

<http://gmv.cast.uark.edu> A Method Store for Advanced Survey and Modeling Technologies Mon, 01 Apr 2013 03:29:18 +0000 en-US hourly 1 <http://wordpress.org/?v=3.5.1>

<http://gmv.cast.uark.edu/modeling/software-visualization/unity-software-visualization/workflow-unity-software-visualization/unity-pro-vs-unity-indie/> <http://gmv.cast.uark.edu/modeling/software-visualization/unity-software-visualization/workflow-unity-software-visualization/unity-pro-vs-unity-indie/#comments> Sat, 11 Aug 2012 11:53:42 +0000 Keenan <http://gmv.cast.uark.edu/?p=11463> [Continue reading →](#)]]>

Unity Indie is a free version of the Unity3D software. It allows anyone to use a modern advance game engine to create interactive realtime 3D visualizations for Windows, Mac, Web player and soon, Linux.

The chief differences between Unity Indie and Unity Pro lies in the ability to optimize your scenes and the ability to create scable worlds. Its significant to point out that most of these features in Unity Pro can be imitated to a certain degree in Indie using code and modeling software.

Intro to Static Batching

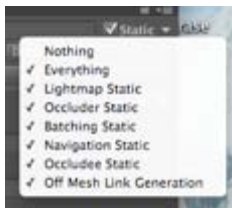
Graphics cards can process many polygons with relative ease. The texture mapped onto those polygons tend to be the source of rendering and performance problems. Whenever you test out your Unity application by hitting play, Unity renders on screen everthing that is aligned with the camera (or view frustrum). Every object using a different material within the field of vision gets sent to the graphics card for processing. Unity does this in passes. Everything in the back is rendered first working its way to the front. Every object getting sent one by one to the GPU is a fairly inefficient procedure. If there are a number of individual models pointing to the same material, Unity can combine these models at runtime reducing the amount of data that is sent to the GPU. Unity in effect renders or “draws” these combined meshes in one batch or “call”. Unity does this to some degree already with dynamic batching (included in the indie version). But the most significant gains come when you can tell Unity which models you’d like grouped together. This is where static batching comes in (only available in the Pro version).

Tag a Static Batch Object

When you tag an object as static batched, then Unity will group that object in with other objects that share the same material.

Go to Edit > Project Settings > Player

Check Static Batching



Specify the models

Now you'll want to tell Unity which models should use static batching. Remember, static batching is only effective on models that share the same texture and material. Duplicates of objects and models using texture atlases are your prime targets.



Static Batching Statistics

-Select a prefab in the Project panel or select a GameObject in the Hierarchy.

-In the top right corner of the Inspector, click on the triangle next to Static. A drop down menu appears. Select Batching Static. We'll talk about some of the other static options in a moment.

-When you click play, you can see how many models are being batched at any given time by click on the Stats button in the upper right corner of the Game panel. You should see a decrease in the Draw Calls also when you enable Static Batching.



Introducing Occlusion Culling

I mentioned above that Unity renders everything in the back first and then everything towards the camera. This means that if you look at a wall in Unity everything behind that wall is getting rendered as well and affecting performance. This seems inefficient and impractical. You could have a sprawling city behind that wall and have it affecting your performance. This is where occlusion culling comes in.

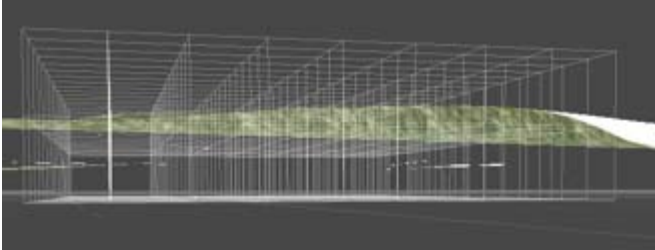
Occlusion culling keeps track of what is visible from any given location via a 3D grid of cells called Potentially Visible Set (PVS). Each cells contains a list of what other cells are visible and which are not. Using this information, Unity can render only that which is visible significantly decreasing the amount of data that needs to be processed to render the scene.

Setting up occlusion culling is fairly straightforward, but keep in mind, depending on the scale of your scene, the baking process could take from 30 minutes to a couple of hours.

Setting up Occlusion Culling

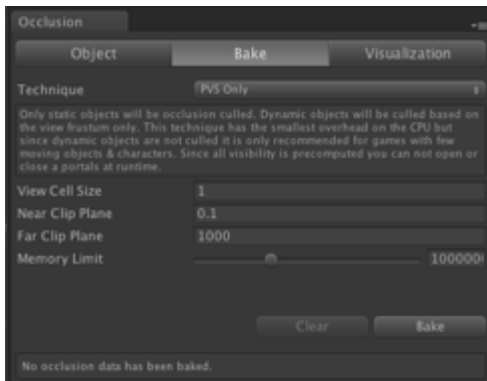
1. Select all the stationary GameObjects
2. In the top right corner of the Inspector, click the triangle next to "Static"
3. Select "Occluder Static"
4. Drag and drop your First Person Controller into the scene.
5. Now go to Window > Occlusion Culling

A new window opens called "Occlusion" along with a 3D grid in the scene view. The 3D grid represents the size of the cell grid the OC process will use. As it stands, the entire scene will be used in occlusion culling.



Bake Occlusion Culling

6. Click on the "Bake" tab. You'll see the settings for the Occlusion Culling.



7. Click on "Bake" and go get some lunch. When the baking finishes, click Play and make sure you do not have geometry that suddenly appears and disappears. You can move the Game view so that it is adjacent to the Scene view. You can see how only the geometry you are viewing in the Game view is the only geometry visible in the scene view and whenever you rotate and move the geometry appears and reappears as needed. This should drastically reduce any frame rate issues.

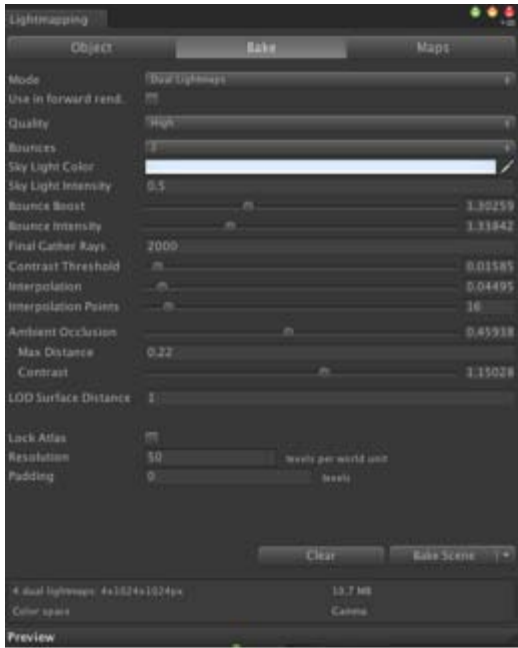
Introducing Lightmapping

When you have lights casting realtime shadows in your scene, the frame rate can be greatly affected as every pixel of an object reflecting light has to be calculated for the right affect and the shadows have to be calculated as well. Lightmapping is a process that "bakes" or draws the shadows and lighting effects onto the textures of an object so that at runtime no light calculations need to be processed. An additional advantage is also that lightmapping can add ambient occlusion to a scene that adds an element of realism and softness to the scene. The setup is similar to Occlusion Culling, but it can take even longer. This is very much an all night process, so be sure to start it before leaving the day, if the scene is large or uses many materials.

As with everything else, you must tag the objects you want to have a lightmap for.

Setting up Lightmapping

1. In the top right corner, click on the triangle next to "Static" and check "Lightmap Static"
2. Go to Window > Lightmapping
3. Click on the Bake tab. The settings for lightmapping appear and at first glance they are daunting. For most situations though, the default settings should do fine. The main thing to look for is to see that "Ambient Occlusion" is at least .4 if you want to add Ambient Occlusion to your scene.
5. Click Bake and call it an evening (depending on the scene size)



]]> <http://gmvcast.uark.edu/modeling/software-visualization/unity-software-visualization/workflow-unity-software-visualization/unity-pro-vs-unity-indie/feed/> 0 [http://gmvcast.uark.edu/modeling/software-visualization/unity-software-visualization/creating-a-terrain-in-unity-from-a-dem/](http://gmvcast.uark.edu/modeling/software-visualization/unity-software-visualization/workflow-unity-software-visualization/creating-a-terrain-in-unity-from-a-dem/) <http://gmvcast.uark.edu/modeling/software-visualization/unity-software-visualization/workflow-unity-software-visualization/creating-a-terrain-in-unity-from-a-dem/#comments> Sat, 11 Aug 2012 11:31:55 +0000 Keenan <http://gmvcast.uark.edu/?p=11449> [Continue reading →](#)]]>

Getting DEMs translated into a form Unity understands can be a bit tricky, and as of yet no perfect solution exists. Nevertheless, DEMs are excellent ways to acquire terrain and elevation models for a variety of purposes. Racing games created in Unity make extensive use of DEMs for levels as well as architectural visualizations. Beware though that accuracy tends to be an issue.

There are two main ways to translate DEM data into Unity. One way is to convert a DEM into a Raw heightmap. The other is to convert a GridFloat dem into a Unity terrain

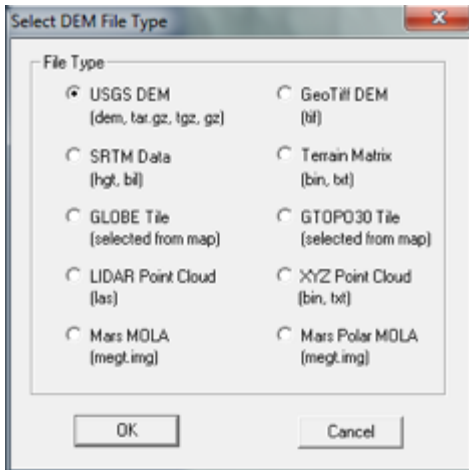
Using Terragen/3DEM

The goal of this method is to convert the DEM into a RAW heightmap that Unity reads natively. For many users of Unity, the most cost effective solution is to use a two step process using Terragen and 3DEM, but if you can obtain a heightmap from DEM by any other means, you can skip to the Import into Unity section.

Convert DEM to Terragen File

The first step is to convert the DEM into a Terragen file. We use the free program 3DEM, which you can download [here](#).

1. Open 3DEM.
2. Choose the format of the DEM in question



Define the export area

3. Select an area you wish to export.



Extract the export extents

4. Here's the tricky part. The selection box does not give us any information concerning the width, length and elevation of the selected area and Unity will need this information. We will have to point our cursor in several spots inside the selection box to extract this data. When you move the cursor, the Northing, Easting and Elevation info appears in the lower right hand corner.

Zone	North	East	Elevation
15	3985118 m	405062 m	331.1 m

We can use this to extract the width of the selection box. If we point the cursor on the left edge of the box and then the right edge, we subtract the value from each and find the width of the box. Similarly, if we find the lowest elevation and subtract it from the highest elevation, we have our elevation height that Unity will need.

6. File > SaveTerragen Terrain > Selected Area and save it in an accessible area

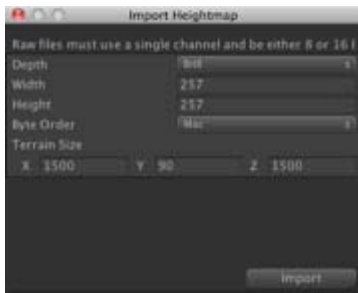
Export from Terragen to RAW

- Now open Terragen
- Open the .ter file you exported from 3DEM. The area you selected should now appear in the Landscape and the Rendering Control dialog.
- Go to Export > Terrain > Export Make sure it is 8 bit Raw.



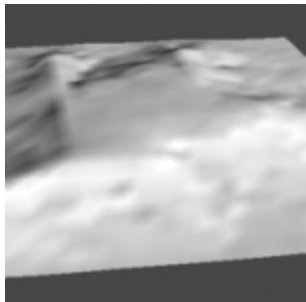
Import into Unity

- Now open Unity
- Go to Terrain > Create Terrain. A large terrain will appear in the scene view.
- Go to Terrain > Import Height – Raw... and select the raw file we exported from Terragen
- The Import Height dialog appears. You'll need the data we collected from step 4 and enter it into the correct slots. The width and height of the selection box go into the X and Z. The elevation difference goes into Y. Check and make sure the Depth is 8bit, (if you used Terragen) and that the Byte order is your respective OS. Click OK.



The result

Unity will create the terrain from the DEM. In the scene view, the terrain may appear pixelated with jagged edges, but this is mainly due to the Scene view using LOD to render the terrain. If you drop a First Person Controller onto the terrain and hit play, the terrain will appear smooth.



Using GridFloat format

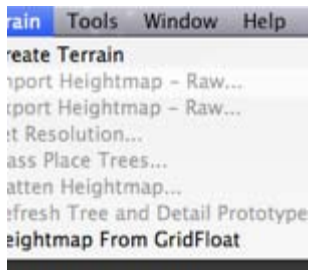
If the DEM is in the GridFloat format, then you can enable Unity to read into a Unity terrain. You'll need to download the script to do this from [here](#). Be sure the header file (HDR) for the GridFloat DEM also resides in the Asset folder.

1. Create a folder in Unity called 'Editor'
2. Copy the HeightmapFrom GridFloat to the Editor folder.

Placing scripts into an "Editor" folder in Unity actually extends the Unity editor. Now a menu will appear underneath Terrain at the top called 'Heightmap From GridFloat'

Create the terrain in the scene

3. Select the .flt file in the asset folder. (Be sure you have the hdr file as well)
4. Go to Terrain > Heightmap From GridFloat.



The terrain should appear in the scene view.

]]> <http://gmvcast.uark.edu/modeling/software-visualization/unity-software-visualization/workflow-unity-software-visualization/creating-a-terrain-in-unity-from-a-dem/feed/> 0
<http://gmvcast.uark.edu/modeling/software-visualization/unity-software-visualization/workflow-unity-software-visualization/basic-interaction-and-scripting-in-unity/> <http://gmvcast.uark.edu/modeling/software-visualization/unity-software-visualization/workflow-unity-software-visualization/basic-interaction-and-scripting-in-unity/#comments> Sat, 23 Jun 2012 12:01:35 +0000 Keenan <http://gmvcast.uark.edu/?p=10291>

This series of posts will teach you how to navigate in Unity and to create a basic virtual museum.

Hint: You can click on any image to see a larger version.

Get Some Scripts

This tutorial will describe a basic setup for adding simple interactions to the walkable demo we created in the previous tutorial. To do this, you'll use several scripts contained in the package you can download [here](#).

Have Unity open with the project used in the last tutorial

Download it and double click on it with Unity open.

Import all the scripts.

Script Documentation

The scripts are documented, if you would like to view them and learn more. You can find additional resources concerning the Scripting API in Unity at the [Unity Script Reference page](#).

Unity Scripting Languages

Unity allows you to create in scripts in three languages: JavaScript, Boo, and C#.

JavaScript is well known for web scripting and is part of the same standard as Flash's ActionScript. Unity's JavaScript has notable differences, but it provides the easiest way to interface with GameObjects.

C# is native to .NET and the Mono project, which Unity is based on. It provides many advance options that JavaScript lacks.

Boo is a lesser known language. Not many people code in Boo, but it has many similarities to Python.

The Museum Scripts

The package contains three scripts:

ArtifactInformation.js : This is a basic data repository script. When you add it to a GameObject, you'll be able to enter basic information about the object in the inspector that can be shown when the user clicks on the object. Place it on each artifact you wish to interact with.

We could obtain the information in a number of ways, MySQL database, XML, etc. For this demo, we are mainly worried with what to do with the data once we receive it.

ObjectRotator.js : This script allows us to rotate the object when we click on it. It also needs to be attached to each object you want to interact with.

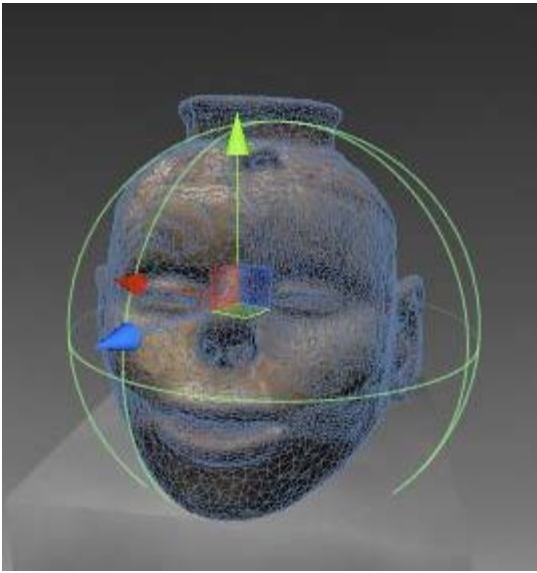
ArtifactViewer.js : This script is attached to the FPC's main camera. This script will detect if we are looking at an object when we left mouse click. If an object is in front, it will deactivate the FPC, show the information of the object and activate object rotate mode. It must go on the Main Camera inside the First Person Controller.

Setup a Sphere Collider

We need to perform several steps to prepare the script for activation. I assume you have placed an artifact on one of the pedestals in the museum. Be sure to make the scale factor of .005. Otherwise, hilarity will ensue. Leave the Generate Colliders box unchecked.

Add a Sphere Collider

1. With the object selected, go to Components > Physics > Sphere Collider



This allows us to detect if we are in front of the object.

Add New “Artifact” Tag

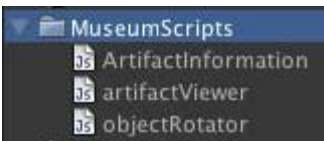
1. With the object selected, go to the top of the Inspector and click the drop down menu next to Tag and click Add Tag.
2. Open the Tag hierarchy and in Element 3, add the text “Artifact”.
3. Go back to the object’s properties in the Inspector, and make sure the newly created “Artifact” tag appears in the Tag drop down menu.



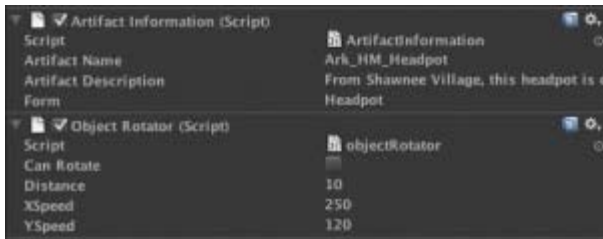
4. Do this for all artifacts you wish to interact with.
5. Select the FPC and assign the “Player” tag to it. The “Player” tag should already exist in Unity

Attach Scripts

1. Find the folder “Museum Scripts” in the Project panel.



2. Drag the scripts “ArtifactInformation” and “ObjectRotater” to each object you wish the user to interact with.



3. The "ArtifactViewer" script has to go in a special place on the FPC. Expand the FPC in the Hierarchy panel and locate the "Main Camera" inside of it.



4. Drag the ArtifactViewer on to the "Main Camera."

Add info about the artifacts

Now if you select the artifact, you'll find where you can enter information about the artifact. You can copy and paste data from the Hampson site or come up with your own.



NB: Although the Artifact Description doesn't show much of the text, it really does have a block of text in it.

Finished Museum Demo

Now if you click Play, you can click on the object and a GUI with information appears. Left clicking will also allow you rotate the model. If the rotation seems off, the pivot point may not be in the center.



You should now have a simple walkthrough demo of a museum.

This project was designed as a launching platform for more ambitious projects.

]]> <http://gmvc.cast.uark.edu/modeling/software-visualization/unity-software-visualization/workflow-unity-software-visualization/basic-interaction-and-scripting-in-unity/feed/> 0

<http://gmvc.cast.uark.edu/modeling/software-visualization/unity-software-visualization/workflow-unity-software-visualization/a-walkable-unity-demo/> <http://gmvc.cast.uark.edu/modeling/software-visualization/unity-software-visualization/workflow-unity-software-visualization/a-walkable-unity-demo/#comments> Sat, 23 Jun 2012 11:30:18 +0000 Keenan <http://gmvc.cast.uark.edu/?p=10259>

This series of posts will teach you how to navigate in Unity and to create a basic virtual museum.

Hint: You can click on any image to see a larger version.

Intro

This chapter will show you how to put together a walkable demo of a simple museum using several models. The techniques in this tutorial can be applied in many ways. Although I use models from the Virtual Hampson museum as examples, you are welcome to use your own.

You will need to download the following model, or use one of your own:

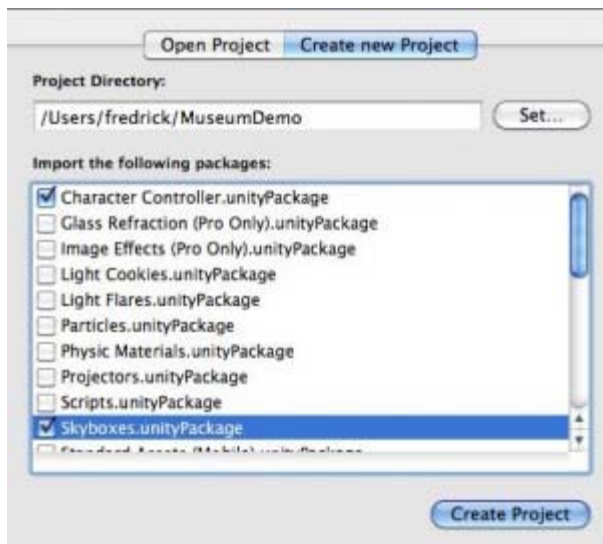
<https://dl.dropbox.com/u/22781413/museumdemo/simpleMuseum.zip>

You can take a look at the final product [here](#).

Create a new project

Open Unity and go to File > New Project.

The New Project file dialog will come up.



Name the project "Museum Demo" and save it in a convenient location.

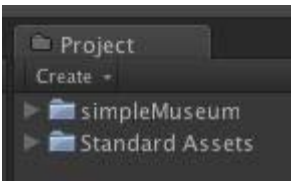
Below you will see the packages Unity comes with. A package is a set of scripts and assets that allow certain functionalities. For this demo, we will choose:

- CharacterController
- Skybox

Unity will open with an empty scene.

Import the model

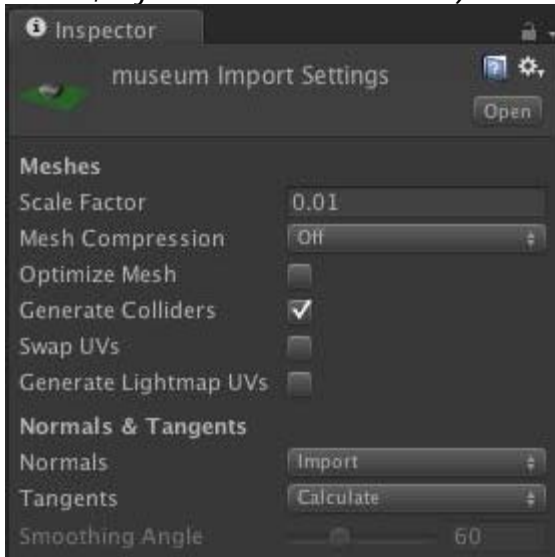
Now we'll import the simpleMuseum model. Either we can copy and paste the file into the asset folder using Explorer or we can drag and drop the file into the Project panel.



Click on the model in the Project panel, and its Import Settings will show up in the Inspector. You'll want to do two things here.

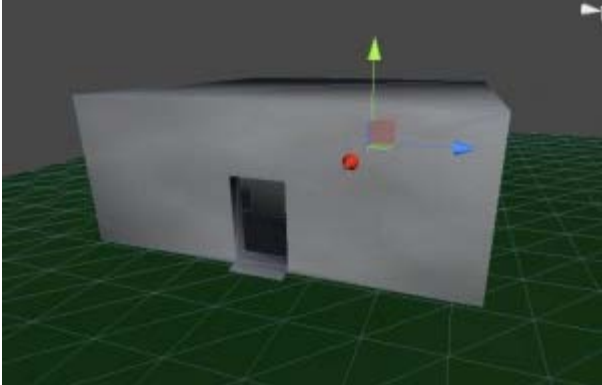
1. Check the Generate Colliders box.

2. Adjust the scale factor. Every modeling application processes units differently. (For the simpleMuseum model, try a scale factor of .01).



Scene View

Now we'll drag and drop the model from the Project panel to the Scene view.

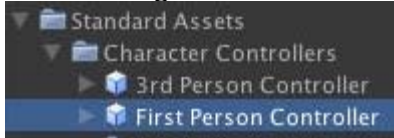


The First Person Controller

It's hard to see if the scale is off without another model for comparison so we'll also drag and drop the First Person Controller into the scene.

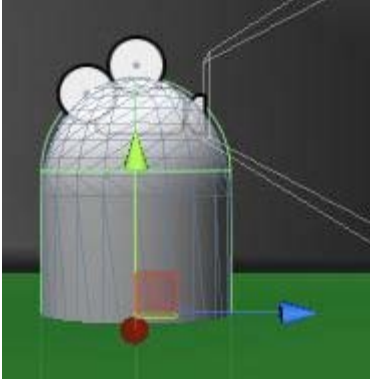
You can find the First Person Controller* (FPC) by locating the Standard Assets folder in the Project panel

and looking in the Character Controllers folder. You will see the 3rd Person Controller and the FPC .

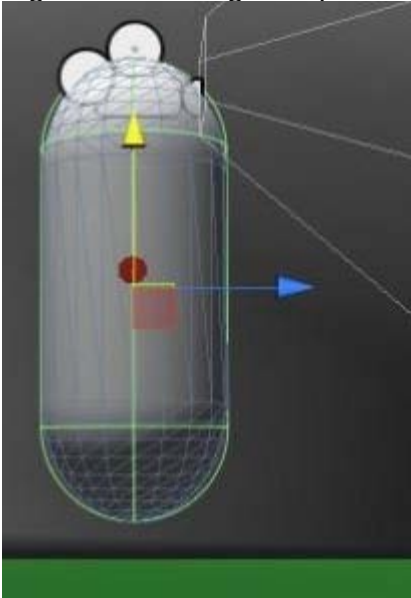


NB: If you do not find the FPC, go to Asset > Import Package > Character Controllers and Import All.

Drag the (FPC) into the scene inside the museum model.



FPC is raised off the floor. If the FPC is partially through floor, the camera will fall to infinity. If the FPC is high above the green plane, it will fall until it collides with it.



FPC and model scale

The FPC is 2 units tall. Unity works best thinking the units in meters, but it mainly depends on the units used in the modeling application. If the model is larger or smaller relative to the FPC, you can adjust the



Scale Factor accordingly.

Preview the Model

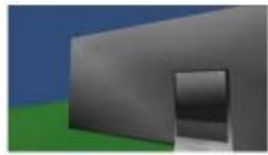
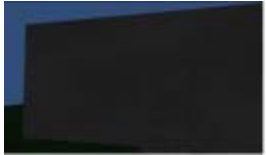
Press the Play button at the top. This will compile the scripts and show you a preview. You can walk around using the arrow keys or WASD. You can look around using mouse movement. You can even jump with space bar. We'll adjust these controls later.



If the camera falls through the model, either the FPC wasn't raised enough off the floor or "Generate Colliders" wasn't checked in the Import Settings.

Scene Lighting

Likely, the preview will be dark and difficult to see clearly. Let's add a light to the scene.



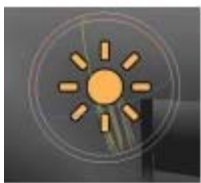
Unity provides three types of lighting typical for 3D applications: Spot, Directional, Point.

Point light is an omnidirectional point of light. Directional light imitates sun light from an angle. Spot light is a concentrated area light. To light an entire scene, we will add a directional light.

Click on GameObject > Create Other > Directional Light.

Edit Unity Lighting

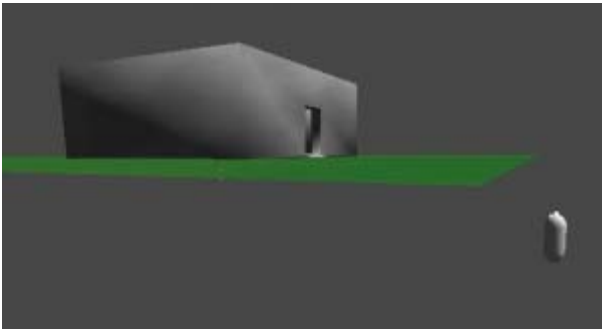
Press E to go into rotate mode and rotate the light to obtain the desired lighting. The position of the light is irrelevant. You can change the color and intensity in the inspector while you have the Directional Light selected.



Invisible Colliders

Sometimes it's convenient to be able to tightly control where users have access to. An easy solution for blocking user access is to place invisible colliders around the prohibited space.

You'll definitely want to block access anywhere the user may fall infinity and break the user experience as in our current museum demo.



Adding invisible collider is very simple. We'll start by adding a cube to the scene.

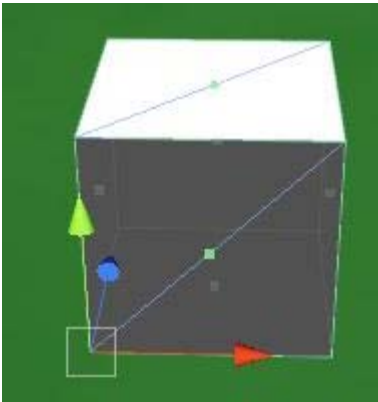
1. Go to GameObject > Create Other > Cube
2. Press 'F' to focus on the cube

A cube appears in the scene view.

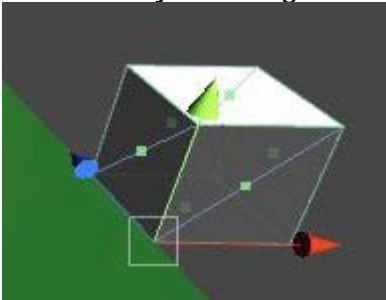
Moving the Cube 1

We'll move the cube to the edge of the green plane. We'll use vertex snapping for moving objects next to other objects.

1. With the cube still selected, holding down 'V', a square gizmo will appear over the nearest vertex.

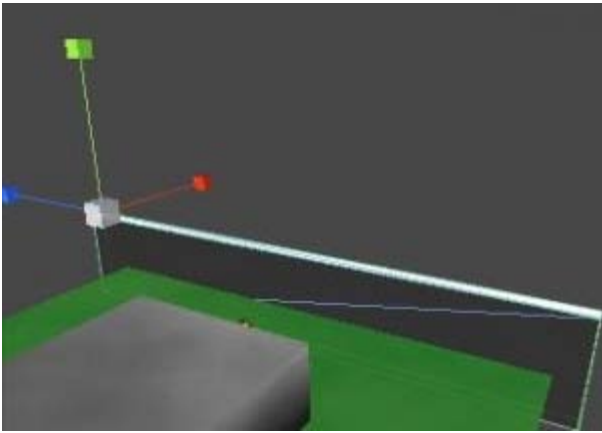


2. With 'V' still held down over the desired vertex, click LMB and the cube will snap to the nearest vertex of another object. Drag the cube so that it lines up next to the edge of the green plane.



Scaling the Cube

3. Now press 'R' to go into scale mode and scale the cube along the X axis (blue handle) along the full width of the green plane. Then scale the cube of the Y axis (green handle) so that the cube is taller than the FPC.



Hide the Cube

4. Finally, with the cube still selected, go to the Inspector and uncheck the box next to MeshRenderer.



This will render the cube invisible, but the collider will still be in effect.



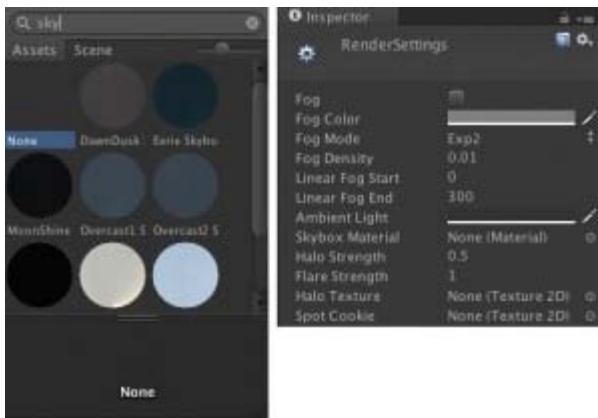
Skybox Creation

Now we can duplicate the object (Ctrl D) and reposition the duplicates adjacent to the other edges of the green plane.

Unity provides a number of skyboxes you can use. If you did not import the Skybox package when you created the project, you can import to Assets > Import Packages > Skyboxes, then click the Import button on the popup window.

Now go to Edit > Render Settings and in the inspector you'll find Skybox Material.

Click on the little circle next to it and you'll open the Material Browser. Unfortunately, it will show every Material existing in the project and will not cull out those only suitable for Skyboxes. But since Skybox materials usually have "sky" in the name, if you type "sky" in the search bar at the top, the browser will only show those materials suitable for Skyboxes. Unity provides various day and night time skyboxes.



Motion Adjustment

Most likely you will want to fine tune the motion controls and interactive experience of your walkthrough model. We'll take a look at adjusting the speed of your movement, sensitivity of the mouse and look brief at how we can modify the scripts Unity comes with.

Character Controller Menu

Select the FPC and you will see in the inspector how the FPC is constructed from individual components.

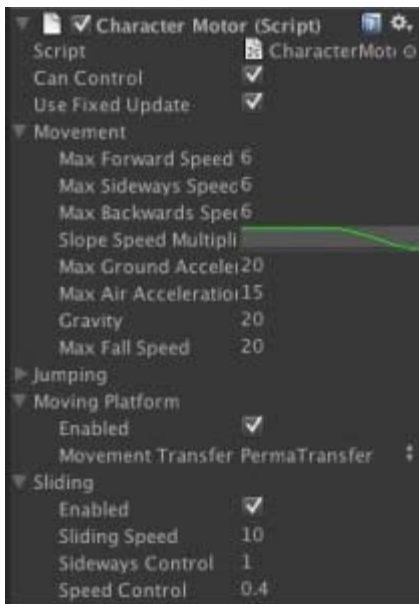
The Character Controller allows you to adjust the width and height of the FPC among other things.



The Character Motor Component

The Character Motor component allows you to adjust movement speed.

-Expand the Movement hierarchy and you'll find the Max Forward, Sideways and Backwards Speed, as well as Gravity and Ground Acceleration.

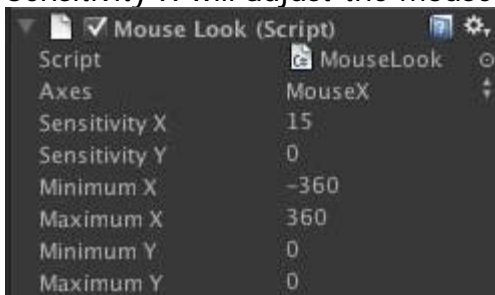


-The most effective strategy for fine tuning is to be in Play Mode and adjust the values. Anytime you adjust a value it will be reflected in the Player. Once you have values that seem appropriate, either take a snapshot of the values or write them down. When you quit Play Mode, the values will reset to their original, and you must enter them in again.

You can also adjust the Jumping or disable it completely. It is unlikely you will need to worry about Moving Platform and Sliding.

Mouse Sensitivity

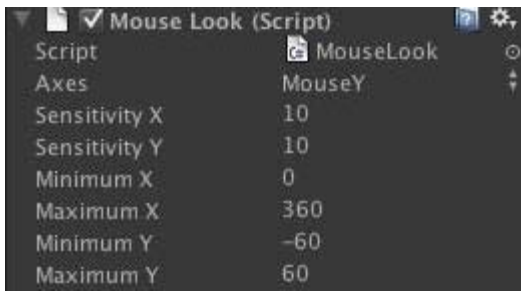
Looking around with the mouse may seem jarring or disorienting, but you can tweak these values so you can have a smooth experience. But we will have to look in two different locations. Unity handles X and Y rotation differently for the FPC. To adjust the mouse sensitivity in the X axis, with the FPC selected, look at the Inspector and find the Mouse Look component. MouseX should be selected in the Axes. Adjusting the Sensitivity X will adjust the mouse sensitivity in the X axes. Adjusting the Y will have no effect.



To modify the Y Sensitivity, select the FPC in the Hierarchy and expand it. You will find Graphics and Main Camera. Select main Camera and you the inspector will show another Mouse Look component. Adjusting the Sensitivity Y will modify your mouse movements up and down.



You can also decide how far the user can look up and down by adjusting the Minimum Y and Maximum Y values



Continue...

These are the basics for a walkable demo in Unity. We will next look at adding a mechanism for gathering information in our walkable demo.

Extra:



A museum should have artifacts! Download several Hampson artifacts, import them and place them on the pedestals inside the museum structure. You can use them in the next tutorial. Be sure to use a scale factor of .005. Leave the Generate Colliders box unchecked.

Continue [here](#)...

```
]]> http://gmvcast.uark.edu/modeling/software-visualization/unity-software-visualization/workflow-unity-software-visualization/a-walkable-unity-demo/feed/ 0 http://gmvcast.uark.edu/modeling/software-visualization/unity-software-visualization/workflow-unity-software-visualization/importing-objects-to-unity/http://gmvcast.uark.edu/modeling/software-visualization/unity-software-visualization/workflow-unity-software-visualization/importing-objects-to-unity/#comments Sat, 23 Jun 2012 10:42:22 +0000 Keenan http://gmvcast.uark.edu/?p=10254
```

This series of posts will teach you how to navigate in Unity and to create a basic virtual museum.

Hint: You can click on any image to see a larger version.

Basic Importing

Unity3D reads these file formats natively:

-FBX : an AutoDesk format designed for interoperability

-.dae : Collada format *

-. 3ds: 3D Studio Max

-.dxf : Drawing Interchange Format, a format used for interoperability between AutoCad and other programs

-.obj : Wavefront Object, an open and fairly common 3D model format

If you are working in a 3D modelling application and are able to export in one of these formats, Unity will be able to read it. Export the file into the Assets folder and the object will appear in your Project panel in Unity. For best results, always import as .fbx or .obj files. Unity works best with these particularly when it comes to scale.

FBX Converter

Autodesk supplies a free FBX converter on their website. You can find it here:

[http://usa.autodesk.com/adsk/servlet/pc/item?id=10775855&siteID=123112.](http://usa.autodesk.com/adsk/servlet/pc/item?id=10775855&siteID=123112)

Automatic Import

Unity is able to automatically import a number of 3D models from the following proprietary vendor format. The links will take you to Unity's reference manual where you can find a fuller discussion of the vendor format.

- Maya
- Cinema 4D
- 3ds Max
- Cheetah3D
- Modo
- Lightwave
- Blender

If you save the scene from any of these modeling packages to the Asset folder you are working in, Unity will automatically import them in the project. You can freely go to and fro Unity and the modeling package and all edits will be reflected in Unity.

2D Assets

Similarly, Unity reads Photoshop files natively so you can save your .psd files straight into the Assets folder. It's a good idea to create a folder named Textures to keep your files organized. Using the Photoshop documents natively allow you to make edits and see the results quickly in Unity. However, you may see a lost in quality using Photoshop Documents. To avoid this, try using 24 bit PNG files. An advisable organization strategy is to save your PSD files in a folder called PhotoshopOriginals and export the png files into the Texture folder in your Asset folder.

Continue...

Continue to read about Unity [here](#)...



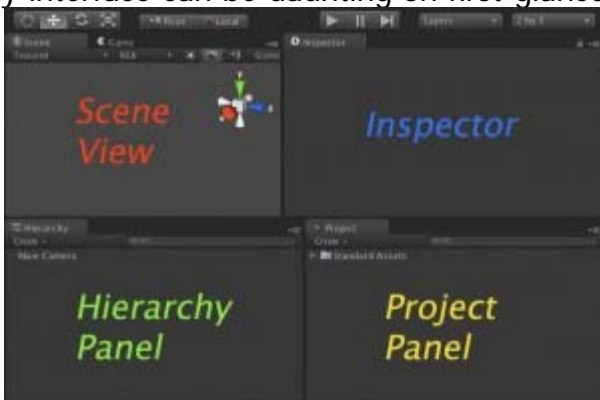
]]> <http://gmv.cast.uark.edu/modeling/software-visualization/unity-software-visualization/workflow-unity-software-visualization/importing-objects-to-unity/feed/> 0 <http://gmv.cast.uark.edu/modeling/software-visualization/unity-software-visualization/workflow-unity-software-visualization/the-unity-interface/> <http://gmv.cast.uark.edu/modeling/software-visualization/unity-software-visualization/workflow-unity-software-visualization/the-unity-interface/#comments> Sat, 23 Jun 2012 10:20:36 +0000 Keenan <http://gmv.cast.uark.edu/?p=10242>

This series of posts will teach you how to navigate in Unity and to create a basic virtual museum.

Hint: You can click on any image to see a larger version.

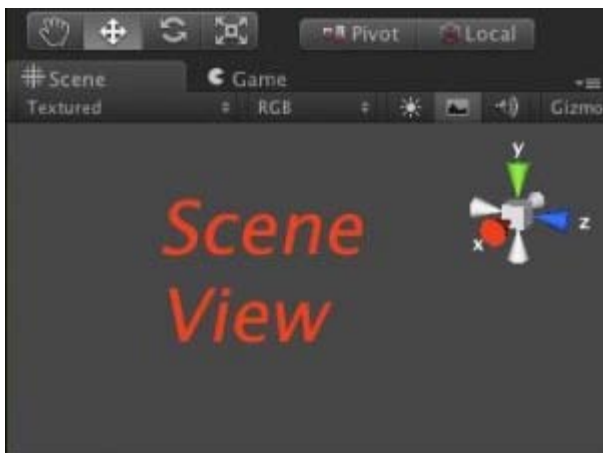
The interface

The Unity interface can be daunting on first glance. But, for getting started, we can break it down into four



sections.

Scene View



This is your 3D view, though it differs from 3D views in Blender or Cinema4D. You don't edit 3D geometry directly in this view. Rather, you use it to place things precisely where you wish them to go. If you wish to edit your geometry, you'll either need to open it up in a 3D modelling application or install additional components into Unity. You can also see the hierarchical structure of how your scene is composed in the Hierarchy panel below.

Basic Movement in the Scene View

Unity uses key shortcuts similar to Autodesk Maya, but you can change them easily by going to **Edit > Preferences**.

Look around in your scene

1. Hold Right Mouse Button (RMB) while dragging your mouse sideways will rotate the camera in place.

Rotate around in your scene

1. Holding ALT and press Left Mouse Button (LMB).

Zoom In and Out

1. Hold ALT and RMB, and then drag your mouse in or out or sideways. Alternatively, you can use the drag wheel

Centering on an object.

1. Select an object either in the Scene view or the Hierarchy view.
2. Make sure your cursor is hovering over the Scene view.
3. Press 'F' to "focus" in on your object.

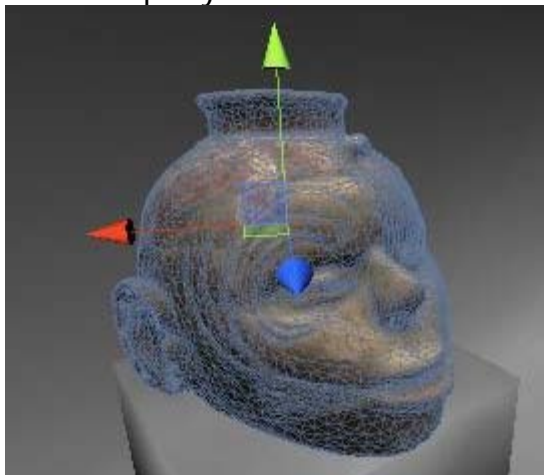
Basic Object Manipulation

You can use the controls in the top left corner to get into any mode.

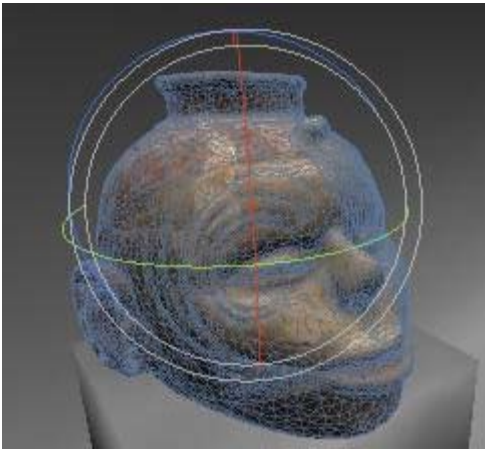


The gizmos around the selected object in the Scene will change depending on which mode you are in.

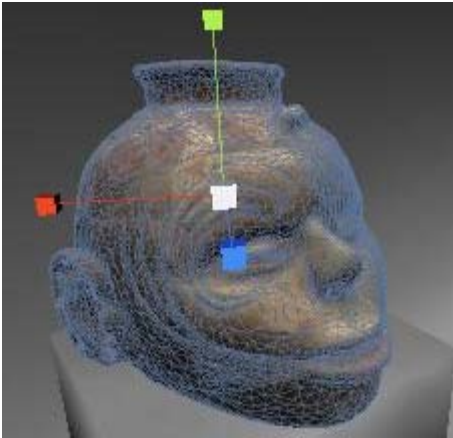
'W' : Will put you into Move model



'E' : Rotate mode



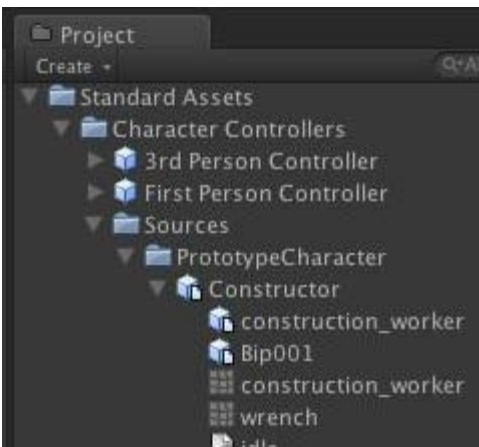
'R' : Scale mode



The Project Panel

The Project panel is your library of readily available assets you can use in your current scene. Any models or artwork you wish to place into a scene can be dragged from this panel to either the Scene view or the Hierarchy view. When you want to add assets into the Project panel, just save them to the Asset folder inside the Project folder you created at the start. If you have trouble locating where you saved your Unity project, right click on any asset in the Project panel and click on Show in Explorer.

Unity also provides a number of assets ready to go. These are located in the folder named Standard Assets. If you do not find a particular asset here, you might not have imported them when you created the project.

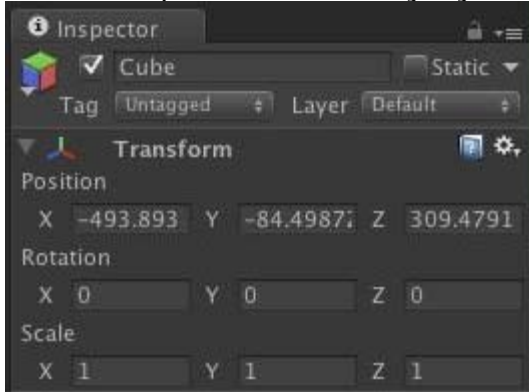


You can easily reimport them by going to Edit -> Import Packages -> DesiredPackage. We will look at the packages in more in depth in the next chapter.

The Inspector

Finally at the far right is the Inspector. This shows attributes of what you have selected. If you just have an empty game object selected, it will only show you the Transform attributes, which are its Position, Rotation and Scale. This is the base component of all GameObjects. You can position, rotate, and scale objects precisely using the coordinates in this field.

GameObjects are the heart and soul of Unity. Everything in Unity is a GameObject and you add functionality by adding Components to GameObjects much like adding layers to a Photoshop document. Common components are Character Controllers, Rigidbody, Particle Systems, and Audio Files. You can add custom functionality by creating scripts and adding them as components to various GameObjects. You can create subtle complex interaction by layering a few components to various gameobjects.



Continue...

Continue to learn about Modeling and Unity [here](#)

]]> <http://gmv.cast.uark.edu/modeling/software-visualization/unity-software-visualization/workflow-unity-software-visualization/the-unity-interface/feed/> 0 <http://gmv.cast.uark.edu/modeling/software-visualization/sketchup/workflow-sketchup/basic-unity-and-importing-a-sketchup-model-3/>
<http://gmv.cast.uark.edu/modeling/software-visualization/sketchup/workflow-sketchup/basic-unity-and-importing-a-sketchup-model-3/#comments> Mon, 11 Apr 2011 19:06:37 +0000 snowball
<http://gmv.cast.uark.edu/891/basic-unity-and-importing-a-sketchup-model/> [Continue reading →](#)]]>

Taste of Unity

Navigation:

in the scene

left click: pan right click: zoom alt + left: orbit around center

in the game

uses WASD controls

W: forward

A: left

S: backwards

D: right mouse: moves your view

Importing a SketchUp model into Unity

Unity provides users a free version of the development application for personal and commercial use. Some features are not available on the free Unity package.

1. Register, double click on the Unity Icon. Select Internet activation and follow the directions.
2. Open Unity, load Islands.unity (File > Open scene, Unity should automatically locate the file)
3. Basics of Unity: Main Editor Window



Every Unity project contains an Assets folder. The contents of this folder are presented in the Project View. This is where you store all the assets that make up your game, like scenes, scripts, 3D models, textures, audio files, and Prefabs. If you right-click on any asset in the Project View, you can choose Reveal in Finder (Reveal in Explorer on Windows) to actually see the asset itself in your file system.

Important Note: You should never move project assets around using the OS since this will break any metadata associated with the asset. Always use the Project View to organize your assets.

Scene View

This is where you will build the game, create interactions and set controls. Use the mouse and alt button to move around.

Game View

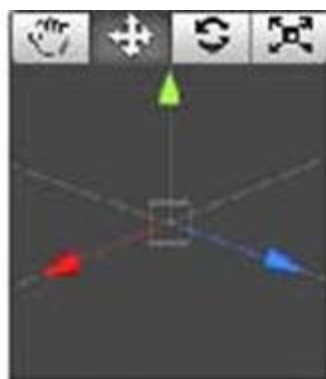
This gives you a rendered game version of your scene and lets you test the interactions and controls. To navigate around, press the play button and use the WASD keys and mouse.

1. In scene view, fly to a spot on the beach near the bridge. In the project window, select Create > Folder and create a new folder, name it SU. Now right click on the SU folder and select Show in Folder. Note where the folder is, you will save your model in SketchUp to this folder. This is the Assets folder.
2. Minimize Unity and open SketchUp.
3. Select Get Models icon, search for "beach chair" and download. Place model at the center of the scene on the axis.
4. File > Export > 3D model Change drop down list to "3DS File", select option, check that *Export two-sided faces As materials* and *Export texture maps Favor Preserving texture coordinates* are selected, click OK, name the file "beach_chair" and then Export to the SU folder created in Unity (see step #4).
5. In Unity, expand the SU folder in the Projects view. Select the beach chair and drag it to the scene, try to place it on the beach. Now translate and scale the beach chair so that it is located correctly and appropriately sized.

Transform Tool



When the hand icon is selected this will allow you to navigate around the scene. Selecting the other icons will let you translate, rotate or scale an



Translate Tool Hotkey "W"



Rotate Tool Hotkey "E"



Scale Tool Hotkey "R"

object.

1. Test the scene, select the play icon to switch to game mode. Navigate to the beach chair and have a seat.
2. Switch to SketchUp and change the chair's texture. Re-export the model to the Unity Assets folder. Now switch back to Unity, the beach chair will now show the changes made in SketchUp.

importing-a-sketchup-model-3/feed/ 0