

# Creating Simple Environment

## Objectives

- Creating a simple scene using primitive 3D objects.
- Understanding Materials and add Texture to them.
- Using Prefabs.
- Parenting GameObjects.
- Introduction to Characters Standard Asset.
- Using Asset Store to import Asset Packages.
- Introduction to terrain.

## Using Primitive Object to Create our environment

We want to create the following seen

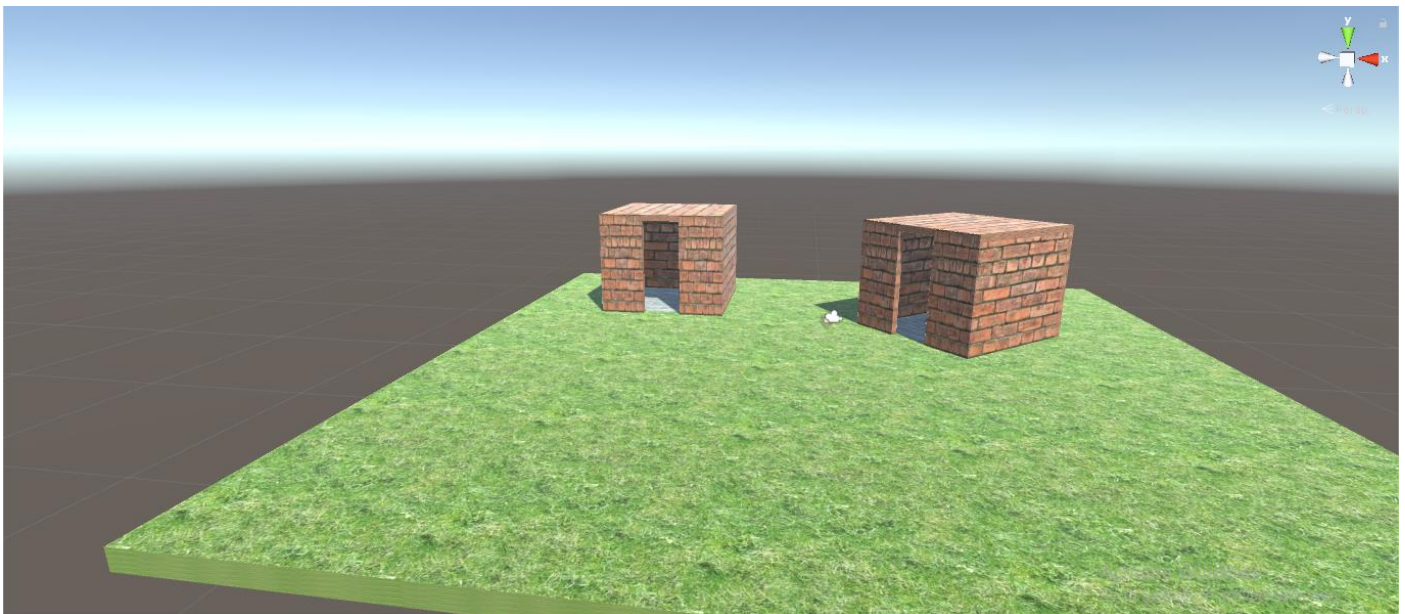


Figure 1 Simple scene that contains Texture, material. And using of a primitive 3d structures to build houses

- Create a new Project.
- Add a *Cube*.
- Change its dimensions to 50, 1 and 50  
Note, dimensions mean the *Scale x, y* and *z* values in the *Transform* component.

- Now Create to new Folders in the *Assets* section of the *Project* view. Call them *Materials* and *Textures*.

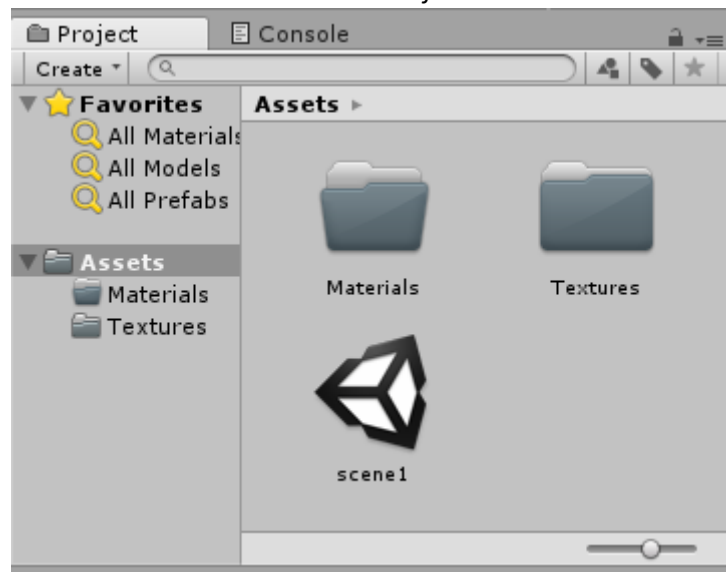



Figure 2 The Project View After Adding 2 folders

- Open the *Textures* folder and drag the “Grass001.jpg” image to it.  
Or you can import it as follow **Right Click in the Assets section** > Select **Import New Asset** a new window will pop up to enable you to select the Asset you want.
- Add a new *Material* in the *Materials* folder, name it “Grass001” and select it.
- After selecting the material, go to the *Inspector* and click on the empty circle before the *Albedo* option  **Albedo**.
- A *Select Texture* window will pop up, select the “Grass001” from it.
- Now drag the material to the Cube we created earlier. The result should look like the following image.

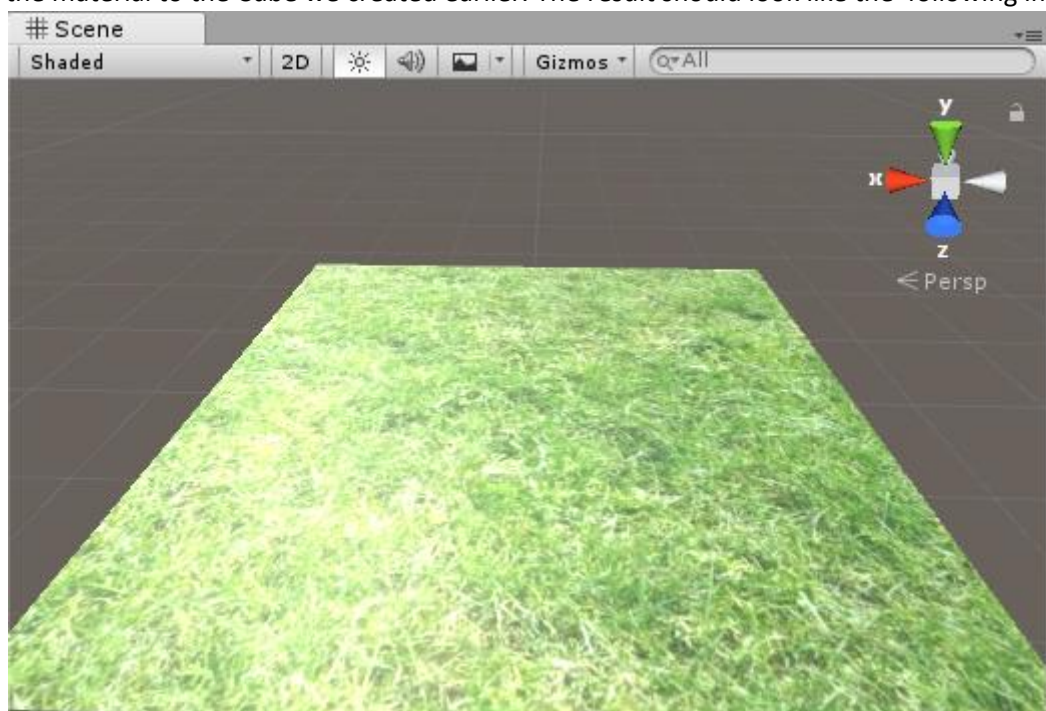


Figure 3 Our cube after adding the Material to it.

- Now select the Cube in the inspector or from the Scene window and go to the inspector.  
You should notice that the *Default Material* changed to *Grass001*.

- Expand the Material and change the *Tiling* in  $x$  and  $y$  to 5. Observe the difference.

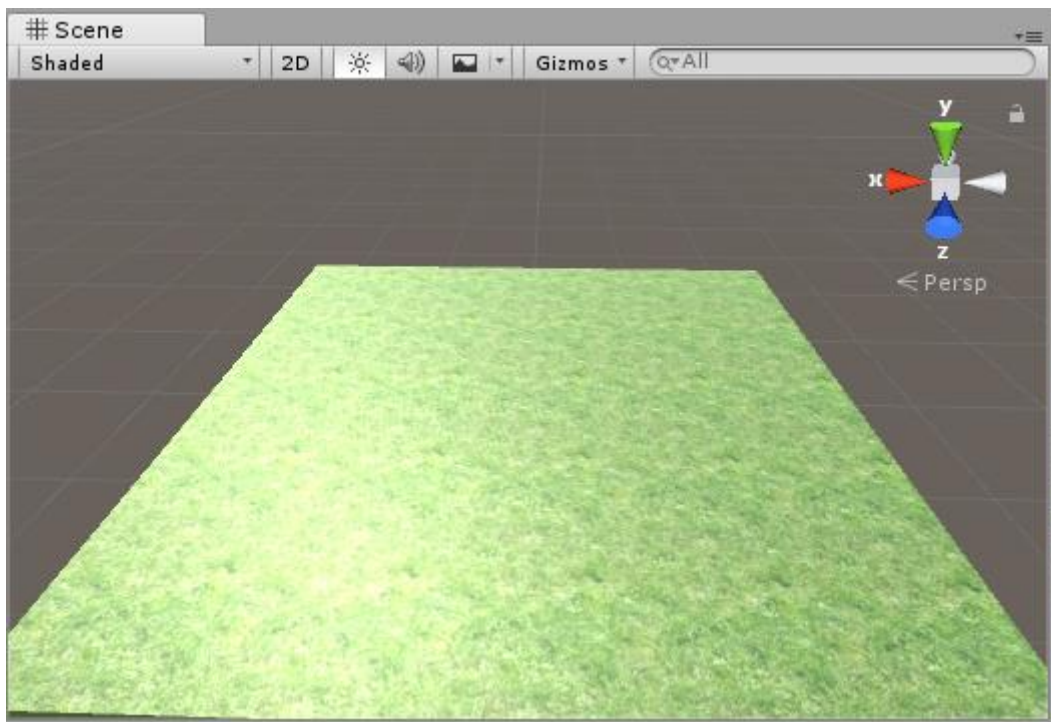


Figure 4 After change the Tiling property in the Material Component of the Cube GameObject

### Materials, Shaders & Textures

Rendering in Unity uses **Materials**, **Shaders** and **Textures**. All three have a close relationship.

- **Materials** define how a surface should be rendered, by including references to the Textures it uses, tiling information, Color tints and more.  
The available options for a Material depend on which Shader the Material is using.
- **Shaders** are small scripts that contain the mathematical calculations and algorithms for calculating the Color of each pixel rendered, based on the lighting input and the Material configuration.
- **Textures** are bitmap images. A Material can contain references to textures, so that the Material's Shader can use the textures while calculating the surface color of a GameObject. In addition to basic Color (Albedo) of a GameObject's surface, Textures can represent many other aspects of a Material's surface such as its reflectivity or roughness.

By following the previous steps, you will finish making your ground. Now we want to make the houses.

- We will create the house by using Cubes as Walls by adjusting their position, scale and rotation.
- Create a new Cube GameObject and reset its position to 0,0,0.
- Change its name to "Wall (1)"
- Change its scale as follow:
  - Set  $x$  to 0.5
  - Set  $y$  to 8
  - Set  $z$  to 10

- Change the *y* position of the Cube to make it on the top of our ground.
  - For me the *y* value was 4.5.

### Prefabs

It is convenient to build a *GameObject* in the scene by adding components and setting their properties to the appropriate values. This can create problems, however, when you have an object like an NPC, prop or piece of scenery that is reused in the scene several times. Simply copying the object will certainly produce duplicates but they will all be independently editable. *Generally, you want all instances of a particular object to have the same properties, so when you edit one object in the scene, you would prefer not to have to make the same edit repeatedly to all the copies.*

Unity has a *Prefab* asset type that allows you to store a *GameObject* object complete with components and properties.

- The prefab acts as a template from which you can create new object instances in the scene.
- Any edits made to a prefab asset are immediately reflected in all instances produced from it, but you can also *override* components and settings for each instance individually.

- Create a new *Prefab* by selecting **Asset > Create Prefab** and then drag our *Wall (1)* *GameObject* from the scene onto the “empty” prefab asset that appears.
- Rename the Prefab to “Wall”
- Now you can create more walls with the same properties by dragging the *Wall prefab* to the scene or to the Hierarchy.
- Use the Wall prefab to create 4 walls, that will make the 4 walls of the house.
- Rotate two of them by 90 degrees in *y* axis to make the front and back walls.
- Create another two walls to make the floor and ceil.  
You can do that by rotating by 90 degrees around *y* and *z* axis.
- You should end up to something like the following.



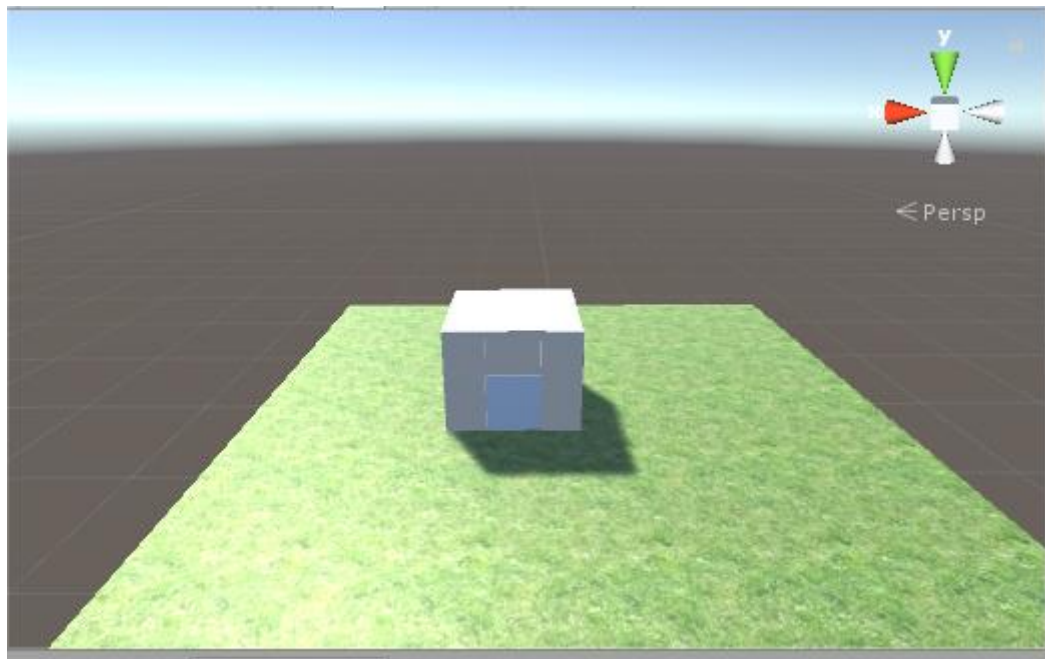
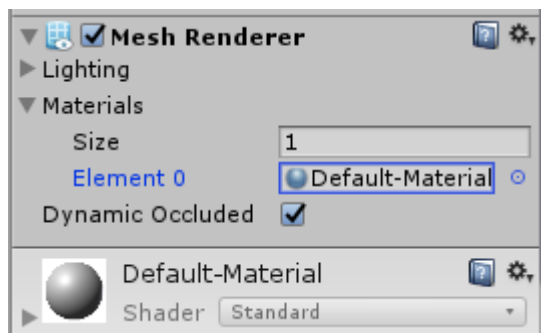


Figure 5

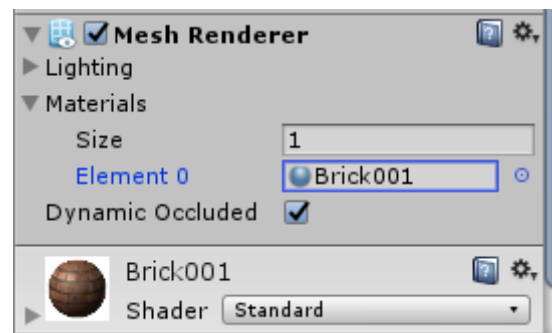
Now we want to add texture to the walls of the house:

- Add "Brick001.jpg", "Brick002.jpg" and "Stone001.jpg" to the texture folder.
- Create a new *material*, name it "Brick001" in the Materials folder.
- Select our "Wall" prefab and expand the *Mesh Renderer* component, you will find *Material* property under it and beneath it you will find "Size", default is 1, and "Element 0" set to Default-Material. Change the Element to "Brick001" material.

Notice that changes are applied to all instances of the prefab.



a



B

Figure 6 Mesh Renderer before (a) and after (b) changing the material

- Now select the "Floor" wall, change its name to "Floor" and drag the "Stone001" texture to it.  
The following had happened behind the scene:
  - A material with the same name of the Texture, "Stone001" had been created.
  - The newly created material had been assigned to the "Floor" GameObject in the *Mesh Renderer* component.
- Do the same thing to the Ceil wall but change the texture to "Brick002".  
Your scene should now look like the following image.

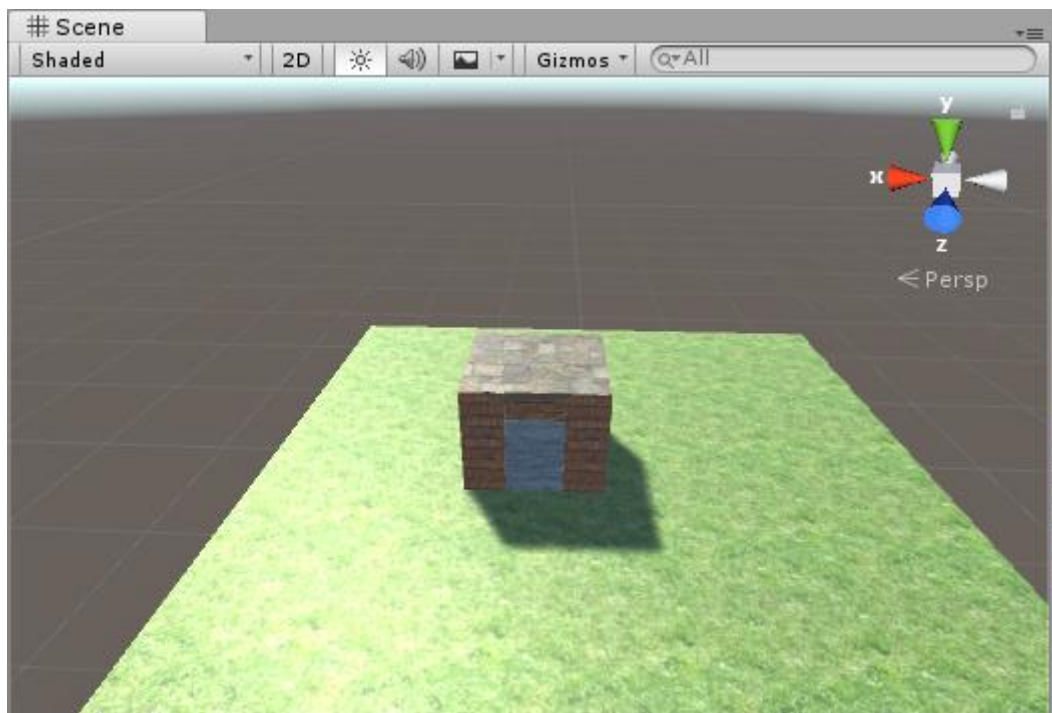


Figure 7

Now our simple house is composed of 8 GameObjects as shown in the image below:



Figure 8 Our house is composed of Wall (1 - 6), Ceil and Floor.

### Parenting

Unity uses a concept called **Parenting**. When you create a group of objects, the topmost object or Scene is called the “parent object”, and all objects grouped underneath it are called “child objects” or “children”. You can also create nested parent-child objects (called “descendants” of the top-level parent object).

It will be very useful to make the house as one big object. We can achieve that by making all the 8 GameObjects making the House children of a “House” object:

- Create a new Empty object and rename it to “House”.
- Select all the GameObjects making our house and drag them into the “House” GameObject.

Now you can move the whole house by moving the House GameObject which contain all Wall GameObjects which make our house.



Figure 9 After making our house Wall GameObjects as children to the House GameObject

Now we want to make a prefab of the house, simply drag the House GameObject to the Asset folder and a new prefab will be created for you.

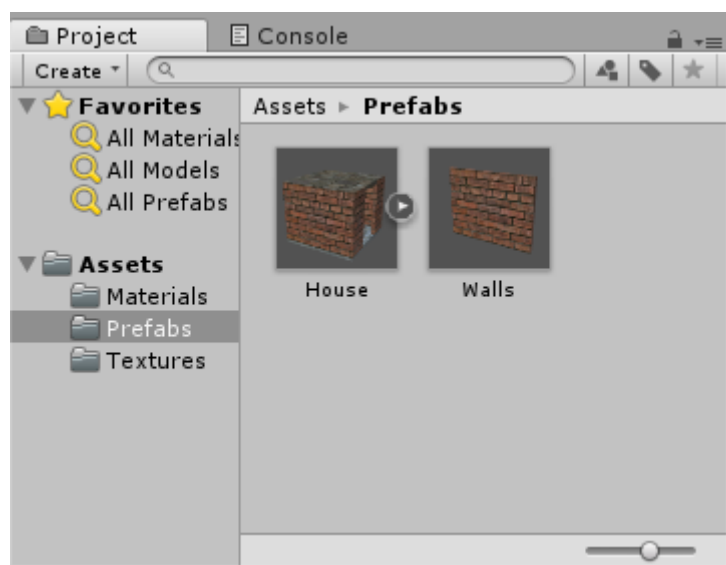


Figure 10 House Prefab. Note the button on the House prefab, it indicate that it is a composite GameObject.

Use the “House” prefab to create two more houses. Try to Rotate one of them and Scale the second.

### Asset Packages

Unity **packages** are a handy way of sharing and re-using Unity projects and collections of assets; Unity **Standard Assets** and items on the Unity **Asset Store** are supplied in packages, for example.

**Packages** are collections of files and data from Unity projects, or elements of projects, which are compressed and stored in one file, similar to Zip files. Like Zip files, a package maintains its original directory structure when it is unpacked, as well as meta-data about assets (such as import settings and links to other assets).

### Import Package

You can import

- **Standard Asset Packages**, which are asset collections pre-made and supplied with Unity.
- **Custom Packages**, which are made by people using Unity.

Choose **Assets > Import Package >** to import both types of package.

Unity '**Standard Assets**' consist of several different packages:

- 2D,
- Cameras,
- Characters,
- CrossPlatformInput,
- Effects,
- Environment,
- ParticleSystems,
- Prototyping,
- Utility,
- Vehicles.

To import a new Standard Asset package:

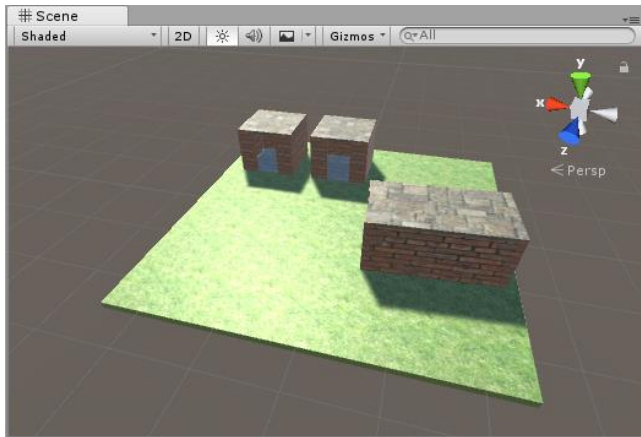
1. Open the project you want to import assets into.
2. Choose **Assets > Import Package >** plus the name of the package you want to import, and the **Import Unity Package** dialog box displays, with all the items in the package pre-checked, ready to install.
3. Select **Import** and Unity puts the contents of the package into a **Standard Asset** folder, which you can access from your Project View.

Let's insert a Character from the **Standard Assets**, this will enable us to walk in our scene.

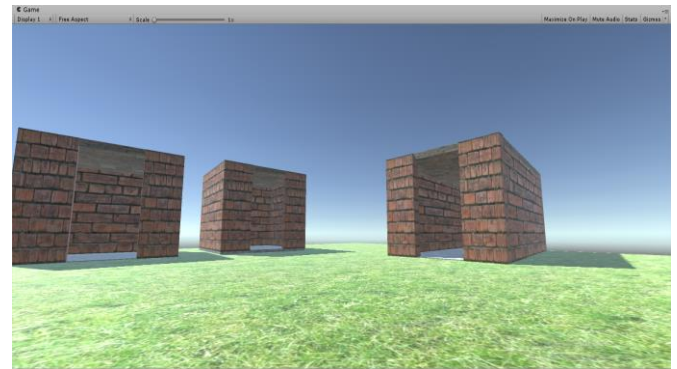
- Choose **Assets > Import Package > Character**
- A new folder, **Standard Assets**, will be created after the import is finished.
- Open **Standard Assets > Characters > FirstPersonCharacter > Prefabs**, drag the *FPSController* prefab to the Scene.
- Reset its position, and adjust its y position to make it on the top of the *Ground* cube.
- Click the Play button and walk in the scene.

This document was prepared by: Abdalla Essam Abdalla





a



b

Figure 11 (a) our Scene View after adding more houses and the character. (b) The Game Window showing what character see.

Finally let us try to use Assets that other built and that is available in the **Asset Store**.

- Choose **Window > Asset Store**.  
Asset Store window will open if you not logged in, you should login first.

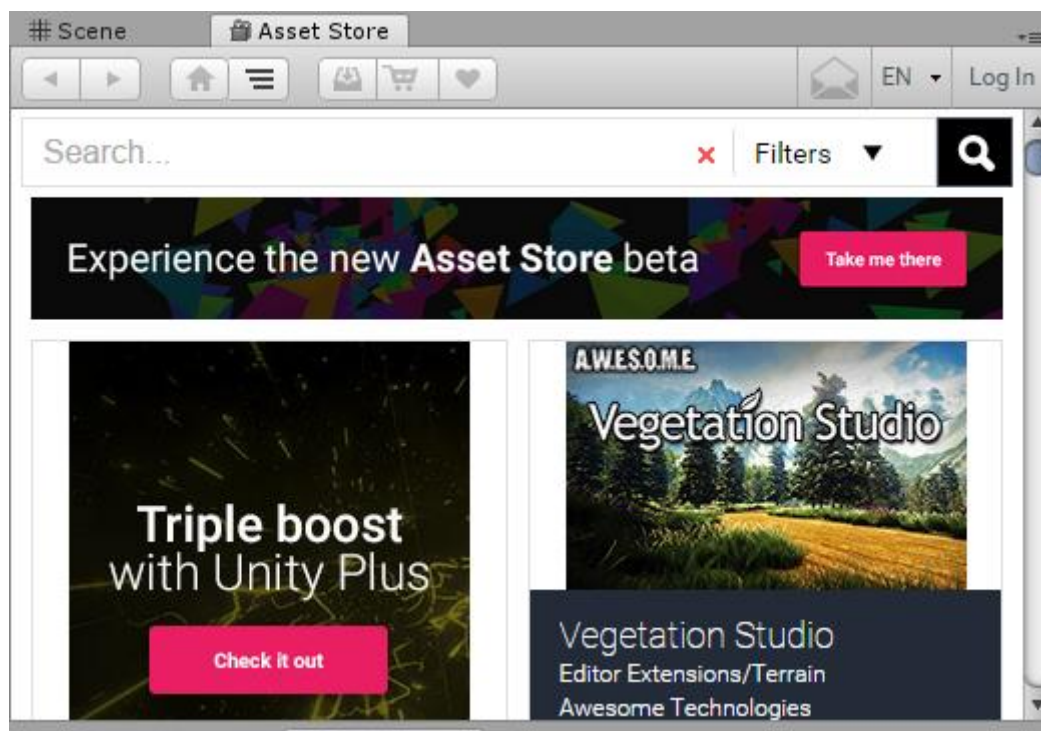


Figure 12 Asset Store Window in Unity Editor

- Search for “Town Creator Kit LITE” and open its link.
- After opening it search for “Download”. After Downloading finish click **Import**. Import Unity Package will open keep all element selected and click import.

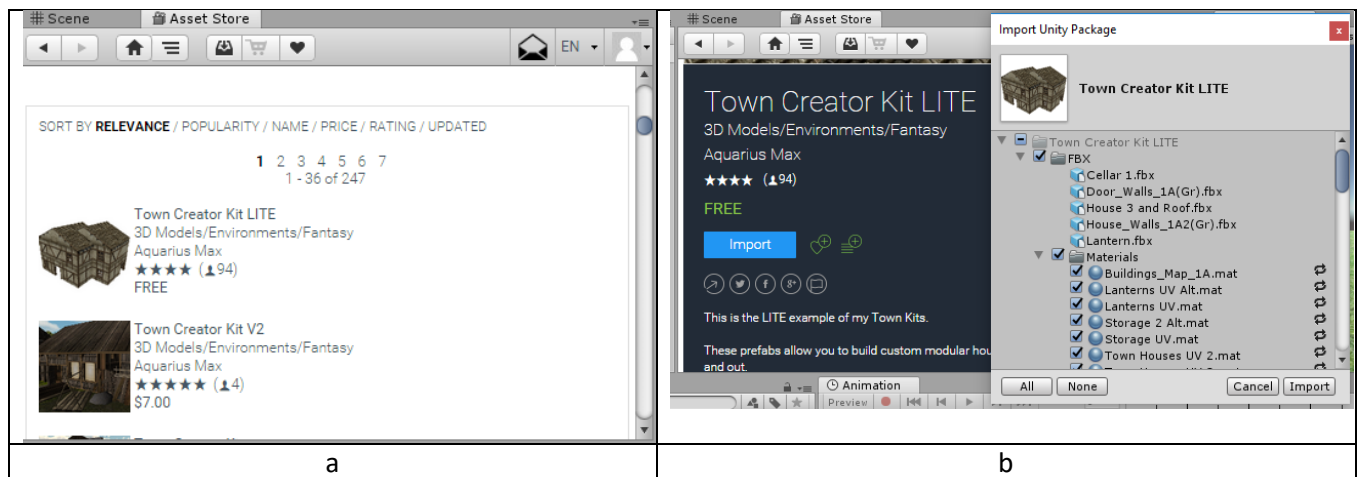


Figure 13

- When importing is finished, a new folder with the name of the package, Town Creator Kit LITE, will be created in the Project Window.
- Open **Town Creator Kit LITE > Prefabs > Town Kit LITE** and choose “**House 3 and Roof**”. Add it to the Scene, adjust its position then play the game.
- Try to investigate this Package and add other models.

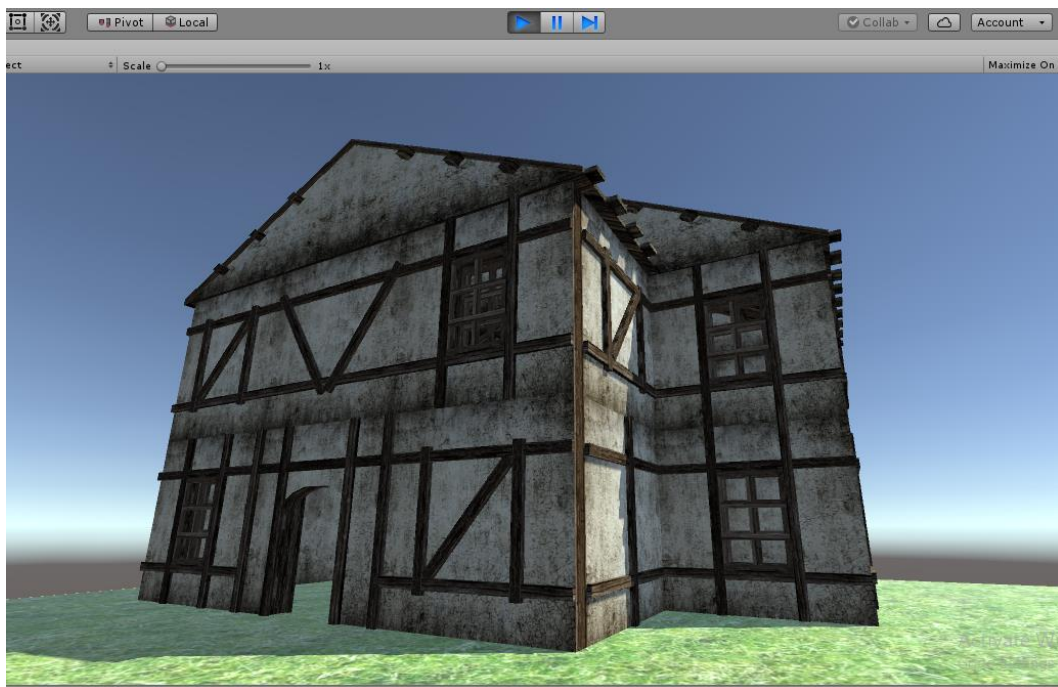


Figure 14 Part of the Game View after adding some Building from the package

## Terrain

Unity's Terrain system allows you to add vast landscapes to your games.

### Creating and editing Terrains

To add a Terrain GameObject to your Scene, select **GameObject > 3D Object > Terrain** from the menu. This also adds a corresponding Terrain Asset to the Project view.

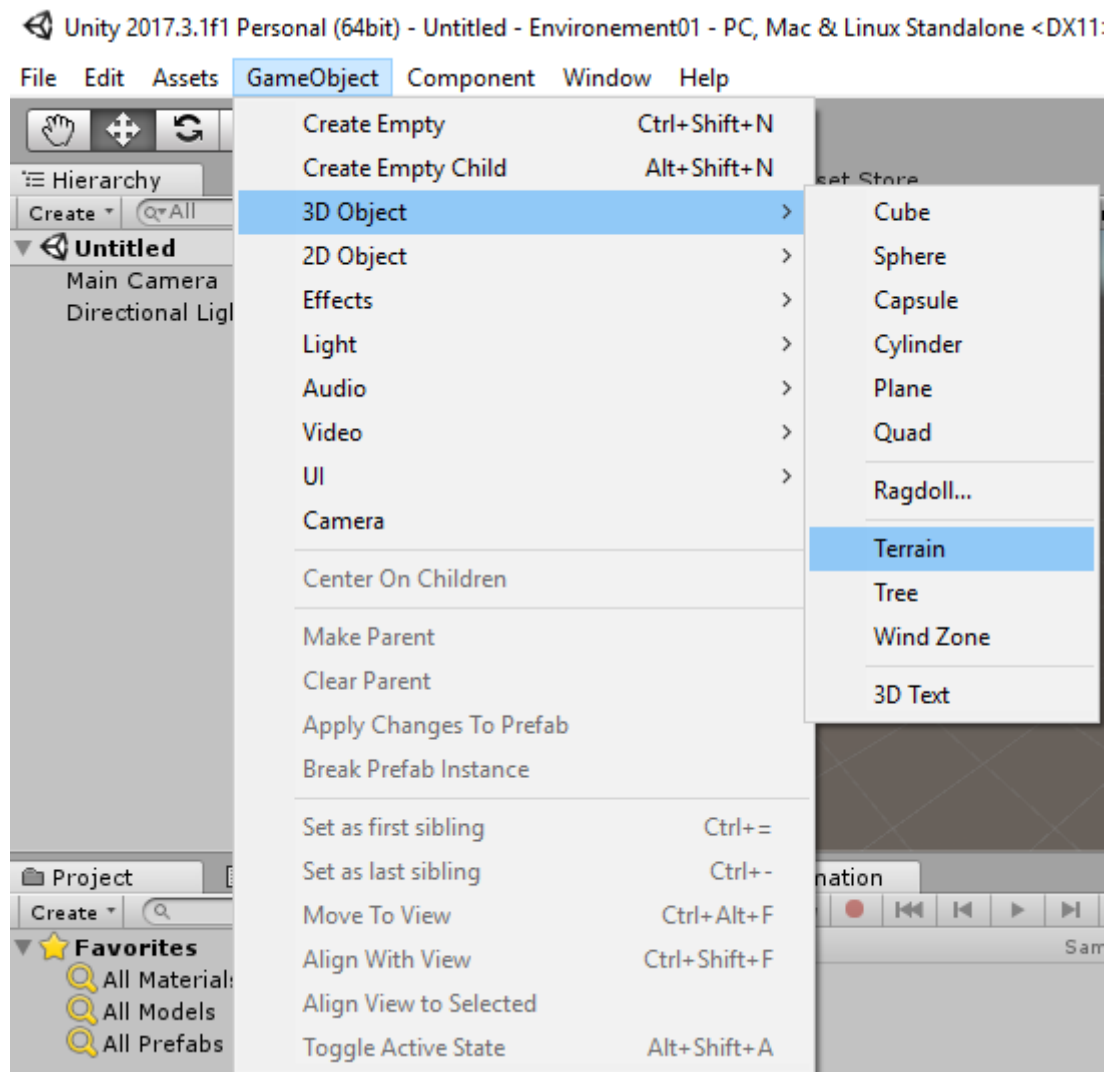


Figure 15 Insertign a new Terrain

When you do this, the landscape is initially a large, flat plane.

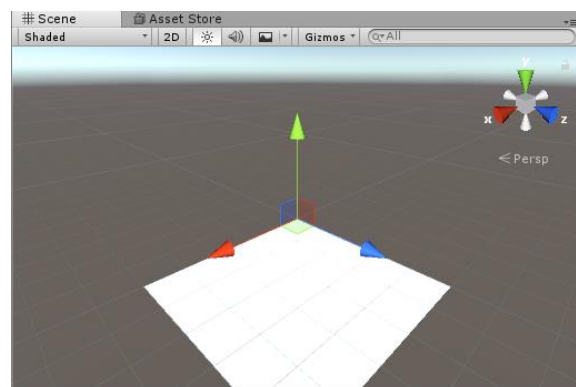


Figure 16 Newly created Terrain in Scene View

The Terrain's Inspector window provides several tools you can use to create detailed landscape features.

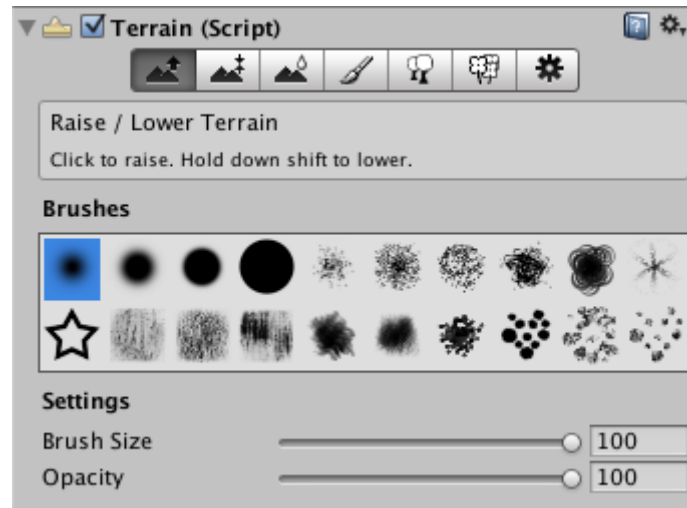




Figure 17 Terrain Editing Tools appear in the Inspector

With the exception of the tree placement tool  and the settings panel , all the tools on the toolbar provide a set of “brushes” and settings for brush size and opacity.

By selecting different options from the *Brushes* toolbar, you can paint with different shapes. The *Brush Size* and *Opacity* options vary the area of the brush and the strength of its effect respectively.

## Height Tools

The first three tools on the terrain inspector toolbar are used to paint changes in height onto the terrain.



Figure 18

**Raise/Lower**  tool:

When you paint with this tool, the height will be increased as you sweep the mouse over the terrain.

The height will accumulate if you hold the mouse in one place.

If you hold down the shift key, the height will be lowered.

The different brushes can be used to create a variety of effects. For example, you can create rolling hills by increasing the height with a soft-edged brush and then cut steep cliffs and valleys by lowering with a hard-edged brush.

**Paint Height**  tool:

It is like the *Raise/Lower* tool except that it has an additional property to set the target height.

When you paint on the object, the terrain will be *lowered* in areas above that height and *raised* in areas below it.

You can use the *Height property* slider to set the height manually or you can *shift-click* on the terrain to sample the height at the mouse position.

Height property is a *Flatten button* that simply levels the whole terrain to the chosen height. This is useful to set a raised ground level, say if you want the landscape to include both hills above the level and valleys below it.

*Paint Height* is handy for creating plateaux in a scene and also for adding artificial features like roads, platforms and steps.

**Smooth Height**  tool:

It does not significantly raise or lower the terrain height but rather averages out nearby areas. This softens the landscape and reduces the appearance of abrupt changes.


This document was prepared by: Abdalla Essam Abdalla

You might use this, for example, when you have painted detail using one of the noisier brushes in the available set. These brush patterns will tend to introduce sharp, jagged rocks into a landscape, but these can be softened using *Smooth Height*.

## Terrain Textures

You can add texture images to the surface of a terrain to create coloration and fine detail. Since terrains are such large objects, it is standard practice to use a texture that repeats **seamlessly** and tile it over the surface (the repeat generally isn't noticeable from a character's viewpoint close to the ground).

One texture will serve as the "background" image over the landscape but you can also paint areas of different textures to simulate different ground surfaces such as grass, desert and snow. The painted textures can be applied with variable transparency, so you can have a gradual transition between grassy countryside and a sandy beach, for example.

The **paintbrush**  button on the toolbar enables texture painting.

Initially, the terrain has no textures assigned for painting. If you click the *Edit Textures* button and select *Add Texture* from the menu, you will see the *Add Terrain Texture* window. Here you can set a texture and its properties.

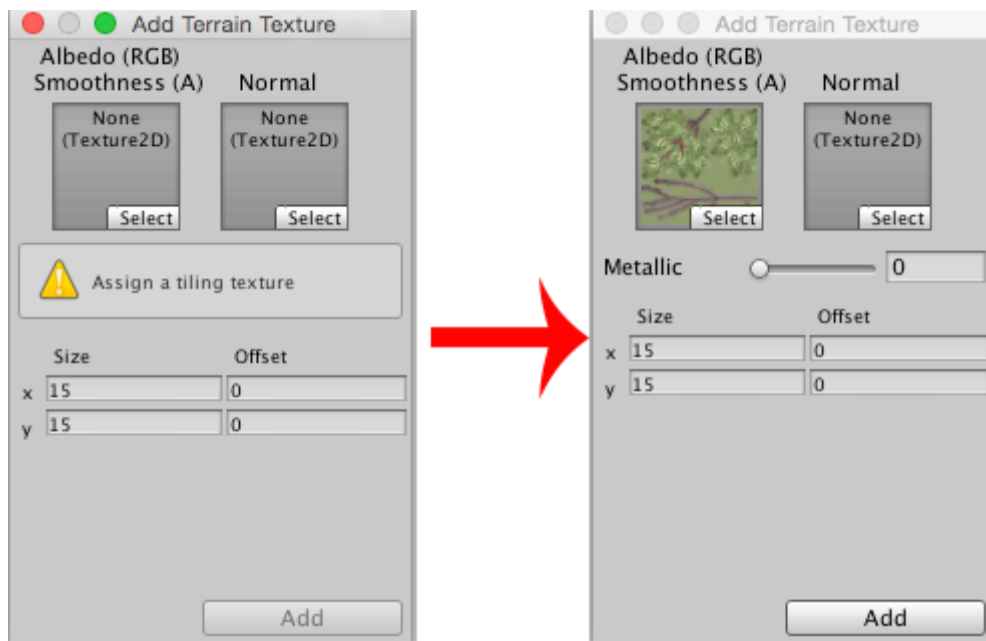


Figure 19 Click on *Select* in the *Add Terrain Texture* window and choose a texture asset from the *Select Texture* window (not shown) - it then displays, ready to add to the terrain

## Exercise

Change our Ground Cube to terrain and then add a Texture to the terrain.

Play with the brushes and try to make something good.