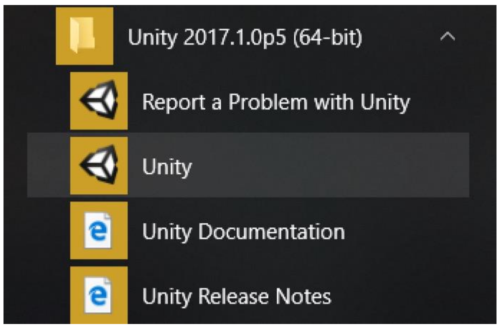
Game Design – Unity Workshop

Activity 5

Goals:

- Examine the multiple ways of implementing terrain for unity games.

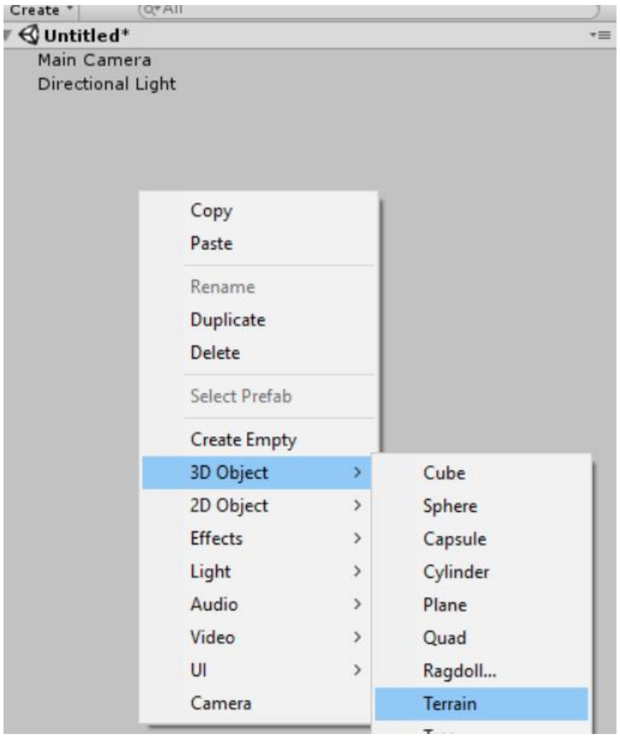
Load up unity



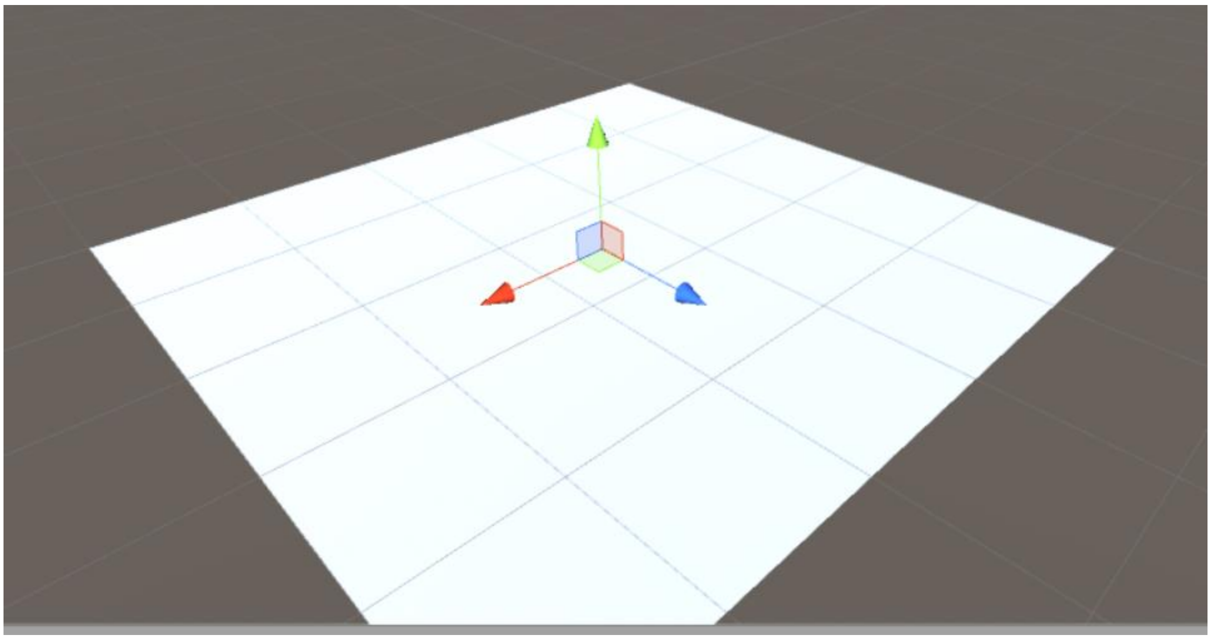
Build Object: Inbuilt tools

Aim: Use Unity's inbuilt terrain tools

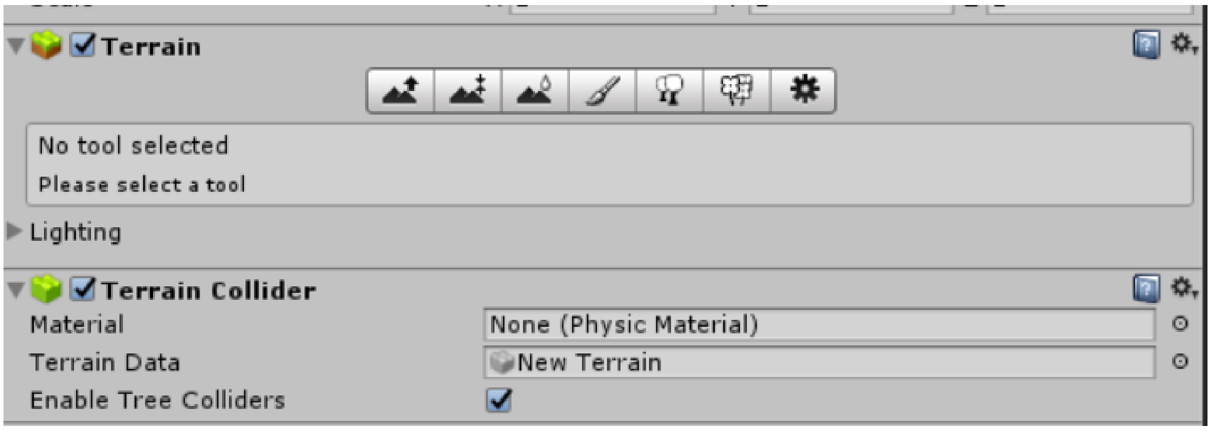
In the hierarchy, right click and create terrain.



This should provide the following in the scene view



This looks exactly like a plane, except that is you examine the inspector you will notice that there are terrain tools no provided.



Terrain

No tool selected
Please select a tool

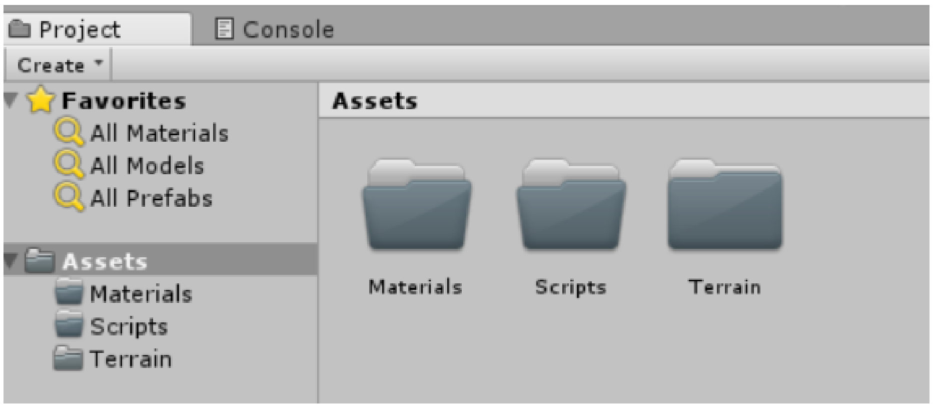
Lighting

Terrain Collider

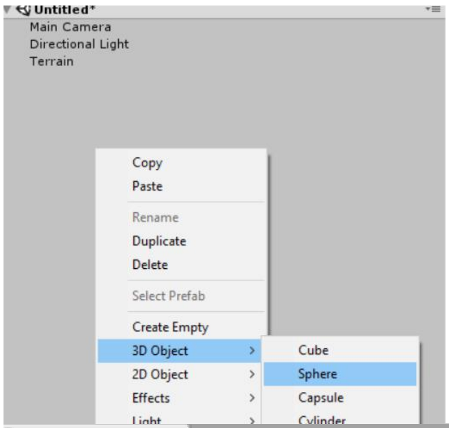
Material	None (Physic Material)
Terrain Data	New Terrain
Enable Tree Colliders	✓

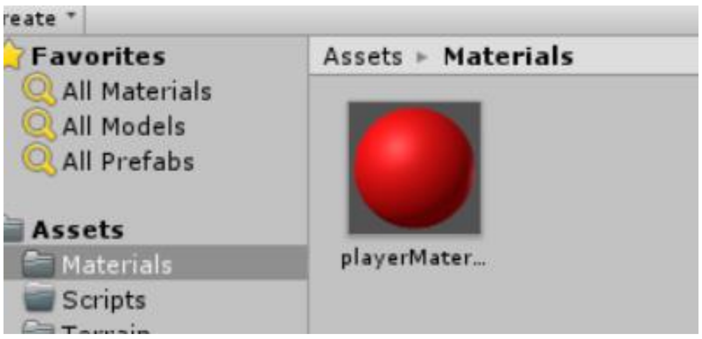
The terrain generated is huge, and the best way show that is to put a player on the terrain and view it.

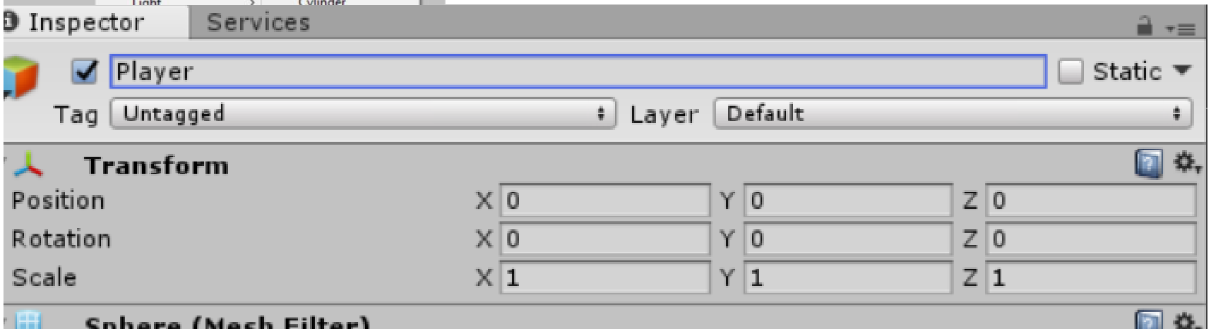
To start with create some folders in the assets panel, if you notice the creation of the terrain has a new asset in place already. Create the following folders.



Next, create a sphere, and assign a red material to it. Put the material for the sphere in the materials folder. Name the sphere player.





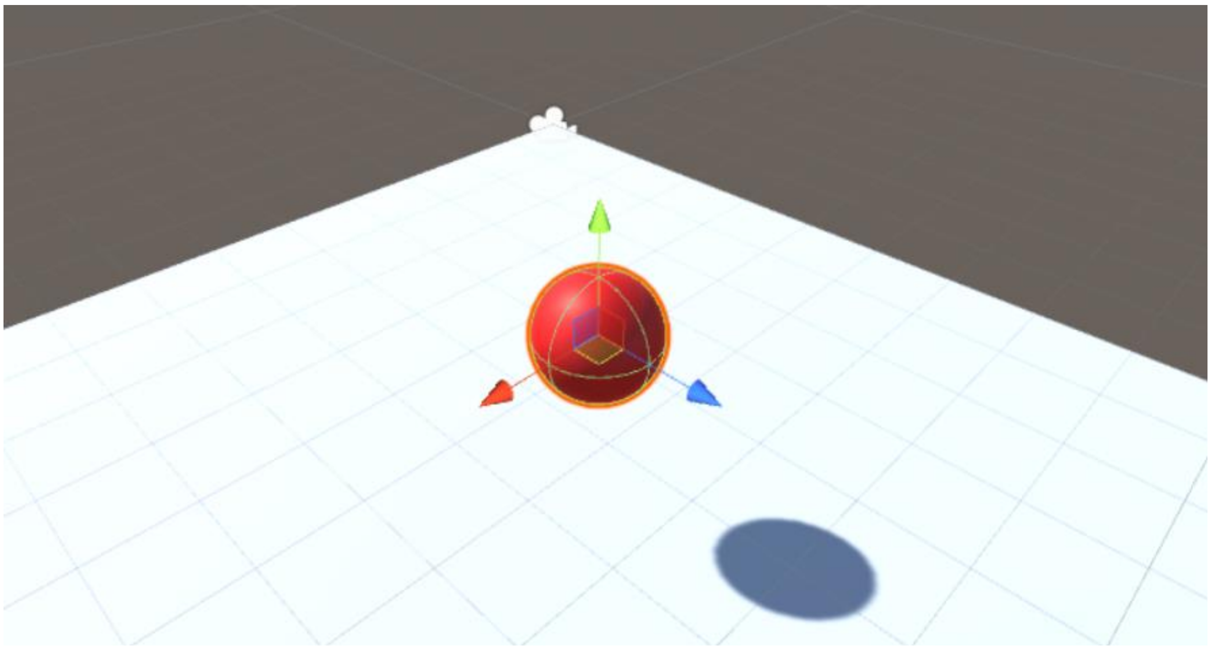


Player
Tag Untagged Layer Default

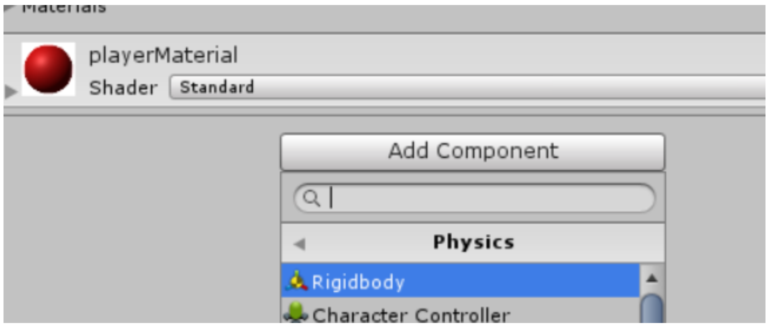
Transform

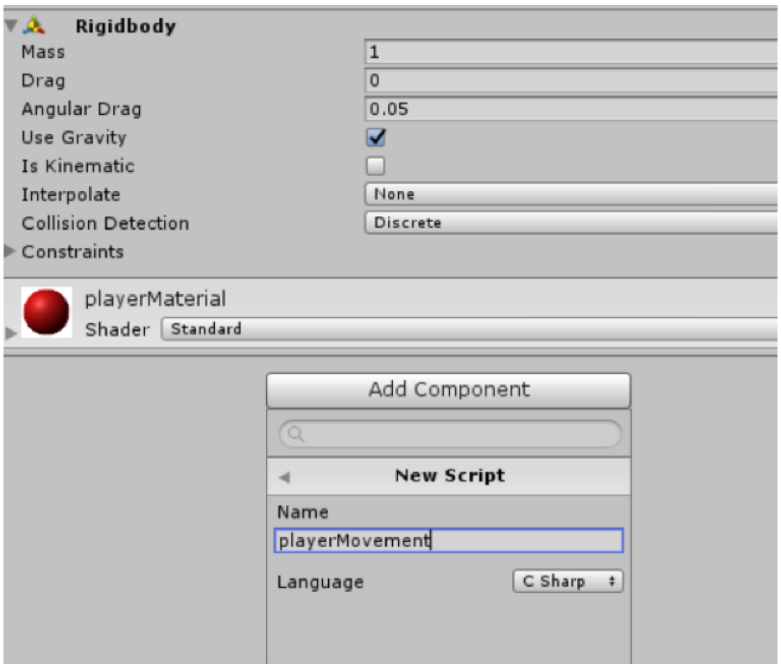
	X	Y	Z
Position	0	0	0
Rotation	0	0	0
Scale	1	1	1

Ensure that you have reset the player position. Drag the material on to the player, when you zoom in, you'll notice that the 0,0,0, point is the top corner of the terrain. Move the player object a little in and above the terrain.



From here, we can add the rigidbody and code to make the player moveable on the terrain.

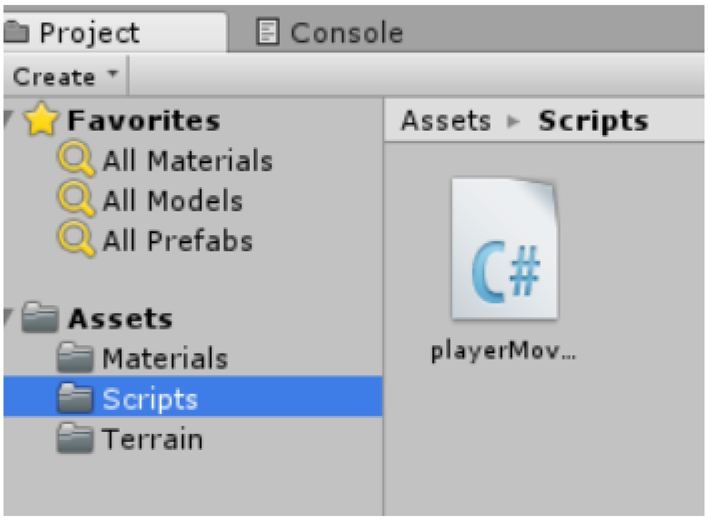


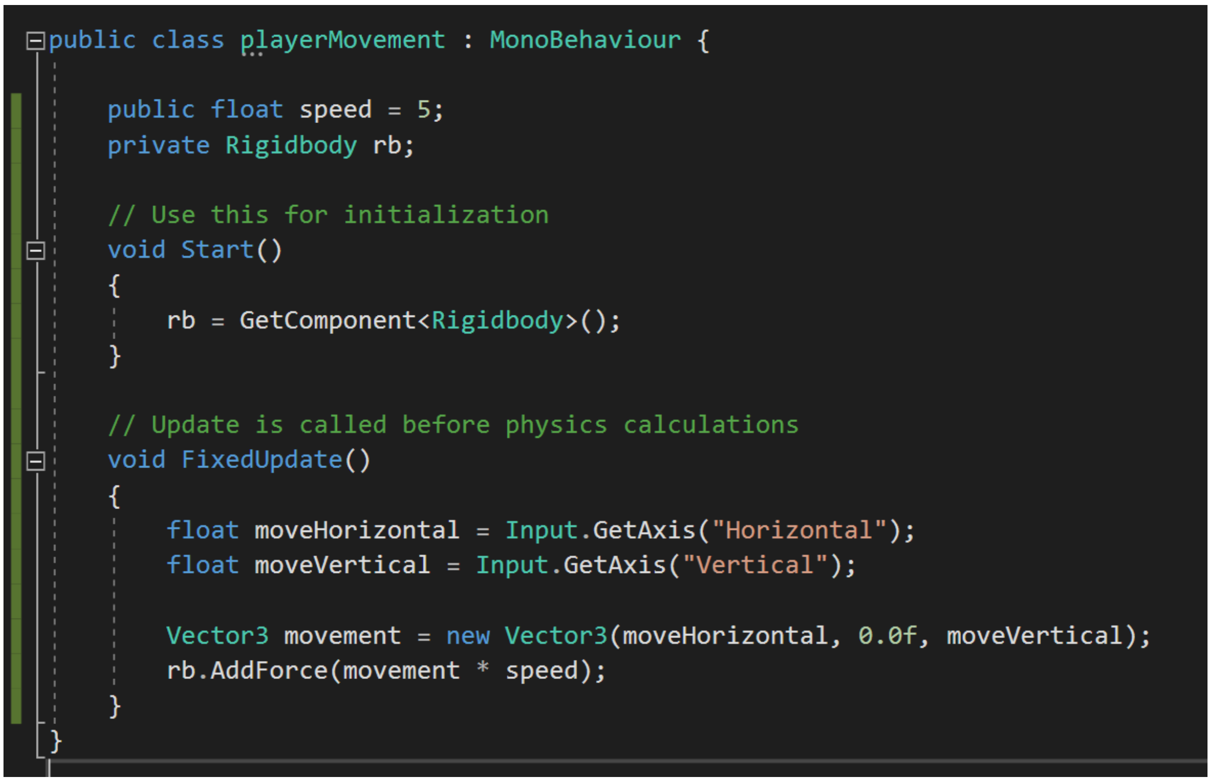


Mass	1
Drag	0
Angular Drag	0.05
Use Gravity	✓
Is Kinematic	☐
Interpolate	None
Collision Detection	Discrete
Constraints	

Name: playerMovement
Language: C Sharp

Drag the newly created script into the scripts folder and then open up the script, so we can write in the movement code

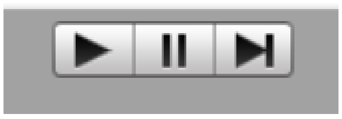


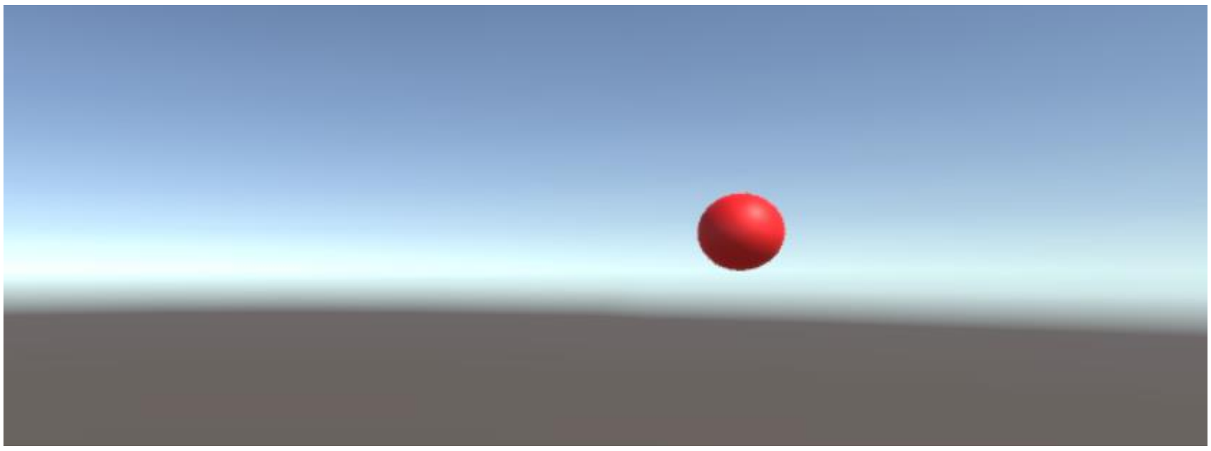


```
public class playerMovement : MonoBehaviour {

    public float speed = 5;
    private Rigidbody rb;

    // Use this for initialization
    void Start()
    {
        rb = GetComponent<Rigidbody>();
    }

    // Update is called before physics calculations
    void FixedUpdate()
    {
        float moveHorizontal = Input.GetAxis("Horizontal");
        float moveVertical = Input.GetAxis("Vertical");

        Vector3 movement = new Vector3(moveHorizontal, 0.0f, moveVertical);
        rb.AddForce(movement * speed);
    }
}
```

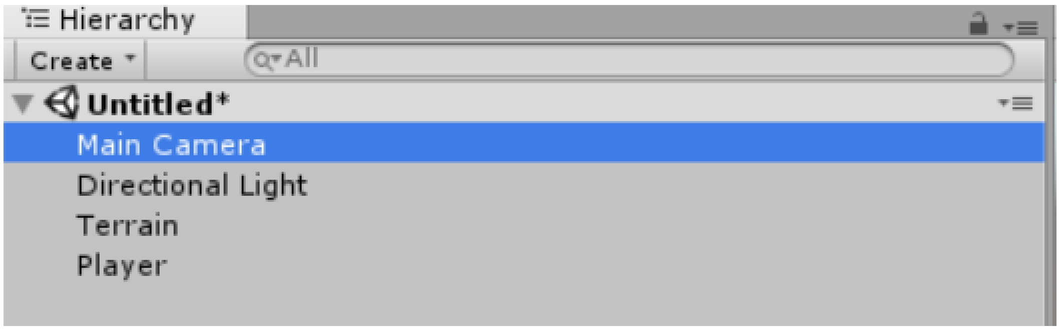
If you notice the difference between this and the other times we've coded this script, the speed is assigned 5, this way we can just have movement without having to enter it via the inspector in unity. We can still change it, if the speed isn't fast enough through the inspector, but 5 is a good starting number.

Save this code then go back to unity. From here, hit the play button and see if we can move the player using the WASD or cursor keys.



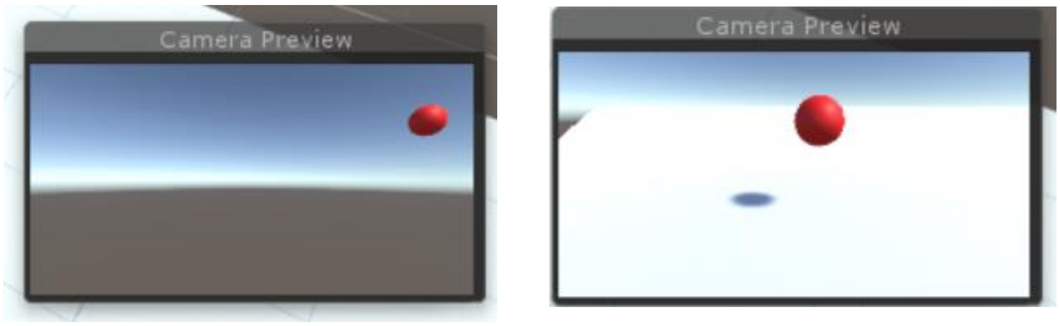


So, we can see the ball drop and can move it around, we need to add the missing element which is the camera position. Ensure you stop the game and then click on the main camera. In the hierarchy.

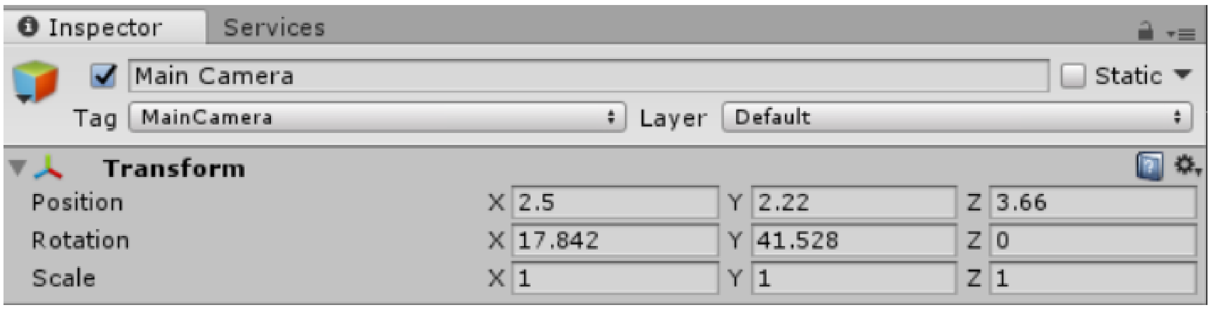


Untitled*
Main Camera
Directional Light
Terrain
Player

The initial preview shows that the camera is under the player, so we will have to move the camera to a better spot.



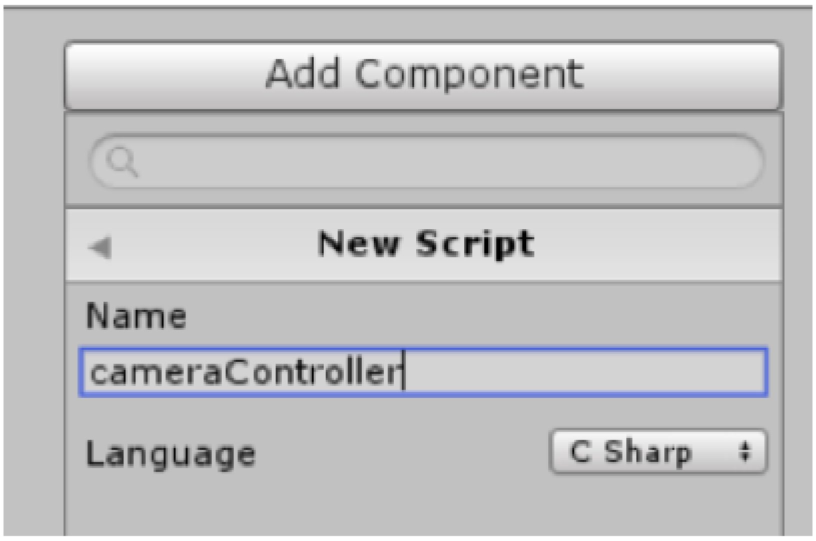
The transform positions are:



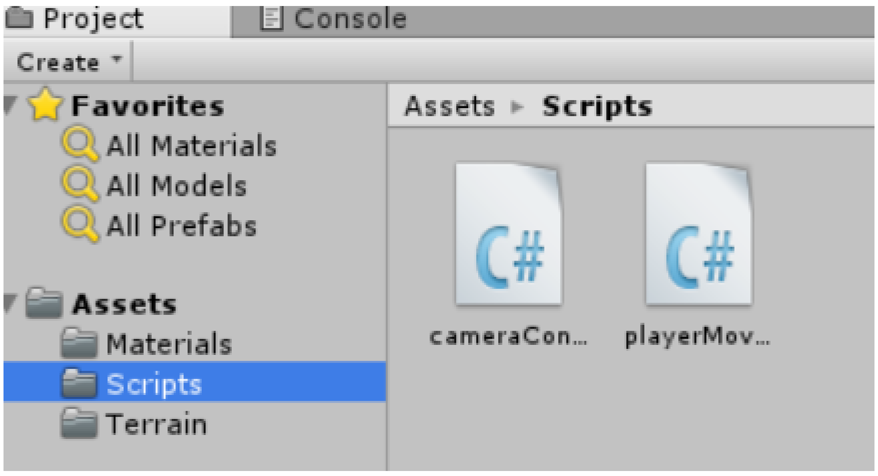
Main Camera
Tag MainCamera Layer Default

	X	Y	Z
Position	2.5	2.22	3.66
Rotation	17.842	41.528	0
Scale	1	1	1

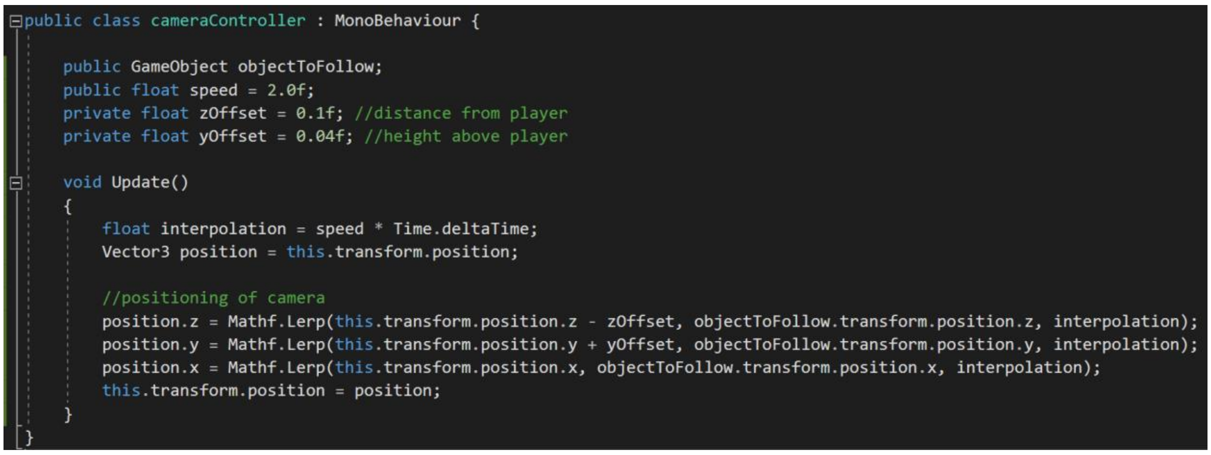
Now if you test it, the camera can watch the player move about on the terrain asset. But, we need to be able to follow the player, to do this, create a new script in the scripts folder and have it attach to the main camera.



Name
cameraController
Language C Sharp

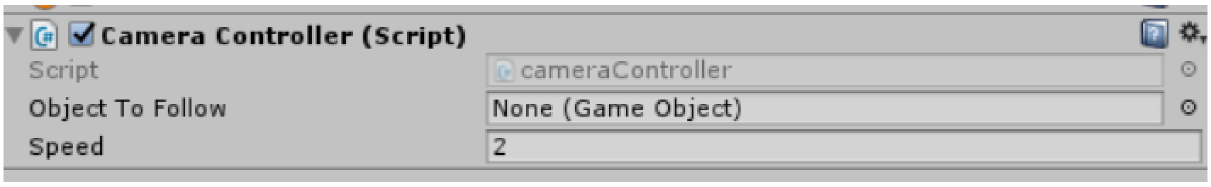


From here, open up the code to edit and add the following code.



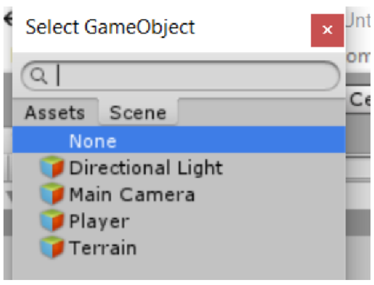
```
public class cameraController : MonoBehaviour {

    public GameObject objectToFollow;
    public float speed = 2.0f;
    private float zOffset = 0.1f; //distance from player
    private float yOffset = 0.04f; //height above player

    void Update()
    {
        float interpolation = speed * Time.deltaTime;
        Vector3 position = this.transform.position;

        //positioning of camera
        position.z = Mathf.Lerp(this.transform.position.z - zOffset, objectToFollow.transform.position.z, interpolation);
        position.y = Mathf.Lerp(this.transform.position.y + yOffset, objectToFollow.transform.position.y, interpolation);
        position.x = Mathf.Lerp(this.transform.position.x, objectToFollow.transform.position.x, interpolation);
        this.transform.position = position;
    }
}
```

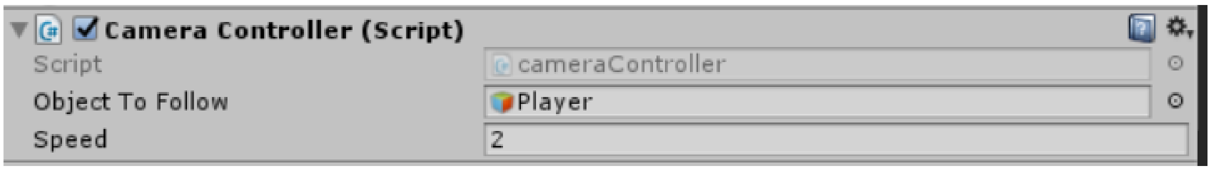
Save and go back to unity, then assign the camera to follow the Player via the target icon.



Camera Controller (Script)

Script	cameraController
Object To Follow	None (Game Object)
Speed	2





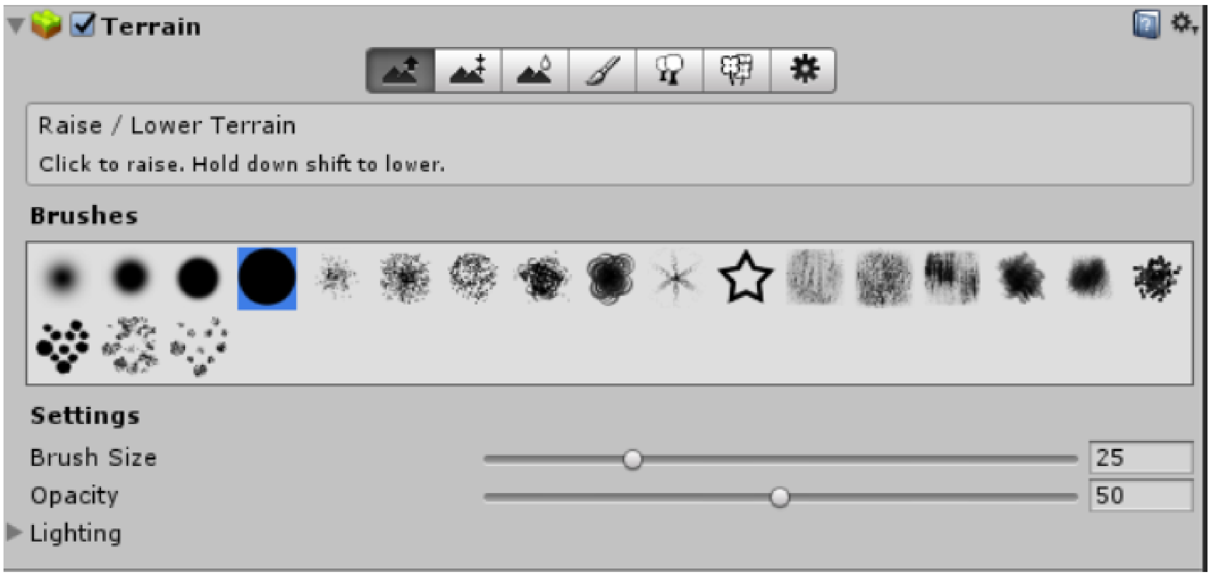
Camera Controller (Script)

Script	cameraController
Object To Follow	Player
Speed	2

Now that this has been done, run the game and make sure that the player can be followed.

As we've got movement for the player, it's time to start playing with the terrain tools. Click back on terrain in the hierarchy and look at the inspector.

If you click on the Raise/Lower terrain , you will see the following:


Terrain

Raise / Lower Terrain

Click to raise. Hold down shift to lower.

Brushes

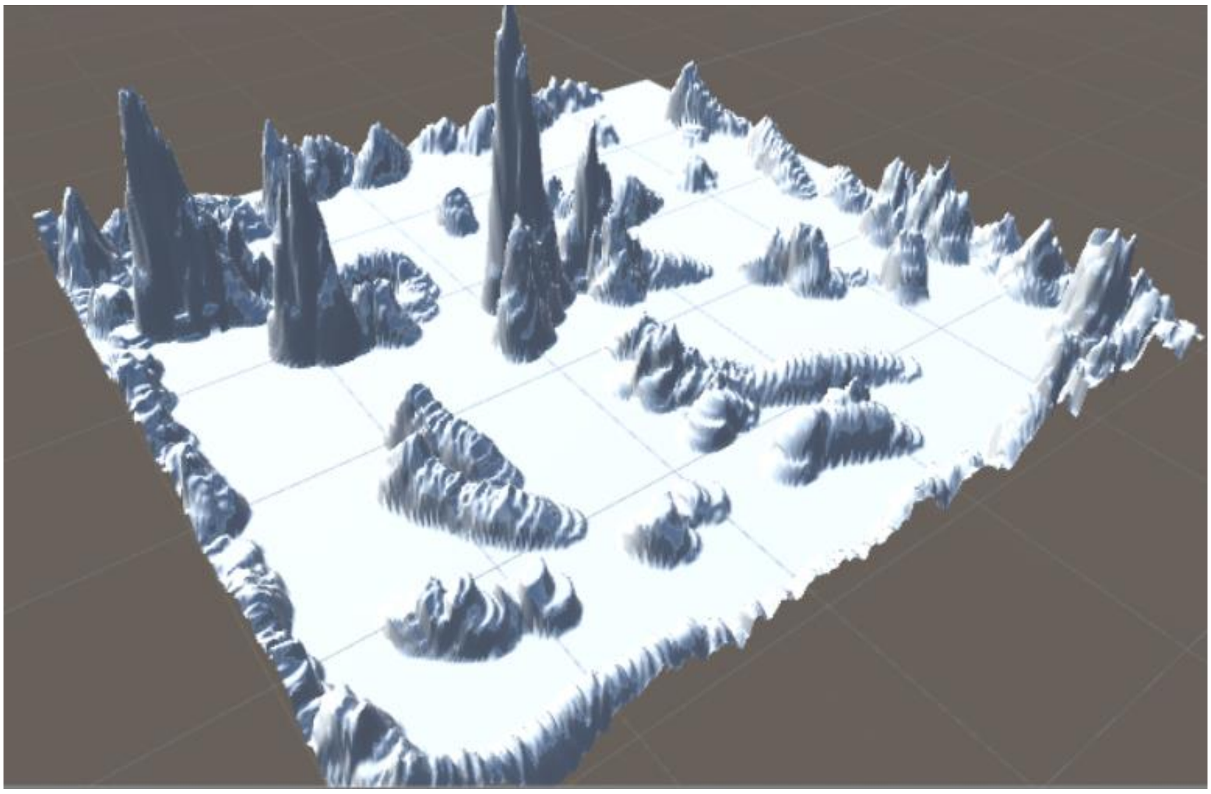
Settings

Brush Size 25

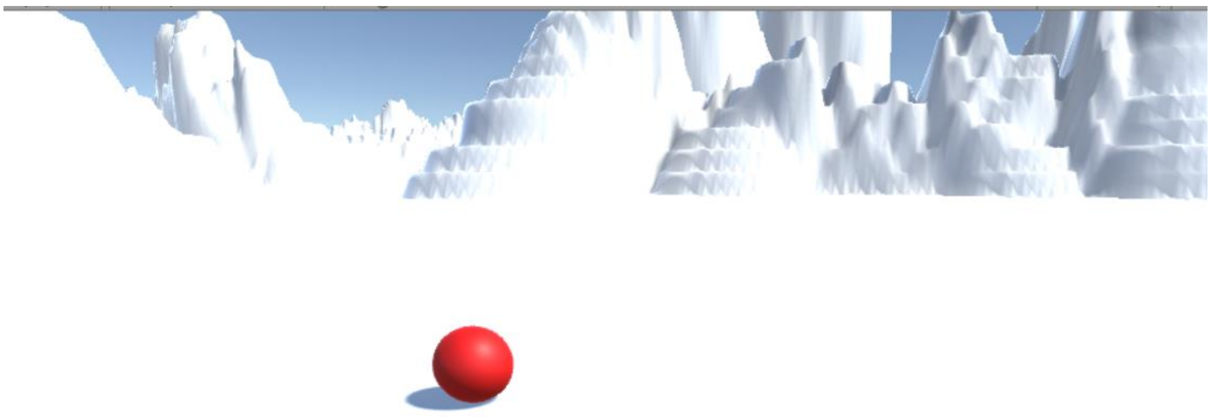
Opacity 50

Lighting

These are the differing attributes that can be applied to the brush. Once this is selected, zoom out and 'paint' some parts of the terrain.



Once you have finished making some terrain, click play and then move your player around the terrain.



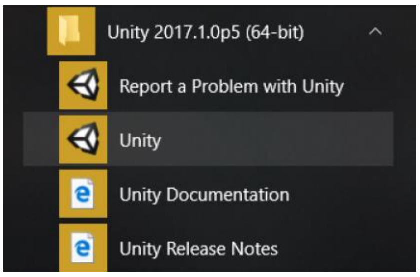
This allows you to see the environment you are making. We will look at texturing next week.

Build Object: Maya and Unity merging for a project.

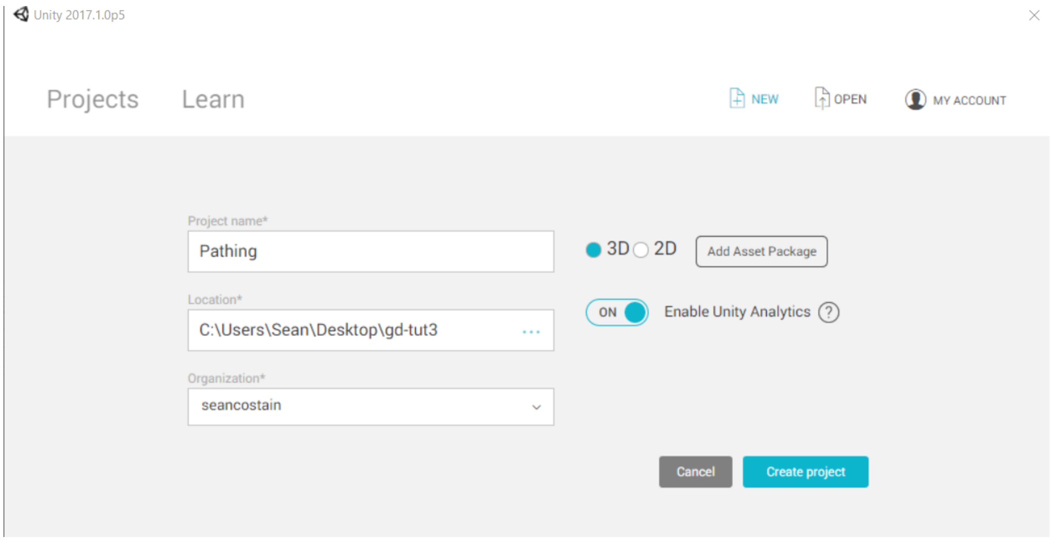
Aim: To show a simple tip to make the importing of maya models into unity.

We will start with a new Unity project:

Load up unity



Create a new project and remember the location.



Project name*: Pathing

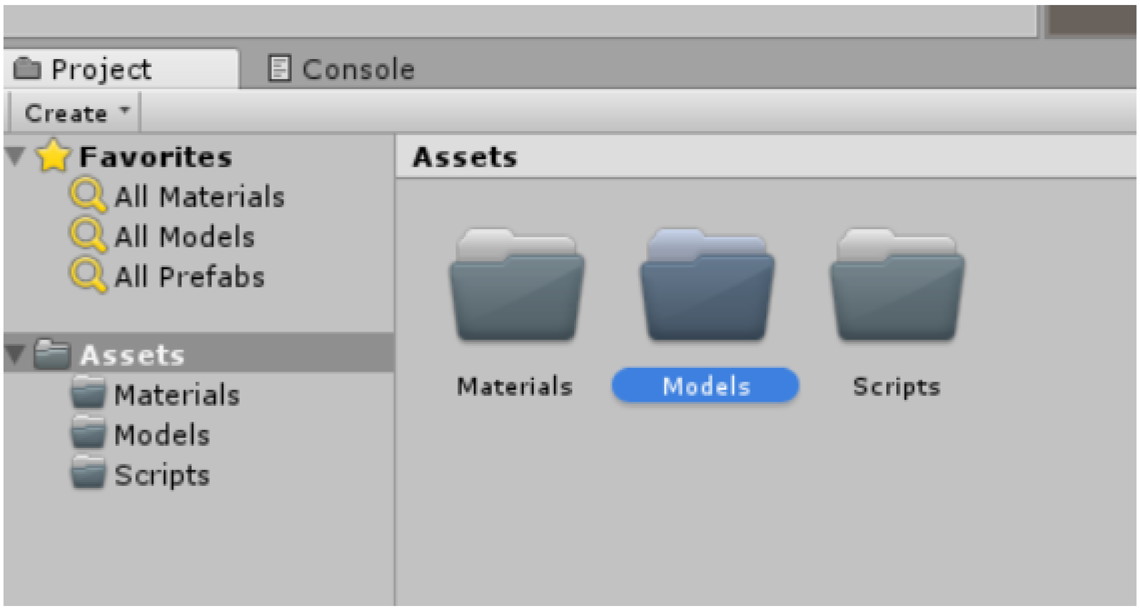
Location*: C:\Users\Sean\Desktop\gd-tut3

Organization*: seancostain

[✓ 3D] 2D

ON Enable Unity Analytics

In this case, the project will end up on my desktop in a folder called gd-tut3. Once the project has been created, go to the assets panel and create the following folder structure.



Models is the path we are going to use for placing our Maya models.

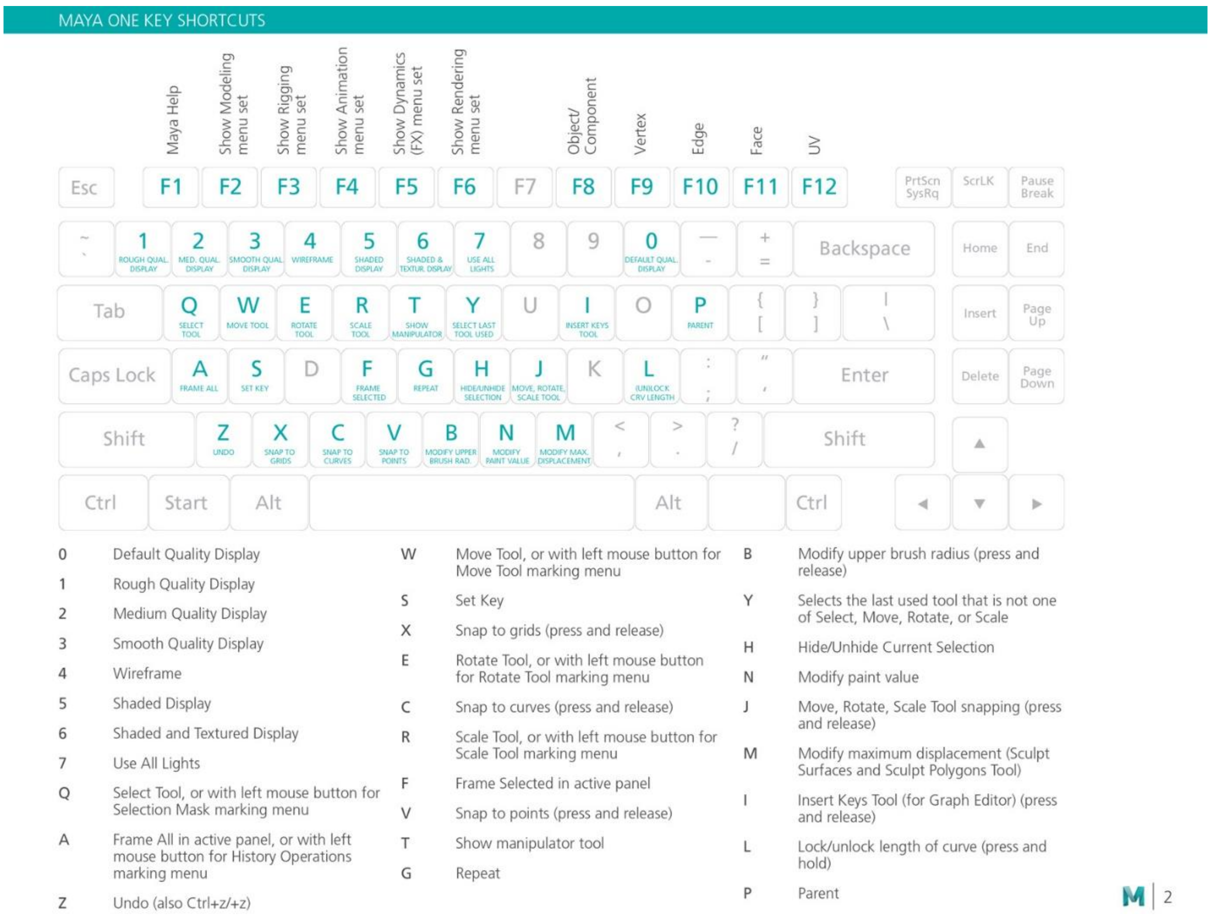
Now that this is set up, let’s jump into Maya.

Maya

Load up Maya

Keyboard Shortcuts

Manipulating elements in Maya is a combination of Keyboard and Mouse. Below are the primary keyboard keys:



0	Default Quality Display
1	Rough Quality Display
2	Medium Quality Display
3	Smooth Quality Display
4	Wireframe
5	Shaded Display
6	Shaded and Textured Display
7	Use All Lights
Q	Select Tool, or with left mouse button for Selection Mask marking menu
A	Frame All in active panel, or with left mouse button for History Operations marking menu
Z	Undo (also Ctrl+z/+z)
W	Move Tool, or with left mouse button for Move Tool marking menu
S	Set Key
X	Snap to grids (press and release)
E	Rotate Tool, or with left mouse button for Rotate Tool marking menu
C	Snap to curves (press and release)
R	Scale Tool, or with left mouse button for Scale Tool marking menu
F	Frame Selected in active panel
V	Snap to points (press and release)
T	Show manipulator tool
G	Repeat
B	Modify upper brush radius (press and release)
Y	Selects the last used tool that is not one of Select, Move, Rotate, or Scale
H	Hide/Unhide Current Selection
N	Modify paint value
J	Move, Rotate, Scale Tool snapping (press and release)
M	Modify maximum displacement (Sculpt Surfaces and Sculpt Polygons Tool)
I	Insert Keys Tool (for Graph Editor) (press and release)
L	Lock/unlock length of curve (press and hold)
P	Parent

The main keys used are:

Q – Select

W – Move

E – Rotate

R – Scale

Spacebar – Change view (4 panels to single view panel)

In addition to one key shortcuts there are combination keys which use Ctrl or Shift plus a key to achieve differing results; primary combinations are:

Edit Operations

Ctrl (or Cmd) + C	Copy

Ctrl (or Cmd) + X	Cut
Ctrl + D	Duplicate
Ctrl + Shift + D	Duplicate Special
Shift + D	Duplicate with Transform
Ctrl + G	Group
P	Parent
Ctrl (or Cmd) + V	Paste
Shift + Z	Redo
G	Repeat
Shift + G	Repeat command at mouse position
Z	Undo (also Ctrl+z/+z)
Shift + P	Unparent
Ctrl + R	Create file reference
Ctrl + Q	Exit
Ctrl + N	New Scene
Ctrl + O	Open Scene
Ctrl + S	Save Scene
Ctrl + Shift + S	Save Scene As

All key combinations can be located here: https://www.autodesk.com/shortcuts/maya#

Mouse Manipulation

Mouse manipulation is enhanced due to a combination of key presses at the same time as mouse click events.

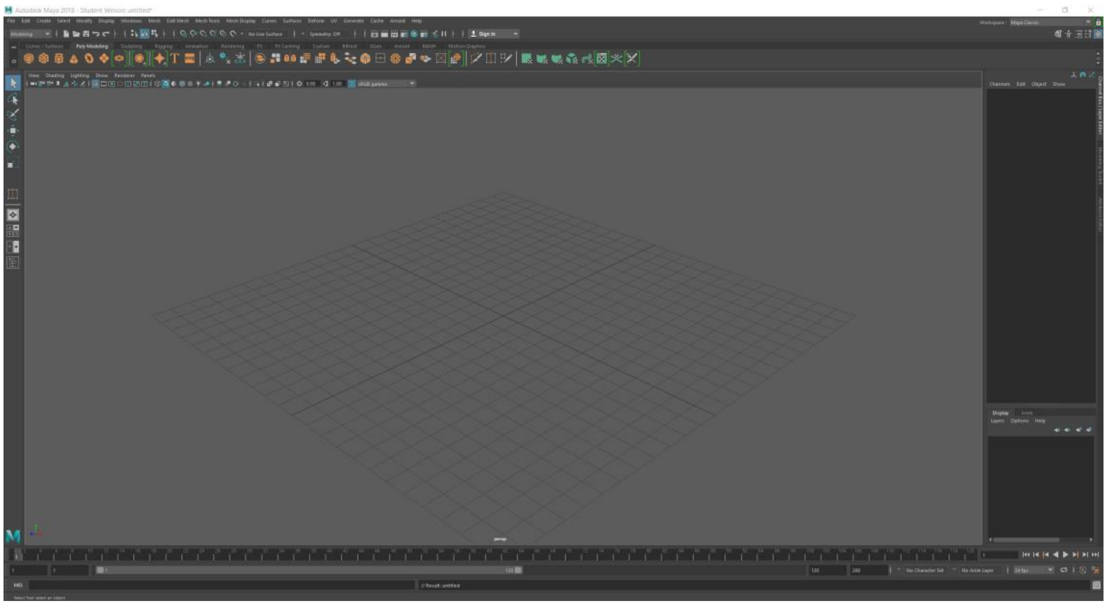
- Left Click: Object Select
- Left Click + Alt: Rotate around object

- Right Click: Menu pop up
- Right Click + Alt: Zoom in/Out

- Scroll Wheel: Zoom in/Out
- Scroll Wheel + Alt: Pan screen

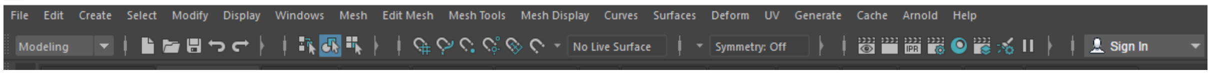
Interface

The Maya interface looks like the following:

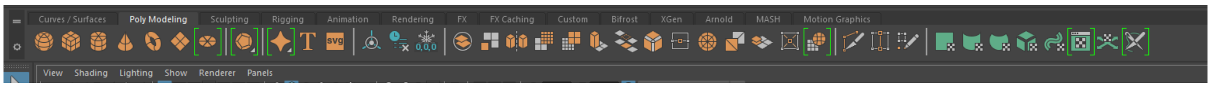


There is a lot going on, but we can simplify the interface down to areas to ease up the learning curve. Primary areas of concern are:

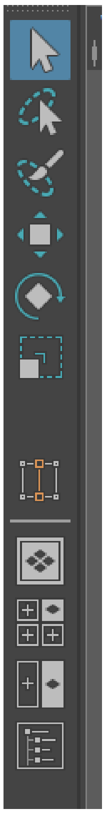
Menu:



Shelf:



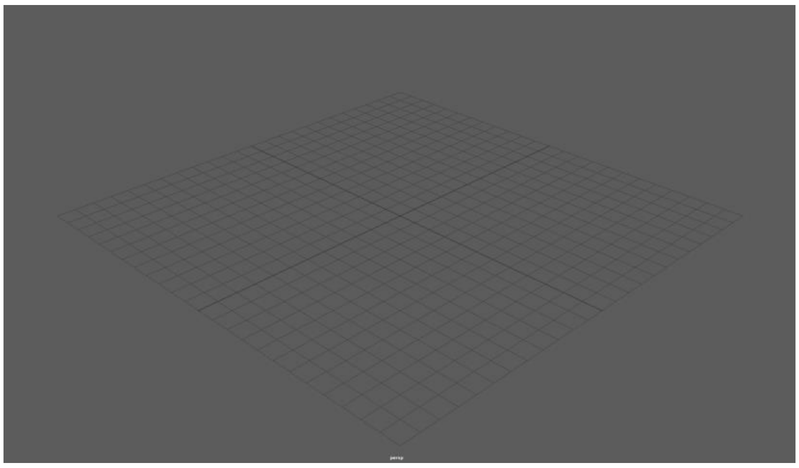
Tool Box:



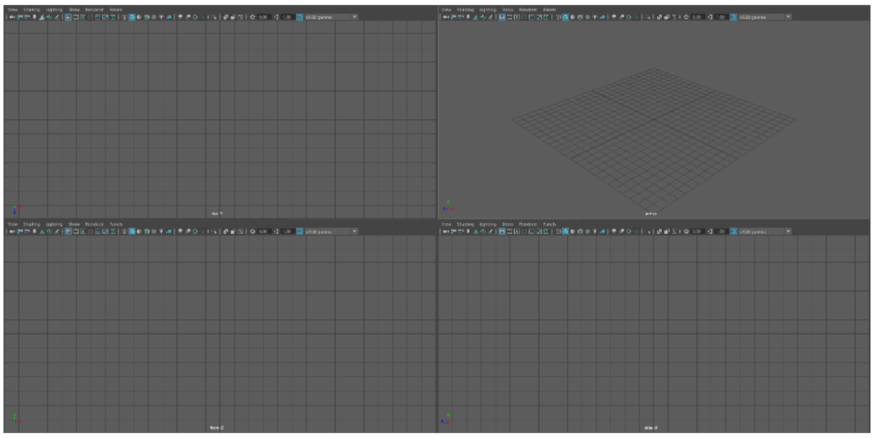
Channel Box:

Channels
Edit
Object
Show
Display
Anim
Layers
Options
Help

View Panel:



or

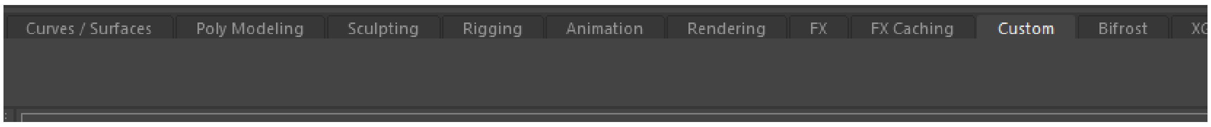


As you can see, the view panel comes in two base layouts, a single view or 4 views; front, top, side and perspective views.

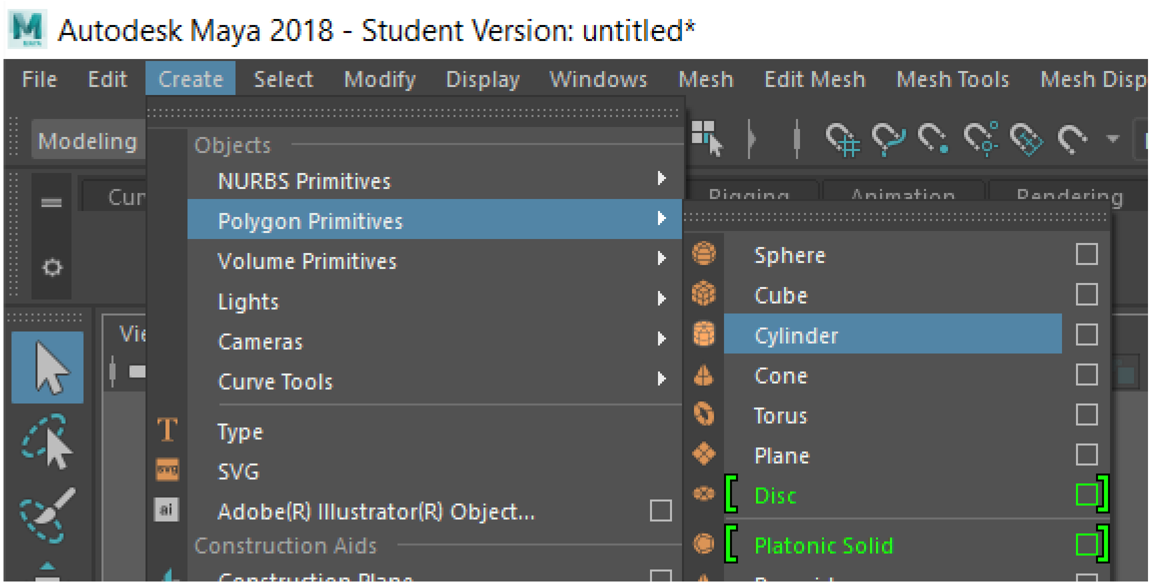
The shelf contains the primary objects that get placed and manipulated in the view panels. It's possible to create a custom shelf, this allows for the focus on the primary objects and tools you will use. Basically, elements are selected from the menu system and added to the shelf. This id done by using Ctrl+Shift+ left click on the icon from the menu whilst the custom shelf is displayed.

Creating a Custom shelf

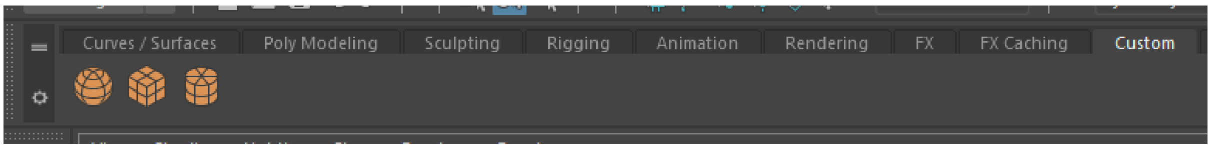
Click on custom in the shelf area



Next we add some objects, in this case we will add the following polygons Sphere, Cube and Cylinder. From the Menu->Create->Polygon Primitives section run the mouse over the icon next to the words Sphere, Cube and Cylinder.



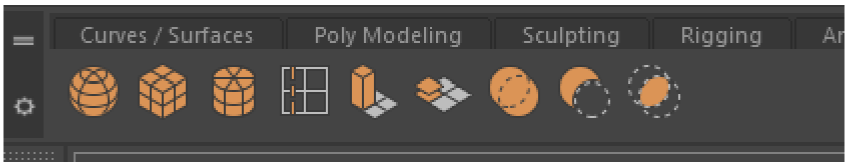
Hold down Ctrl+Shift keys and then click on the icons, this should change the blank custom shelf to contain the following:



In addition to adding objects to the shelf, we can also include tools. Add the following tools to the custom shelf:

- Mesh Tools-> Insert Edge Loop
- Edit Mesh -> Extrude
- Edit Mesh->Duplicate
- Mesh->Boolean->Union
- Mesh->Boolean-> Difference
- Mesh->Boolean-> Intersection

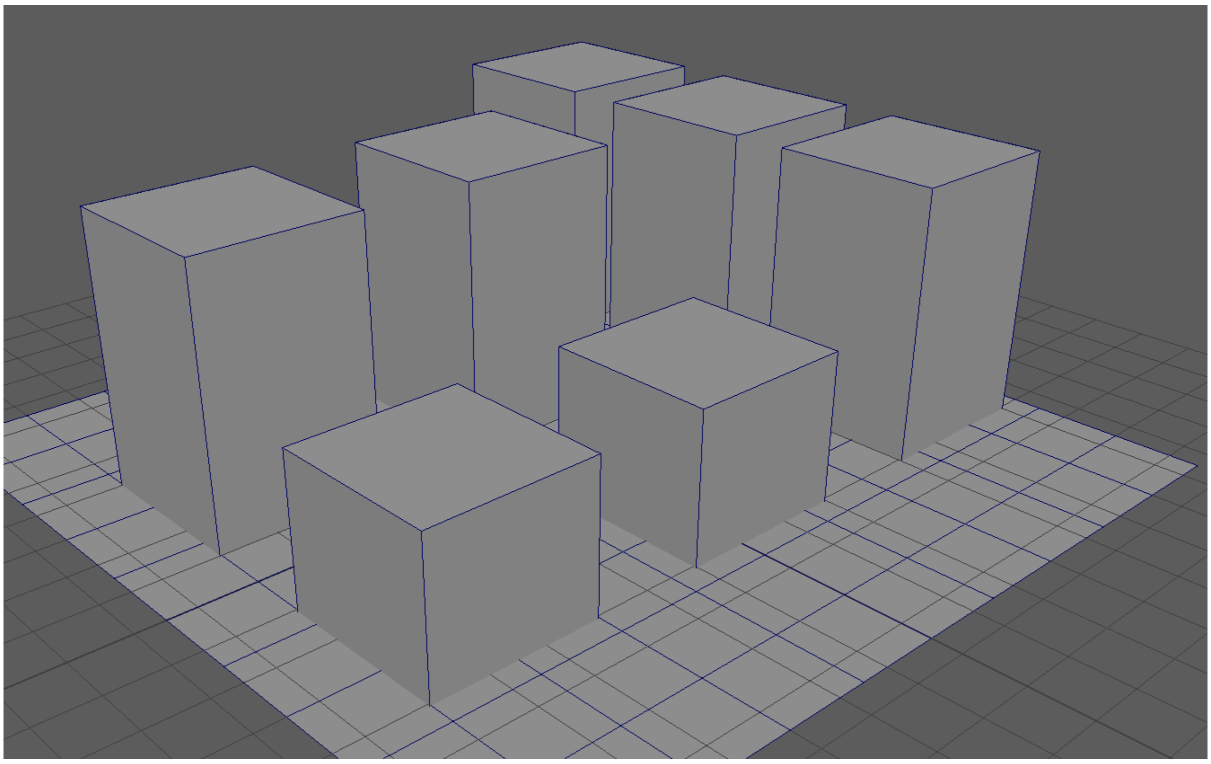
This should give you the following Custom Shelf:



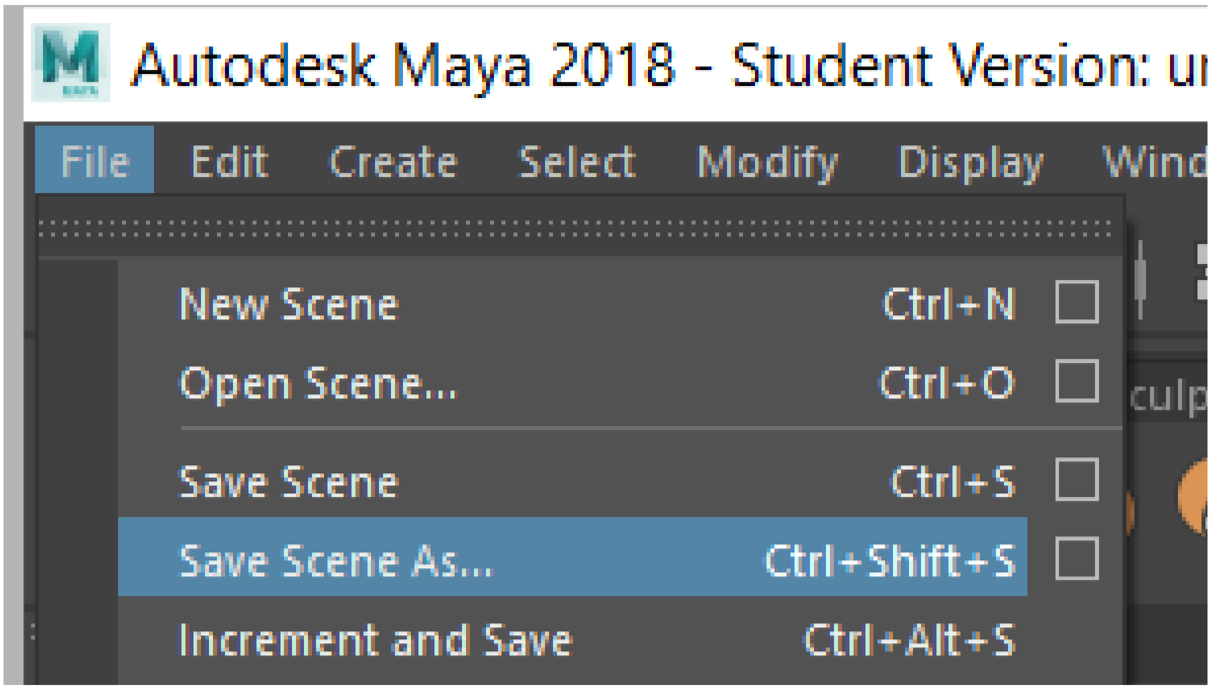
Now that we have a custom shelf, let’s build an object.

City Object

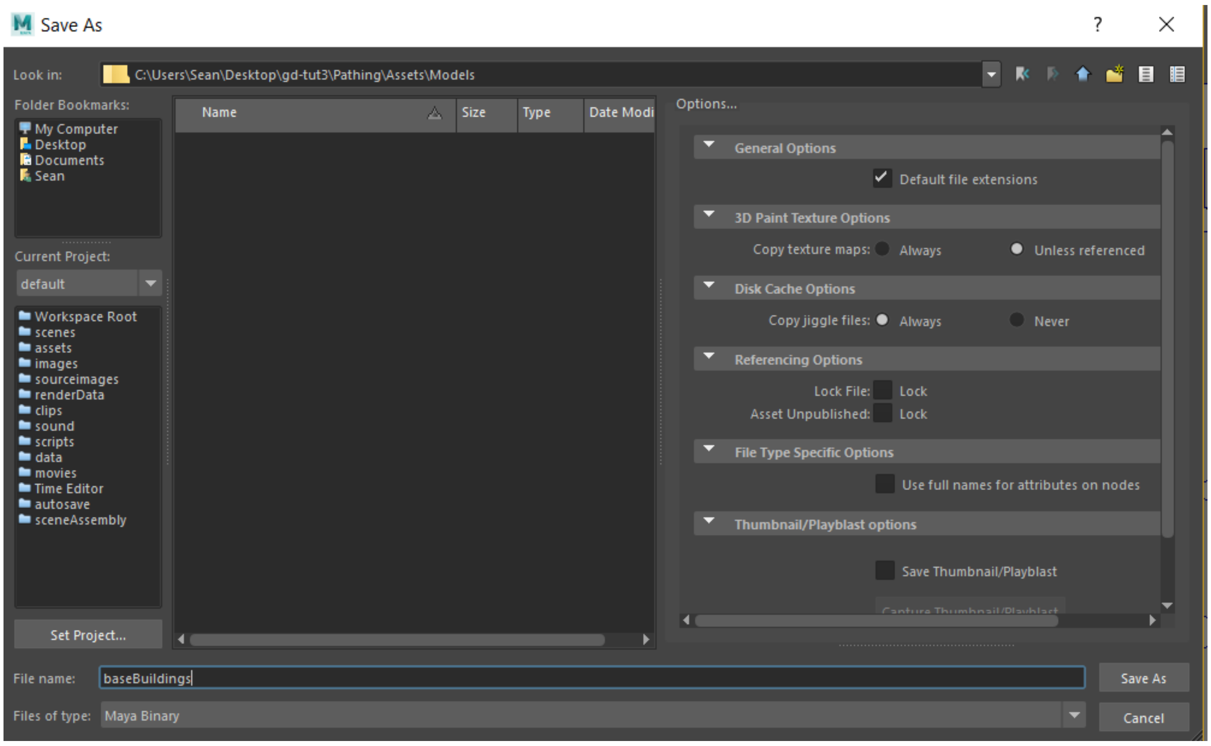
From here, quickly create a simple block layout for a rough city.



Now, the trick to making this work is the saving of this project. From here, go file -> save scene as and then locate the folder in which the unity project is placed.



Autodesk Maya 2018 - Student Version: u
File
Edit
Create
Select
Modify
Display
Wind
New Scene Ctrl+N
Open Scene... Ctrl+O
Save Scene Ctrl+S
Save Scene As... Ctrl+Shift+S
Increment and Save Ctrl+Alt+S



General Options: Default file extensions [✓]

3D Paint Texture Options: Copy texture maps: Always [✓ Unless referenced]

Disk Cache Options: Copy jiggle files: [✓ Always] Never

Referencing Options: Lock File: Lock; Asset Unpublished: Lock

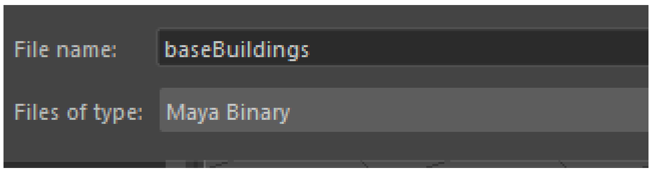
File Type Specific Options: Use full names for attributes on nodes

Thumbnail/Playblast options: Save Thumbnail/Playblast

File name: baseBuildings

Files of type: Maya Binary

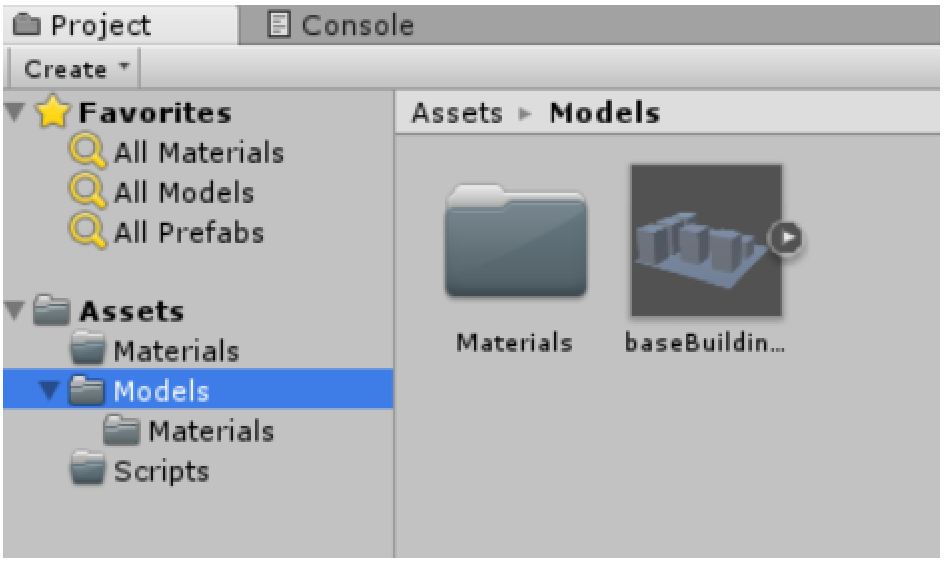
The critical part is to ensure that the file is stored as a Maya Binary:



File name: baseBuildings

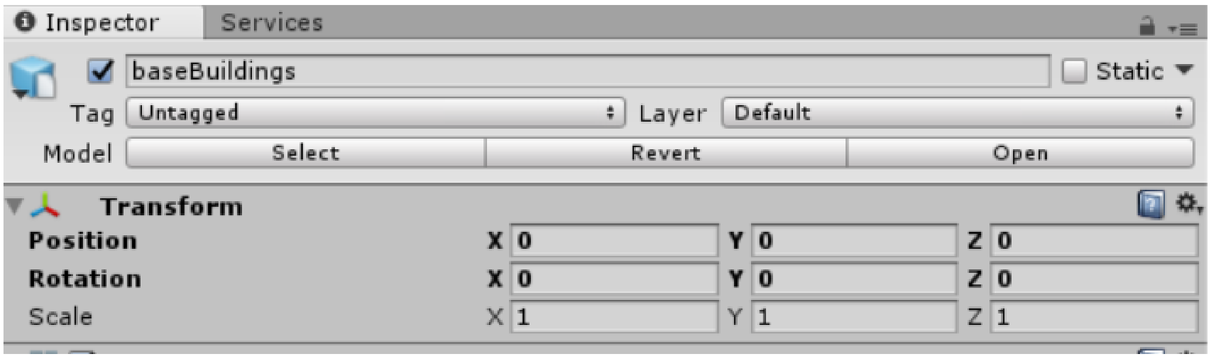
Files of type: Maya Binary

Once the file has saved, go back into unity. There will be a little bit of time as unity re=populates the assets folder, but once done you should see the following:



Notice that there is a materials folder that came with the model. This allows for the model and elements to be transferred into unity as a whole. This will avoid missing textures and so forth. From here, you can drag the model into the scene view.

You will probably have to reset the position, but you should end up with something like this:



Inspector | Services

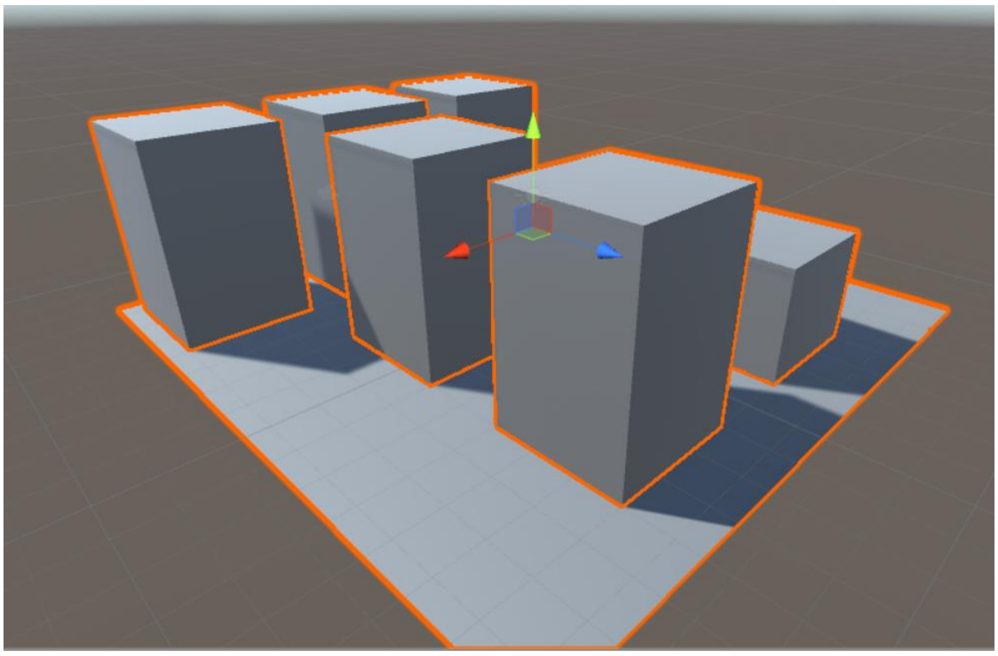
☑ baseBuildings ☐ Static

Tag: Untagged | Layer: Default

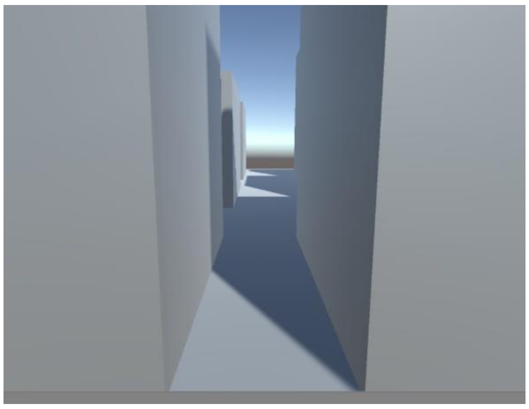
Model: Select | Revert | Open

Transform

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

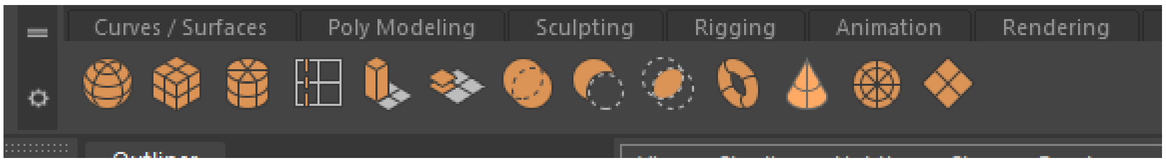


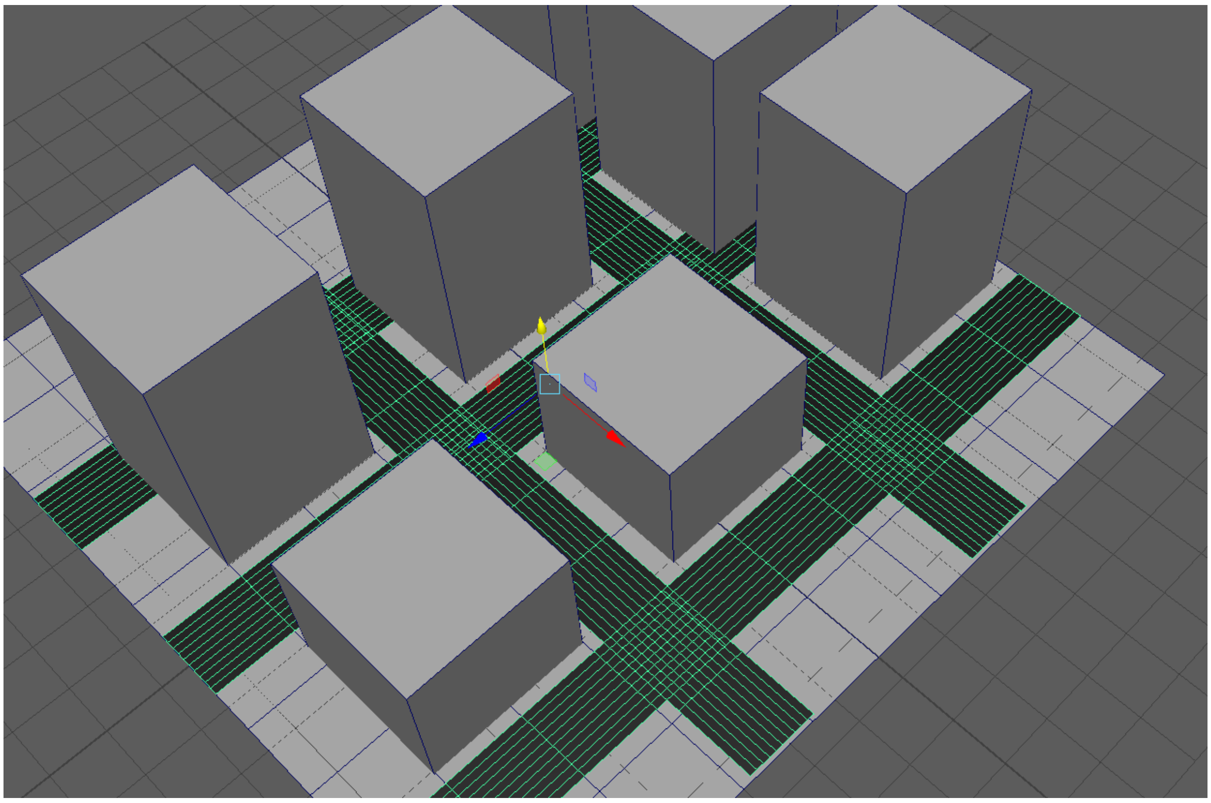
From here the terrain can be used like a normal object, such as sphere or cube. When the game runs, depending on how the camera is position you should see the following:



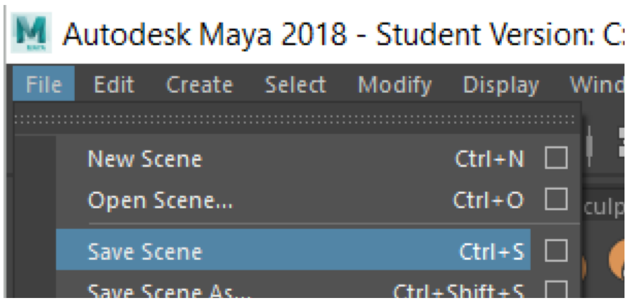
This is a good start, but everything looks the same, so let's jump back into Maya and add a quick and simple road between the buildings.

In maya, to add a road, add in new plane and assign a black material too it.

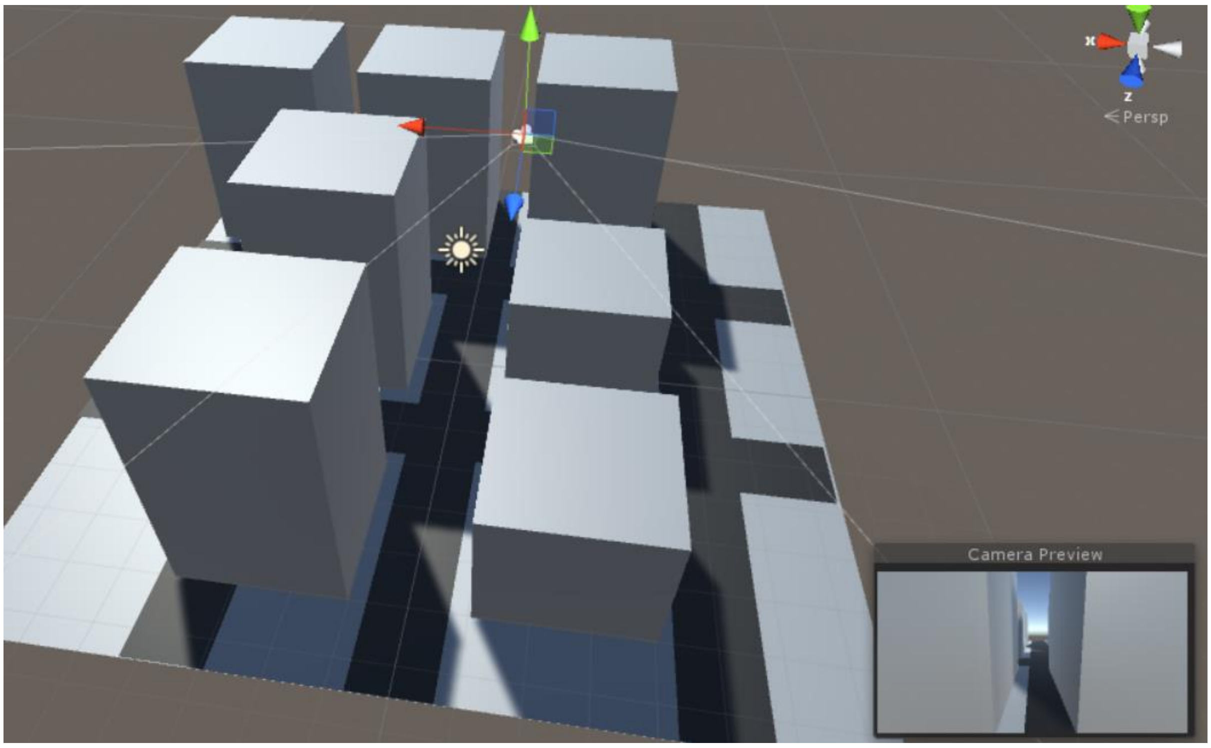




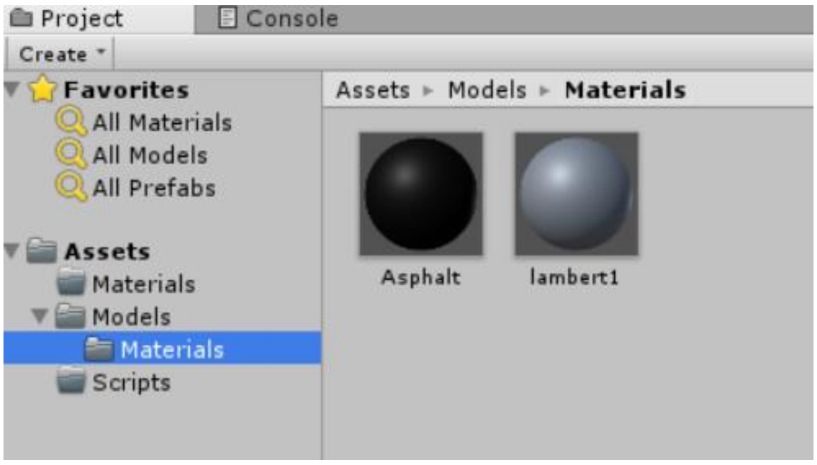
Once done, save the project and jump back to unity.



Unity will automatically update the model, so you should see something similar too



If you examine the assets folder you should see the following.


Project
Console
Create
Favorites
All Materials
All Models
All Prefabs
Assets
Materials
Models
Materials
Scripts
Assets ▸ Models ▸ Materials
Asphalt
lambert1