Chunity Reference

For the most up-to-date information, please visit http://chuck.stanford.edu/chunity/.

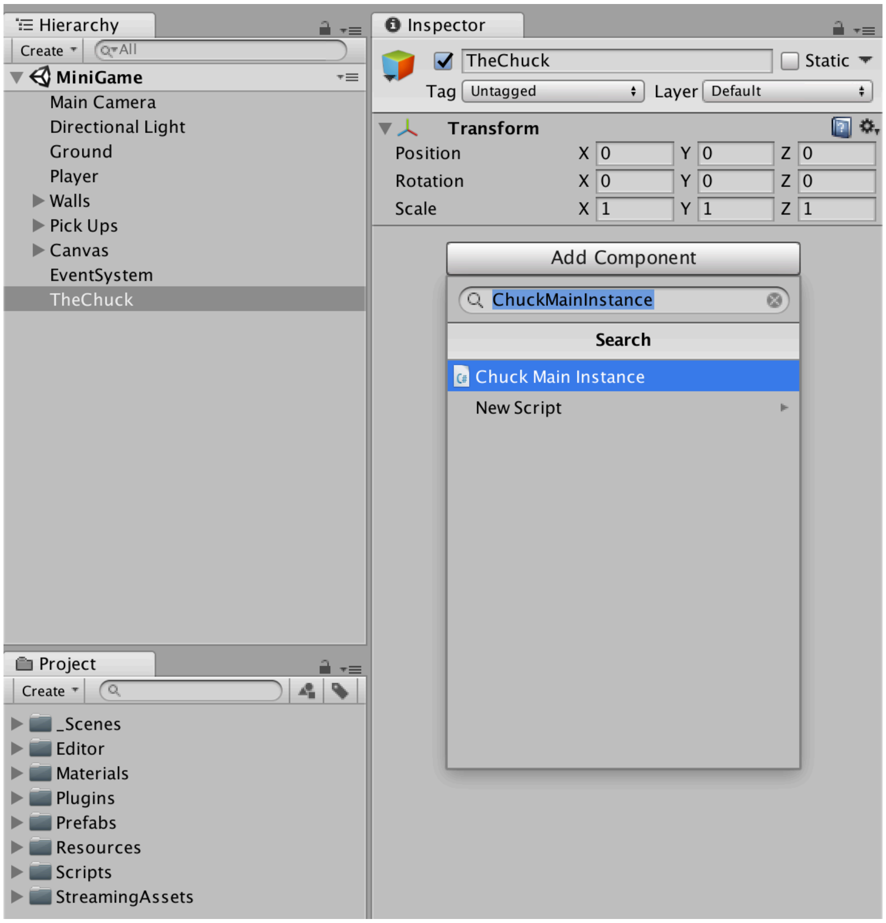
For the full set of tutorials, please visit http://chuck.stanford.edu/chunity/tutorials/.

Initial Setup:

This tutorial will cover how to add ChucK to an individual game object.

First, we need to add a global game object to hold our scene's ChuckMainInstance. All objects that play ChucK code will contain a reference to this object. (It's possible to have as many of these in a scene as you need, but in practice, this is only for special circumstances like accessing two different microphones.)

Here, I have created a new empty game object called TheChuck and I am adding the ChuckMainInstance component to it.



☑ TheChuck □ Static

Tag Untagged Layer Default

Transform

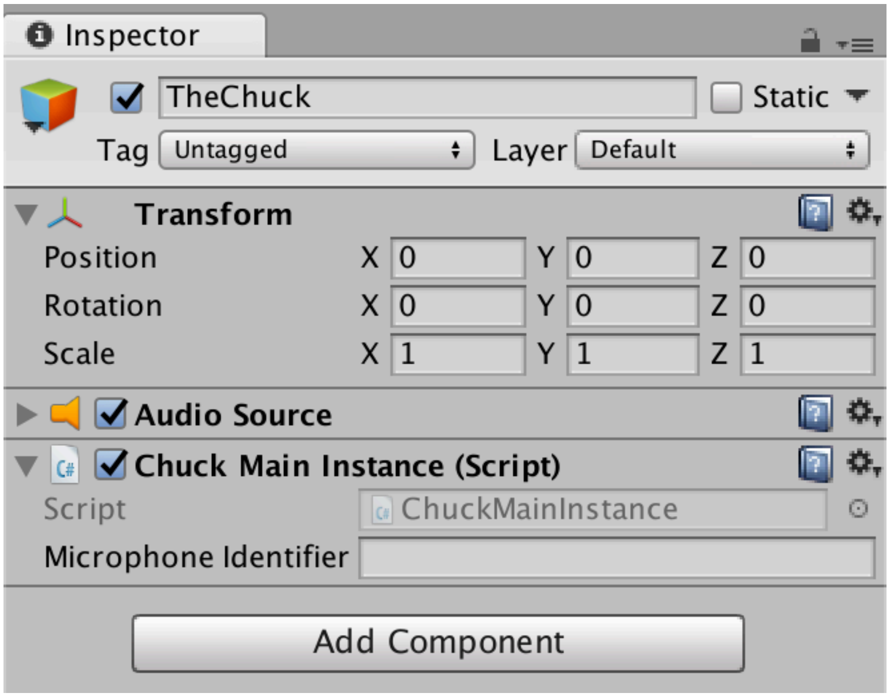
Position	X	0	Y	0	Z	0
Rotation	X	0	Y	0	Z	0
Scale	X	1	Y	1	Z	1

Add Component

ChuckMainInstance

Doing so will automatically add an Audio Source to the object. This is used to fetch the microphone input correctly. You can ignore it. (If you want to play audio files in ChucK, you should not use this Audio Source but should instead follow the directions in the Audio Files tutorial.)

You can also ignore the Microphone Identifier field, unless you want to use a microphone different than your system's default microphone. (In that case, put some substring of the name of the microphone into this field.)



☑ TheChuck □ Static

Tag Untagged Layer Default

Transform

Position X 0 Y 0 Z 0

Rotation X 0 Y 0 Z 0

Scale X 1 Y 1 Z 1

☑ Audio Source

☑ Chuck Main Instance (Script)

Script ChuckMainInstance

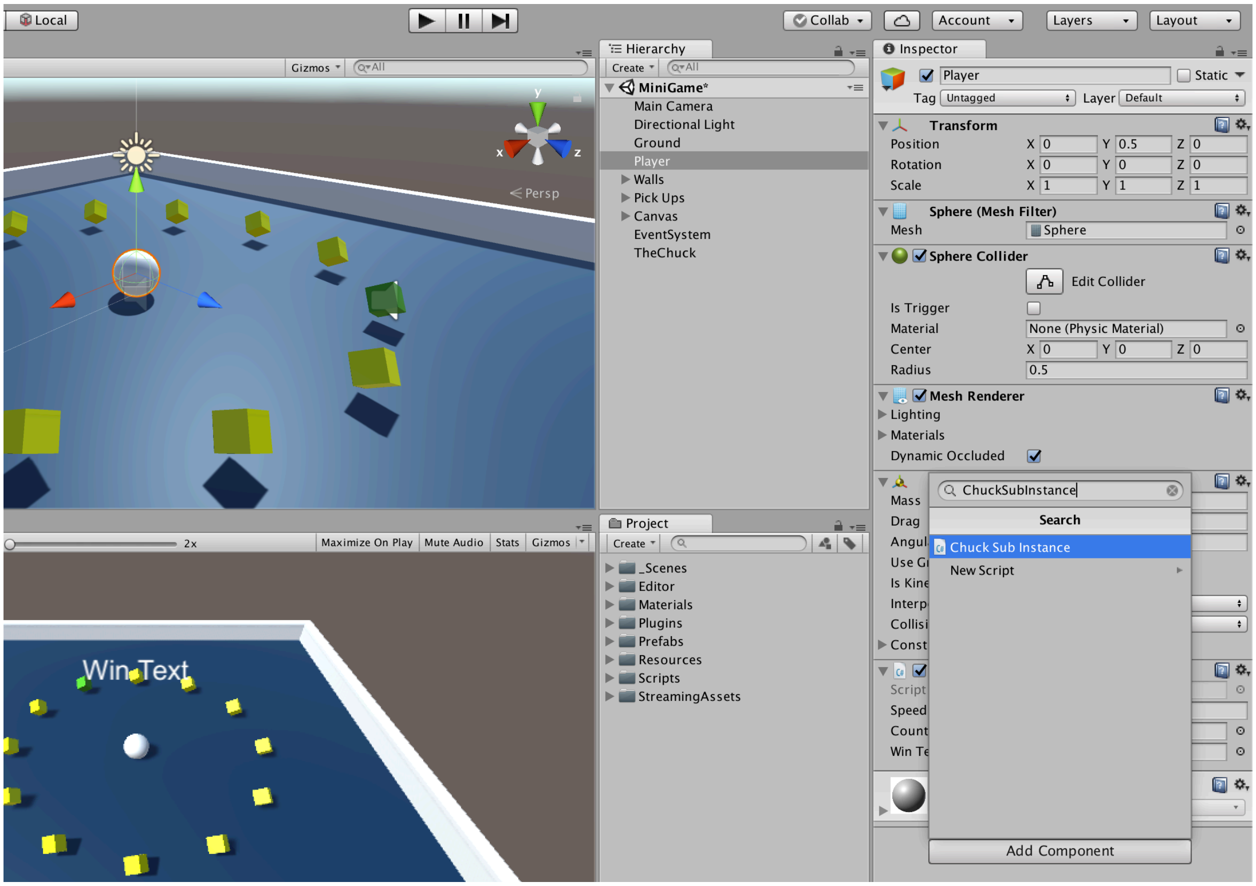
Microphone Identifier

Now that we have created a ChuckMainInstance for our scene, we can begin creating ChuckSubInstances on game objects in our scene.

Pick the game object you want to add ChucK to. In this case, I will add it to my Player object.

Add a ChuckSubInstance script to this game object.

By adding this script, an Audio Source will also be added to your game object. This Audio Source is used by ChucK to function correctly. (If you want to play audio files in ChucK, you should not use this Audio Source but should instead follow the directions in the Audio Files tutorial.)



Inspector

☑ Player ☐ Static

Tag Untagged Layer Default

Transform

Position X 0 Y 0.5 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

Sphere (Mesh Filter)

Mesh Sphere

☑ Sphere Collider

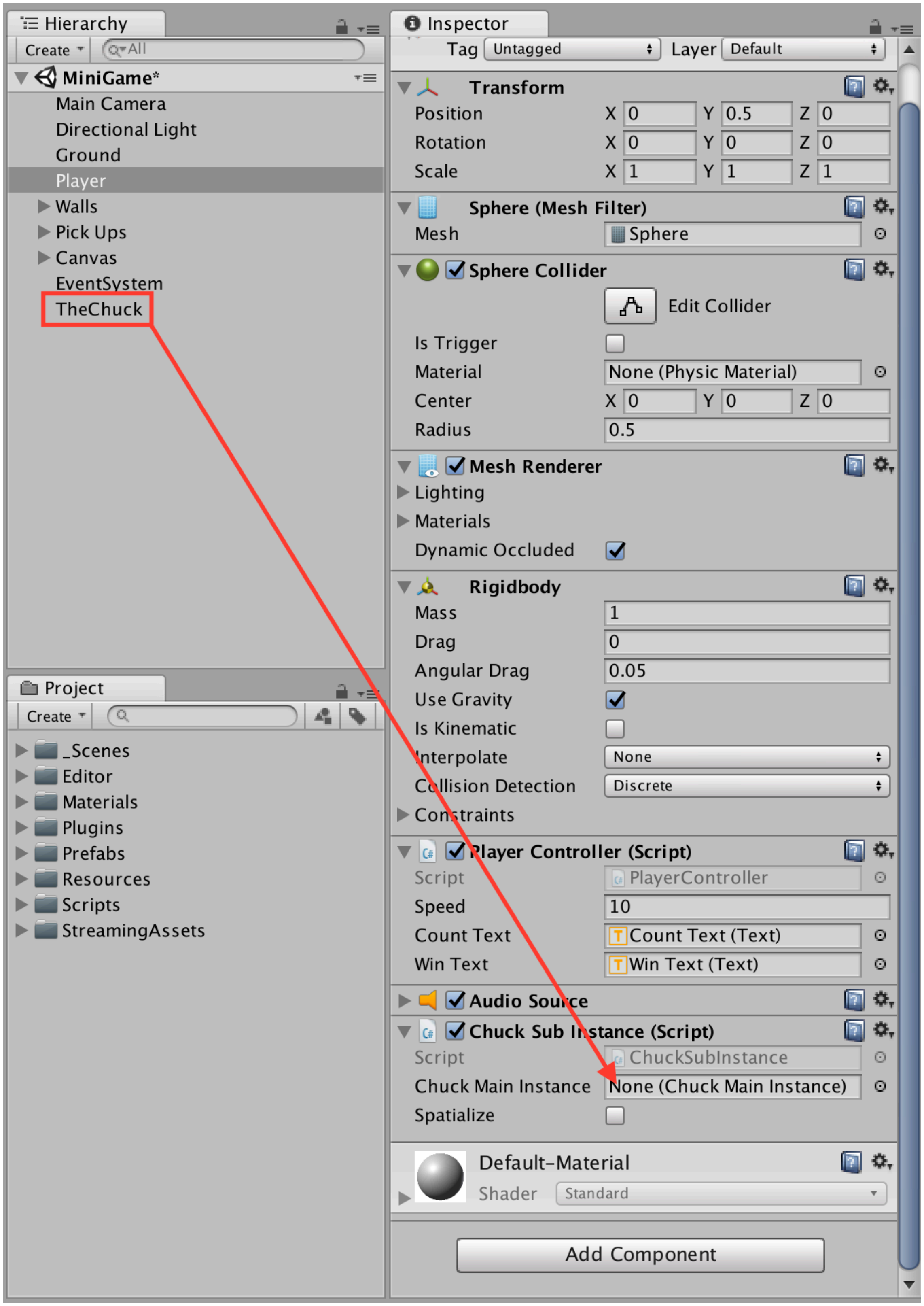
Is Trigger ☐
Material None (Physic Material)
Center X 0 Y 0 Z 0
Radius 0.5

☑ Mesh Renderer

Dynamic Occluded ☑

ChuckSubInstance

The next step is to give this component a reference to the ChuckMainInstance we created earlier. Drag it into the Chuck Main Instance slot on the new ChuckSubInstance component.



Tag Untagged Layer Default

Transform
Position X 0 Y 0.5 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

Sphere (Mesh Filter)
Mesh Sphere

☑ Sphere Collider
Is Trigger □
Material None (Physic Material)
Center X 0 Y 0 Z 0
Radius 0.5

☑ Mesh Renderer
Dynamic Occluded ☑

Rigidbody
Mass 1
Drag 0
Angular Drag 0.05
Use Gravity ☑
Is Kinematic □
Interpolate None
Collision Detection Discrete

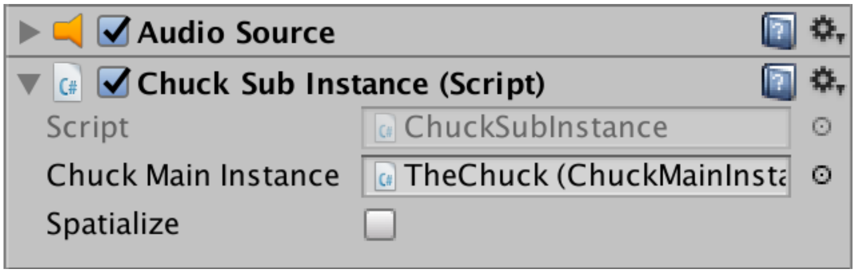
☑ Player Controller (Script)
Script PlayerController
Speed 10
Count Text Count Text (Text)
Win Text Win Text (Text)

☑ Audio Source

☑ Chuck Sub Instance (Script)
Script ChuckSubInstance
Chuck Main Instance None (Chuck Main Instance)
Spatialize □

Default-Material
Shader Standard

If you are using ChucK spatially, you can edit the 3D Sound options of this Audio Source. Since we are not using this ChucK spatially, we can leave all the settings alone. Now we are done adding a ChucK Instance to our scene!



☑ Audio Source

☑ Chuck Sub Instance (Script)

Script: ChuckSubInstance

Chuck Main Instance: TheChuck (ChuckMainInsta

Spatialize: □

You can access this component as you would any other component, e.g.

GetComponent<ChuckSubInstance>().RunCode(@“…”);

For example:

```
void Start() {
        rb = GetComponent<Rigidbody>();
        count = 0;
        SetCountText();
        winText.text = "";
        GetComponent<ChuckSubInstance>().RunCode(@"
                SinOsc foo => dac;
                while( true )
                {
                        Math.random2f( 300, 1000 ) => foo.freq;
                        100::ms => now;
                }
        ");
 }
```

You're now set up to run ChucK code in Unity! Please refer to the tutorials: http://chuck.stanford.edu/chunity/tutorials/

and documentation: http://chuck.stanford.edu/chunity/documentation/