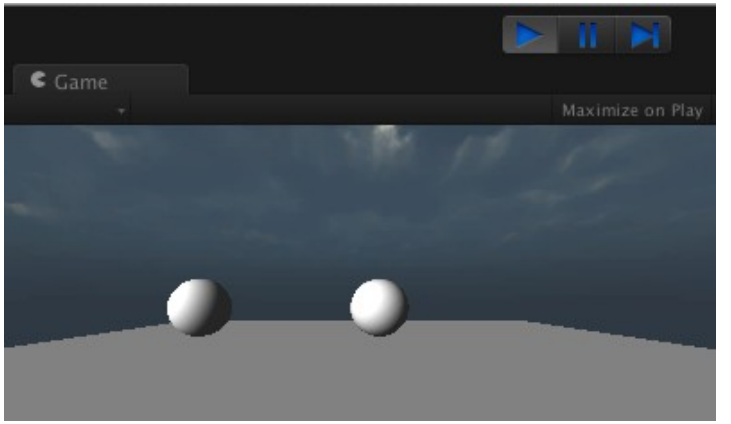
Using the internal physics engine

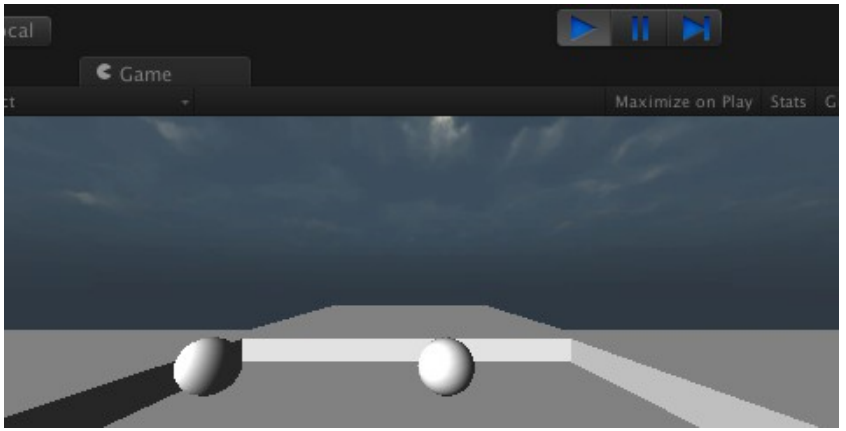


We'll use the internal physics engine to power our game.

Make a sphere, and give it a “Rigidbody” component

Click play to watch the physics simulation

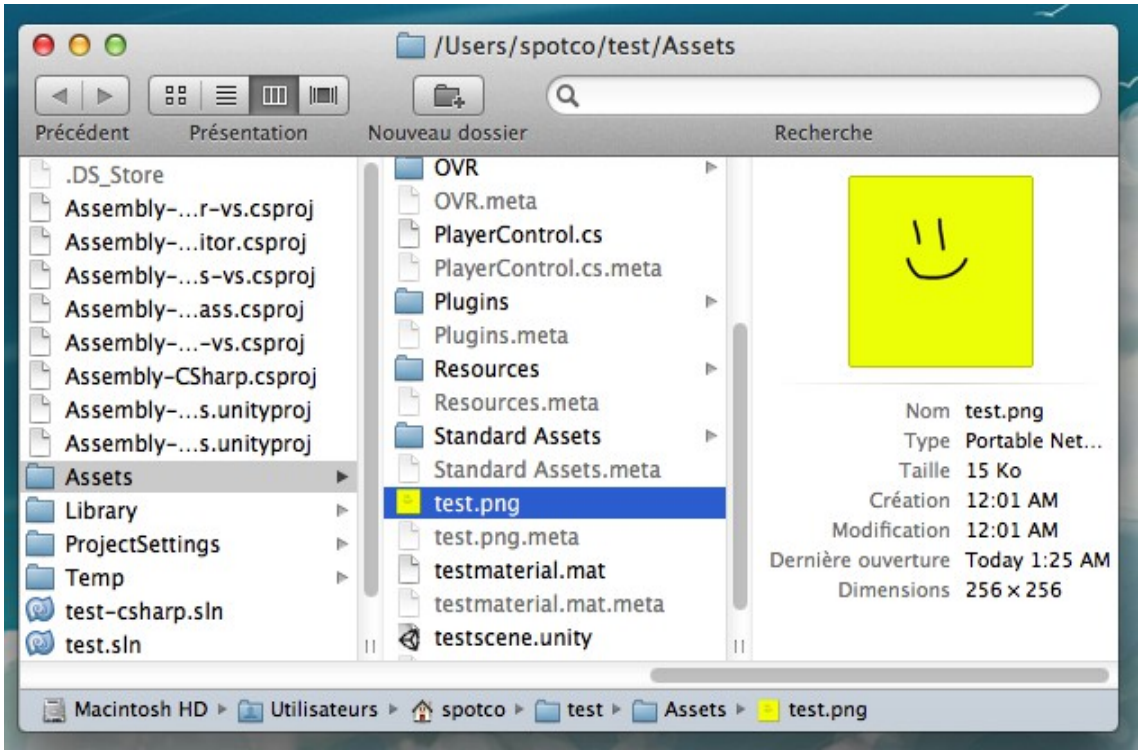




Duplicate some more boxes to keep the balls from falling off

UW GAME DEV CLUB

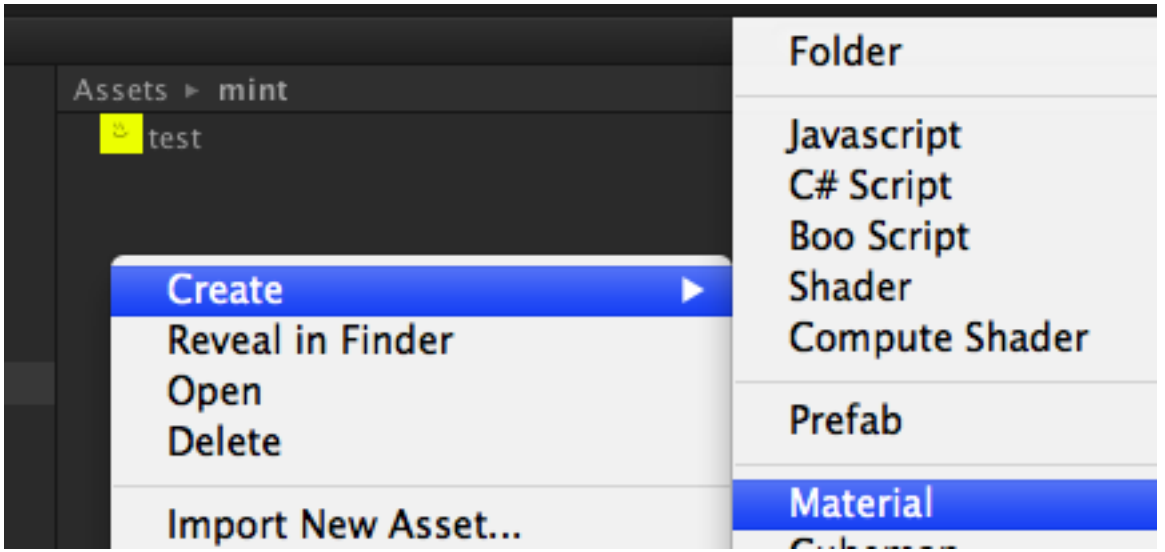
Texturing stuff



Get an image file and put it into the “Assets” folder

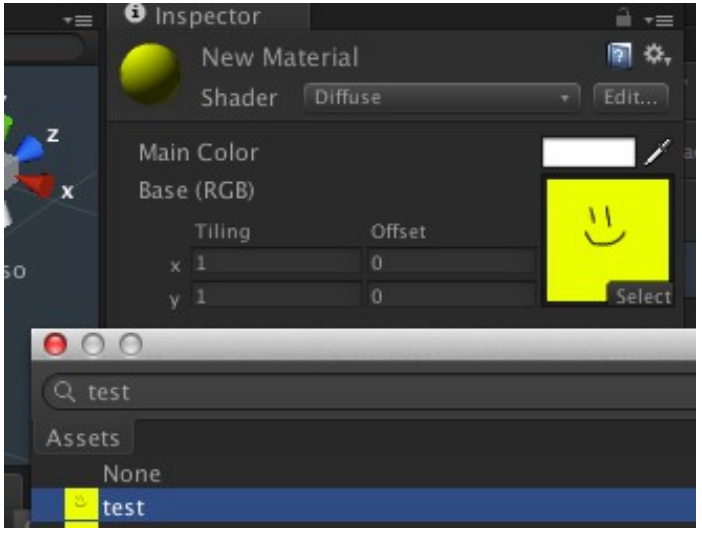
Refresh to see the new file. Then, create a “Material”

(Think of this as a texture + shader combo)

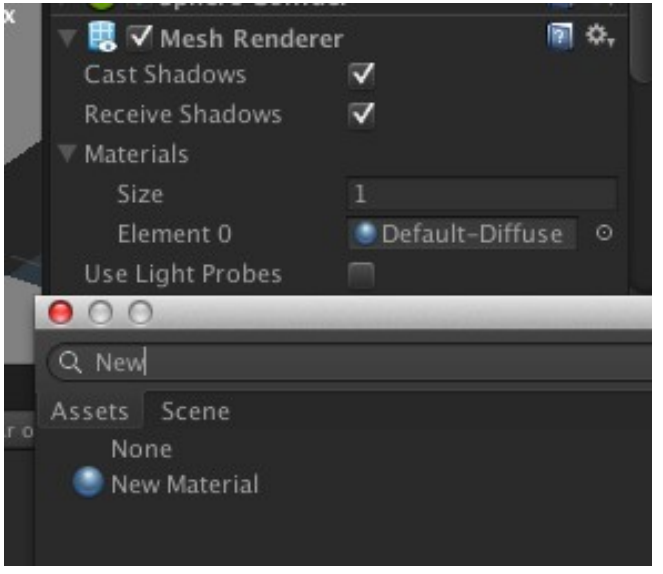


UW GAME DEV CLUB

Texturing stuff



Select diffuse shader, and then select your image as the texture

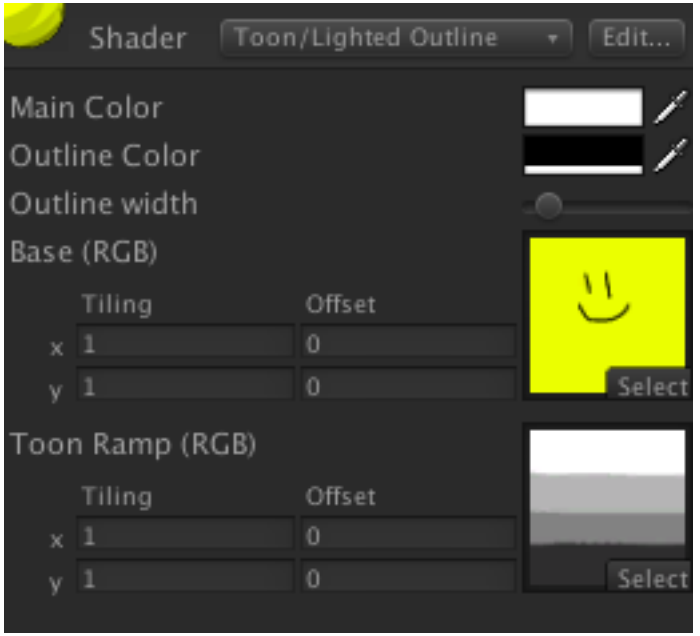


In the Mesh Renderer component of selected object, set “Element 0” of Materials to your new Material.

Quality!

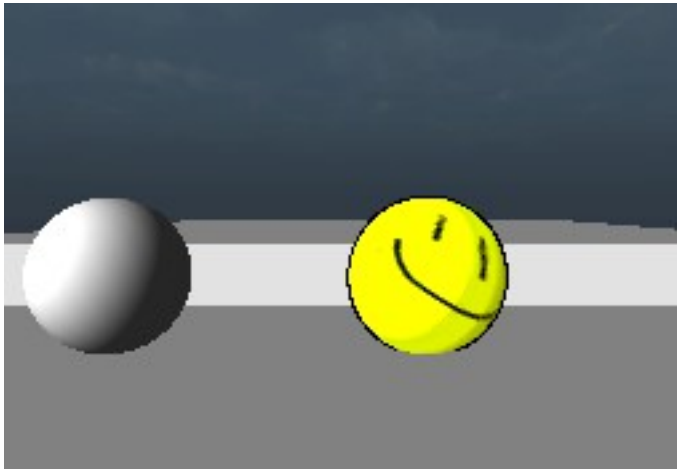
UW GAME DEV CLUB

Cel Shading



For extra fun, set your shader to to “Toon”

Give it a ramp image like the one I'm using (Think of the ramp image as how many possible “bands” of shading color the cel-shaded object can have)

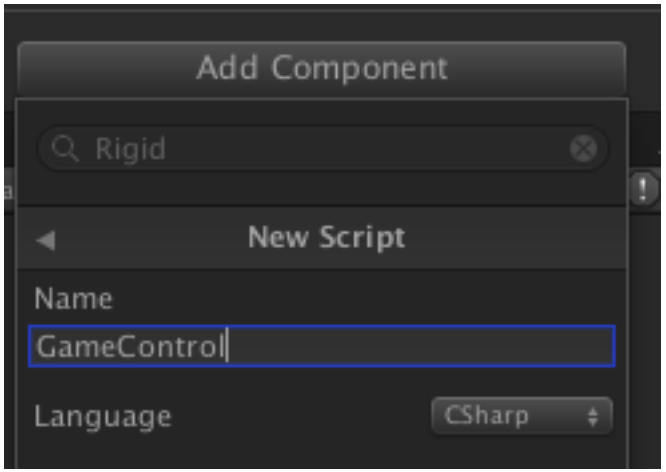


Extra Quality!

UW GAME DEV CLUB

Adding a custom component

Time to write a tiny bit of code to move the ball around. This'll be very similar to other “game control” code you've written (but in 3d)



Add a new component onto your sphere

All running “code” in unity is a component.
Components must be attached to an object in order to run.

```
public class MyComponent : MonoBehaviour {
     void Start () {}
     void Update () {}
}
```

All components are subclasses of “MonoBehaviour”
Override two methods, Start (called when object with this component is created) and Update (self explanatory)

UW GAME DEV CLUB

Unity Component Coding

Fields we care about:

```
this.gameObject
//the object this component is attached to

this.gameObject.transform.position
//the position of the object this component is attached to

his.rigidbody
//the rigidbody component of the object this component is attached to (if it exists)
```

To get keyboard-controlled movement, we'll modify the rigidbody's velocity (to play nice with the physics engine)

```
Vector3 vel = this.rigidbody.velocity;

if (Input.GetKey(KeyCode.W)) {
     vel.z = 1.0f;

}  else if (Input.GetKey(KeyCode.S)) {
     vel.z = -1.0f;

}  else {
     vel.z *= 0.99f;
}

this.rigidbody.velocity = vel;
```

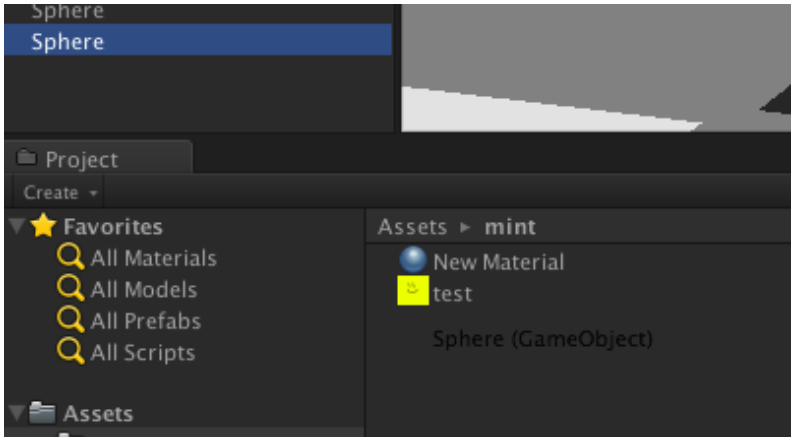
Repeat for x direction (keep in mind, y direction is up)

http://docs.unity3d.com/Documentation/ScriptReference/MonoBehaviour.html

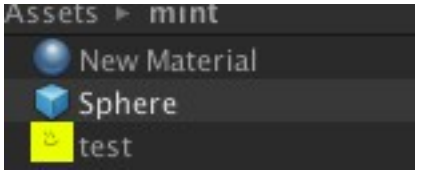
UW GAME DEV CLUB

Prefabs

Say we want to spawn multiple players (controlled the same way).



Drag any object from the “Hierarchy” to the “Project” navigator.



This'll create a “Prefab” copy of that object.

Think of a “Prefab” as an exact copy of the cloned object:
All the same initial transforms, components, component fields, etc.

You can spawn a “Prefab” by either dragging and dropping a prefab onto the “Hierarchy” or through code:

```
Instantiate(Resources.Load("player"))
```

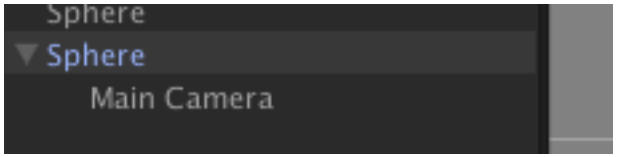
(Assuming you have a prefab in the “Assets/Resources” folder)
(Note the “Assets/Resources” folder (if you make it) are all assets loadable at runtime)

Where should we be placing this bit of code to spawn another player when Q is pressed?

UW GAME DEV CLUB

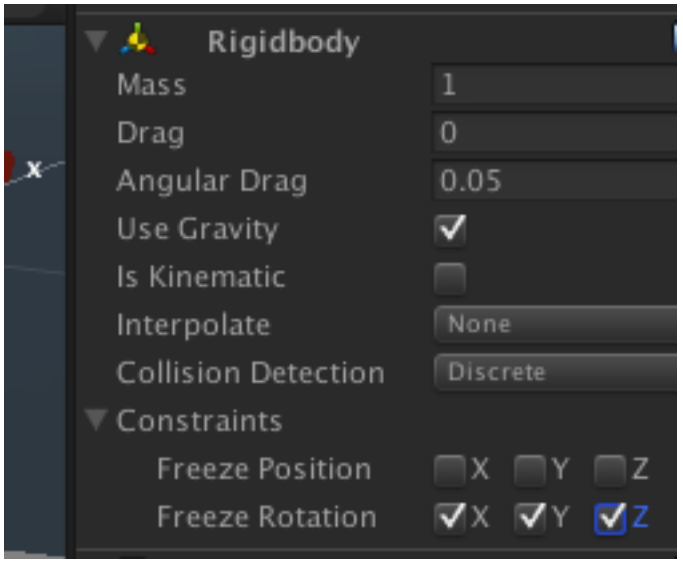
Transform Hierarchy

Say we want the camera to follow the player. How do we do that?



Drag and drop the camera “into” the player object

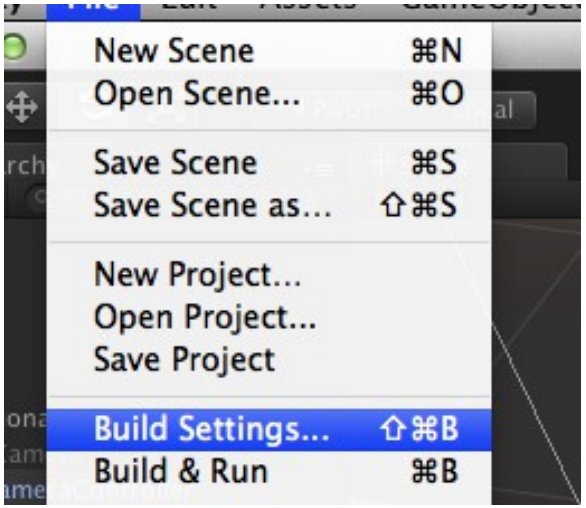
Any transforms on the parent object will get applied to the children. It's a scene graph!



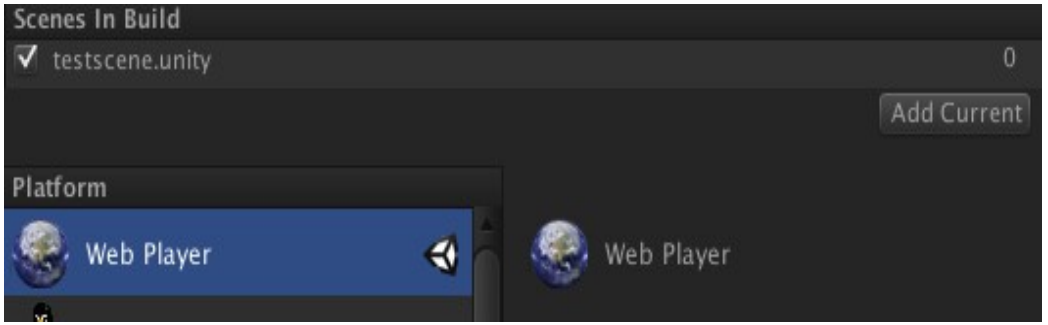
(If this is making you dizzy enable the freeze rotation in the rigidbody)

UW GAME DEV CLUB

Exporting



File->Build Settings->Add Current (Only need to do this once), Build



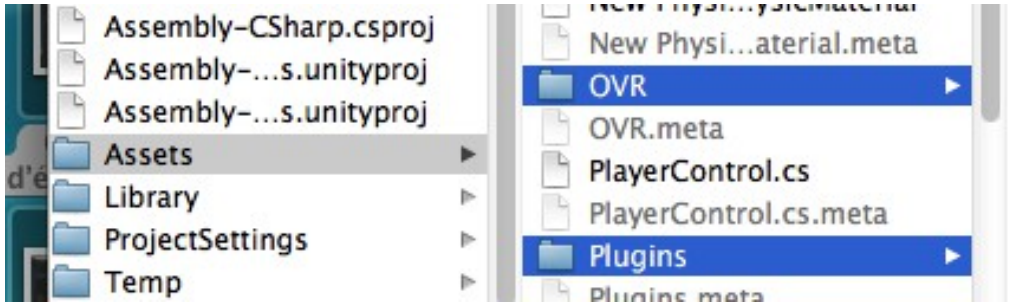
Mobile builds will need some modifications (Obviously Input.GetKeyDown won't work on an iPhone).

UW GAME DEV CLUB

Bonus: Oculus Rift Integration (Unity Pro only)

http://oculusrift.com/sdk/

Go here, make an account, download the Unity SDK

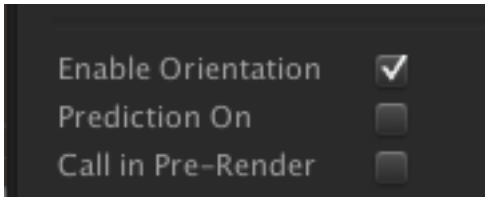


Drag the OVR and Plugins folder into Assets

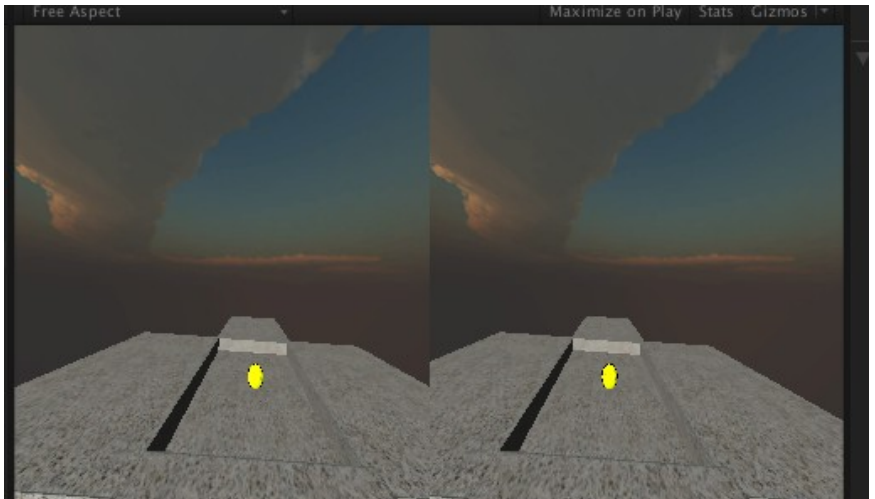
UW GAME DEV CLUB

Bonus: Oculus Rift Integration (Unity Pro only)

Replace your MainCamera with the "OVR/Prefabs/OVRCameraController" prefab



Check "Enable Orientation" in the "OVRCameraController" component



WOW