# Unity Terrain Engine Tools

源自<https://gamedevelopment.tutsplus.com/tutorials/unity-terrain-engine-tools--cms-28623>

What You'll Be Creating

Unity is a multi-platform game engine developed by Unity Technologies and is used to create video games and applications for a multitude of devices (PC, consoles, mobile devices, and even websites). Unity's core advantages are its robustness and portability; Unity targets several known APIs such as Direct3D, OpenGL, OpenGL ES, and the recent Vulkan.

Due to the aforementioned characteristics, Unity has become more and more popular among AAA software development houses and aspiring game programmers.

Unity supports several technologies and components. One of the key components is the terrain engine system. Unity's terrain system allows you to create vast landscapes for your games or applications. You can use a selection of tools available to create terrains easily and quickly.

This tutorial will focus on explaining how terrain engine tools work, and how to use them to create vast and rich terrain environments.

**Prerequisites**

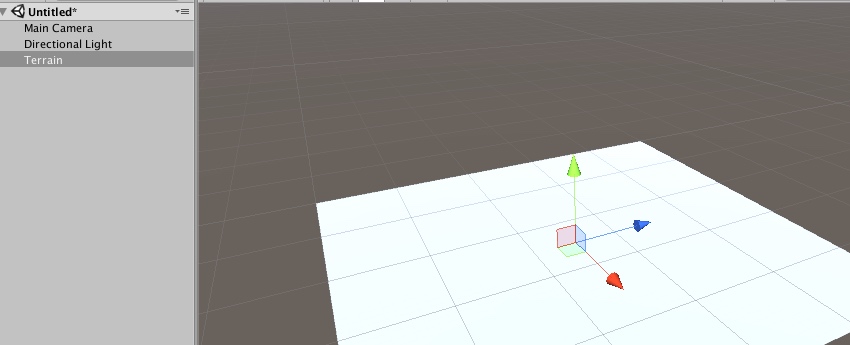
First, ensure you have the [latest version of Unity](https://unity3d.com/get-unity/download). In this tutorial we're using version **5.6**. Make sure that you are using the latest Unity version; otherwise you may find small differences following the tutorial and using the physics joints.

For this tutorial, you will not use any starter file. The goal is to create a new project and perform the tutorial from there.

**The Terrain Tools**

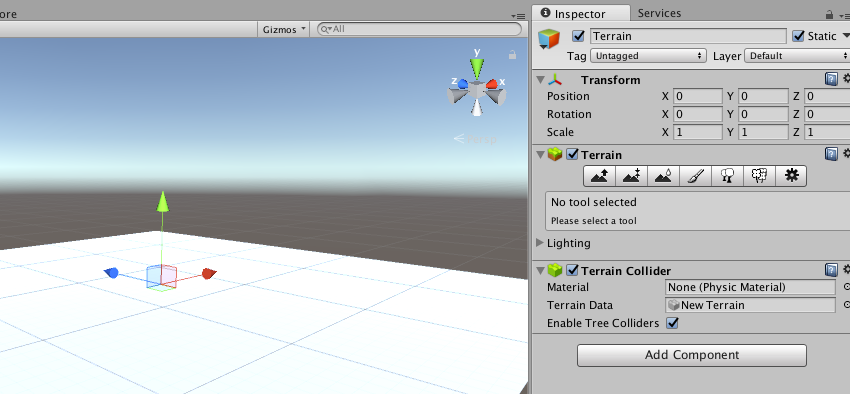
The first step is to create a new Unity project. Now add a new scene (**File > New Scene**). The new scene has been created with two game objects, the **Main Camera** and a **Directional Light** (Unity 5 will automatically add a skybox into the scene).

Now, in order to add a terrain game object, go to the **GameObject** menu, and select **3D Object > Terrain**. This will add a flat plane (called **Terrain**) into your scene. This plane is the mesh that we are going to sculpt in order to create your level terrain.



If you check your **Assets** directory (bottom part of Unity), you will see that a **New Terrain** file was created.

Select the Terrain and look at the **Inspector** tab. You will see that the terrain has three components: **Transform**, the **Terrain** script, and the **Terrain Collider** component.



The terrain component provides you the tools you need to edit your terrain. All the tools on the toolbar, with the exception of the tree placement tool and the settings panel, provide a set of "brushes" and settings for brush size and opacity, just like painting tools from a normal image editor. This allows you to sculpt your terrain similarly to a painting tool.

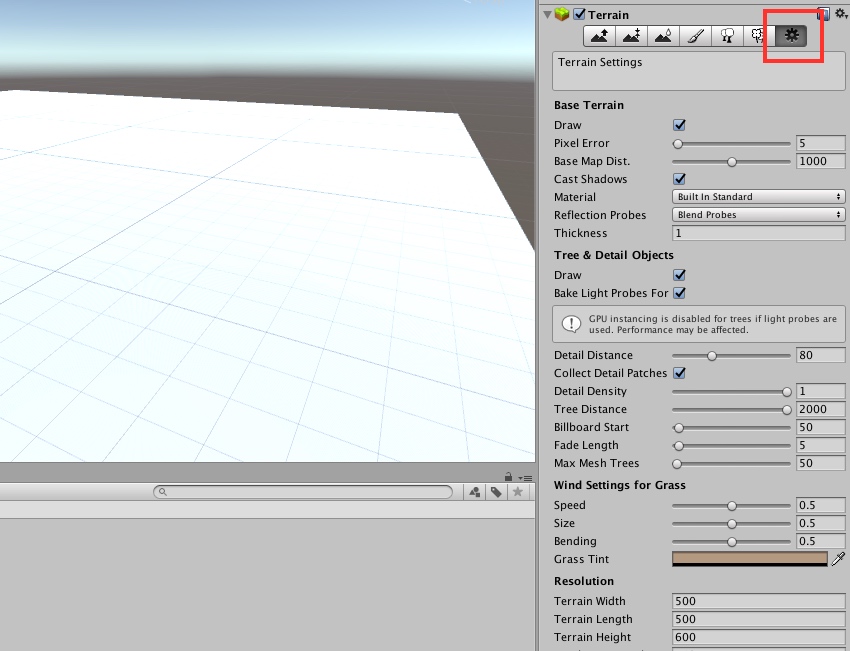
If you select any brush under the **Terrain** component and mouse over the terrain, you will see a blue area projected on the terrain. This lets you know the area of the terrain that your brush will affect.

As you may have already realised, the **Terrain** is quite big. To navigate through the **Scene**, you can use the mouse and keyboard. Inside the **Scene** tab, press and hold your **Right mouse button** (the cursor should change to an eye).

While holding the mouse button:

* You can change the angle of your view by moving the mouse.
* Use the **W, S, A, D** keys to move around the 3D scene.

Before you start exploring the terrain tools, let’s take a look at the **Terrain** properties. The last button on the Terrain component (inside the **Inspector**) will show you the base parameters of the terrain.



The properties are divided into sections:

* Base Terrain
* Tree & Detail Objects
* Wind Settings for Grass
* Resolution
* Heightmap

Inside the first section **Base Terrain**, you will find several properties and parameters.

The **Draw** option toggles the rendering of the terrain on or off. The **Pixel Error** represents the accuracy value of the mapping between the terrain maps (for example textures, heightmaps, and generated terrain). Higher values represent lower accuracy, and lower values will result in rendering overhead. It is your responsibility to balance this value in order to create a nice, rich environment.

The **Base Map dist.** is the maximum distance at which terrain textures will be displayed at full resolution. You can set shadows on or off by selecting **Cast Shadows**.

Next, we have **Material**. Here you set the material that is going to be used to render the terrain. This will affect how the color channels of a terrain texture are interpreted. You can select:

* **Built In Standard** represents the Physically-Based Rendering material that was introduced in Unity 5. If you select this option, for each splat layer, you can use one texture for albedo and smoothness, one texture for normal, and one scalar value to tweak the metalness. Keep in mind that if "**Overwrite Smoothness**" is checked, instead of reading from texture maps, the smoothness of the entire terrain will be controlled only by the **Smoothness** value.
* **Built In Legacy Diffuse**represents the legacy built-in terrain material from previous Unity releases (Unity 4 and backwards). It uses a Lambert lighting model and has optional normal map support.
* **Built In Legacy Specular** uses the Blinn-Phong lighting model ([please consult this previous Unity 5 lightning tutorial](https://cgi.tutsplus.com/tutorials/lighting-in-unity-5--cms-27992)), and has optional normal map support. This option also allows you to specify the overall specular color and shininess for the terrain.
* **Custom**option uses a custom material of your choice to render the terrain. This material should be a shader that is orientated for terrain rendering. This is an option that I only recommend to advanced users. However, if you want to give it a try, take a look at the source code of some of the built-in terrain shaders and make modifications on top of them.

You can also set **Reflection Probes**for some materials, but they will only be used when you’re using a built-in standard material or a custom material which supports rendering with reflection. The options for reflection probes are:

* **Off**: Reflection probes are disabled, and only the skybox will be used for reflection.
* **Blend Probes**: Reflection probes are enabled. The blending occurs only between probes. Default reflection will be used if there are no reflection probes nearby, but there will be no blending between default reflection and probe.
* **Blend Probes And Skybox**: Reflection probes are enabled. Blending occurs between probes or probes and default reflection.
* **Simple**: Reflection probes are enabled, but no blending will occur between probes when there are two overlapping volumes.

Finally, the last parameter in this section is **Thickness**. This value specifies how much the terrain collision volume should extend along the negative Y-axis. Objects are considered colliding with the terrain from the surface to a depth equal to the thickness. This helps to prevent high-speed moving objects from penetrating into the terrain without using expensive continuous collision detection.

The next settings section is **Tree & Detail Objects**.

The first option is **Draw**. Here, you can set the draw of the details on or off. Basically, it tells the engine if you want trees and grass to be drawn or not. Next, you have **Bake Light Probes** for trees. If you enable this option, Unity will create light probes at the position of each tree. The probes will then be applied to the rendering of the trees. If you disable this option, trees will still be affected by light group probes. Take note that this option is only available for trees that have light probes enabled on their prefabs.

The **Detail Distance** is the distance from the camera beyond which details will be culled. **Detail Density**representsthe number of detail objects in a given unit of area. **Tree Distance** represents the distance from the camera beyond which trees will be culled. The **Billboard Start** is the distance from the camera at which 3D tree objects will be replaced by billboard images. The **Fade Length** represents the distance over which trees will transition between 3D objects and billboards. Finally, the **Max Mesh Trees** is the maximum number of visible trees that will be represented as solid 3D meshes.

Moving on, you'll see the **Wind Settings for Grass** section.

**Speed**defines the speed of the wind as it blows grass. The **Size** represents the ripple size on grassy areas. **Bending** lets you definethe degree to which grass objects are bent by the wind. **Grass Tint** lets you select the overall color tint applied to grass objects.

Inside the **Resolution** section you will find several resolution properties, namely the **Terrain Width**, **Length**, and **Height**. All three are very explicit of their meaning. The **Heightmap Resolution** is the pixel resolution of the terrain’s height map. It should be a power of 2 plus 1, for example, 513, where 512 is a power of 2, and you add 1.

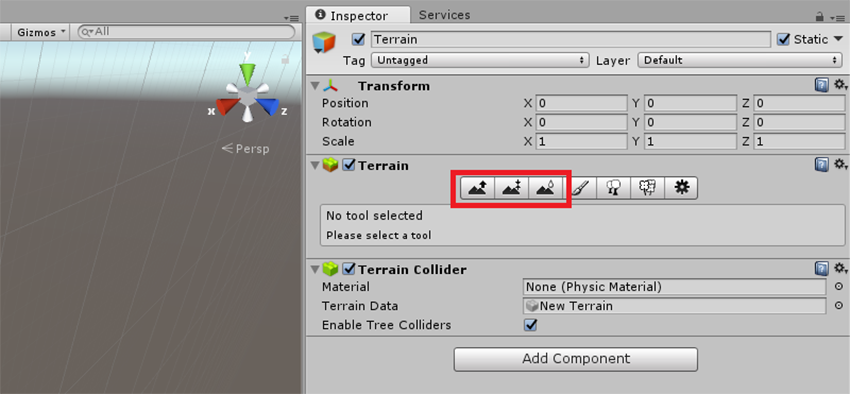
**Detail Resolution** lets you define the resolution of the map. Higher resolutions will give you smaller and more detailed grass patches. The **Detail Resolution per Patch** sets the length and width of the square of patches rendered on a single draw call. The **Control Texture Resolution** is the resolution of the splat map that controls the blending of the different terrain textures. Finally, the **Base Texture Resolution** sets the resolution of the composite texture used on the terrain when viewed from a distance superior to the base map distance.

The last section is **Heightmap**. Here you can import a heightmap raw file or export your current terrain into a heightmap raw file. This can be very useful if you want to use external third-party software to interact with your application or game.

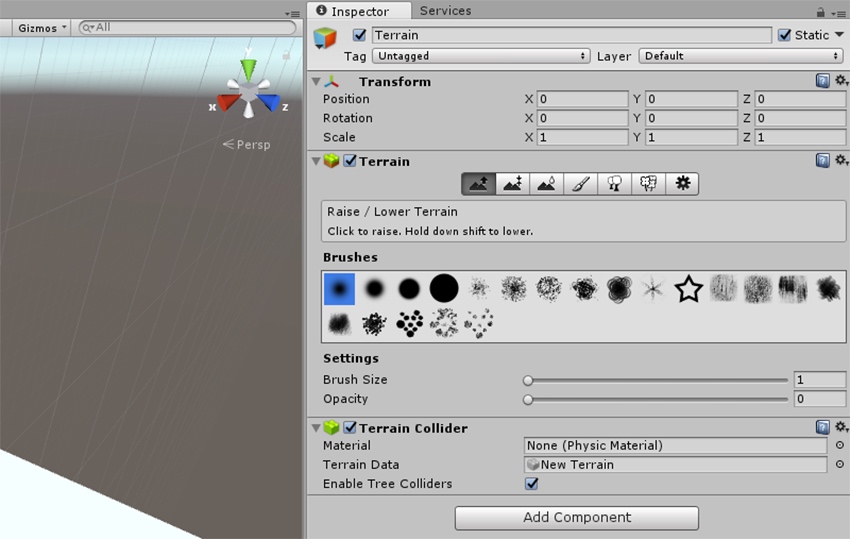
**Height Tools**

Before you move on to the height tools, remember that you must set the resolution settings first. Otherwise, if you try to change them later, all of the sculpting of the terrain will be lost.

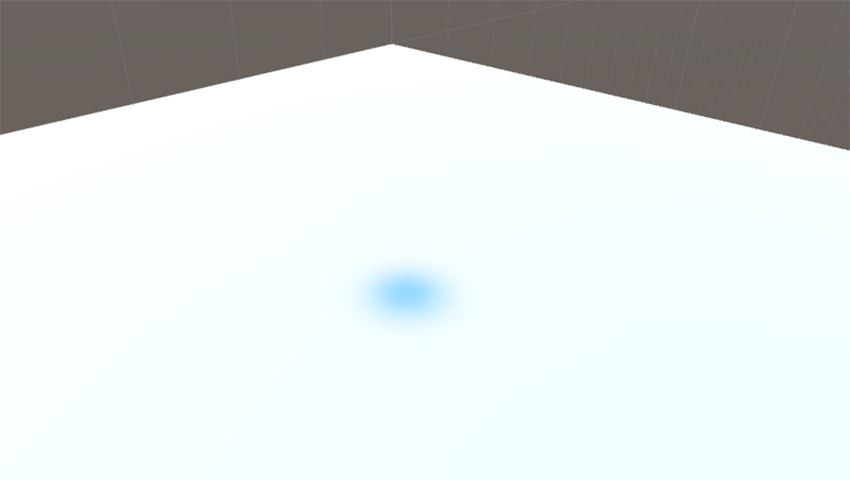
The first three tools on the **Terrain Inspector** toolbar are the height tools. These tools are used to paint height changes onto the terrain.



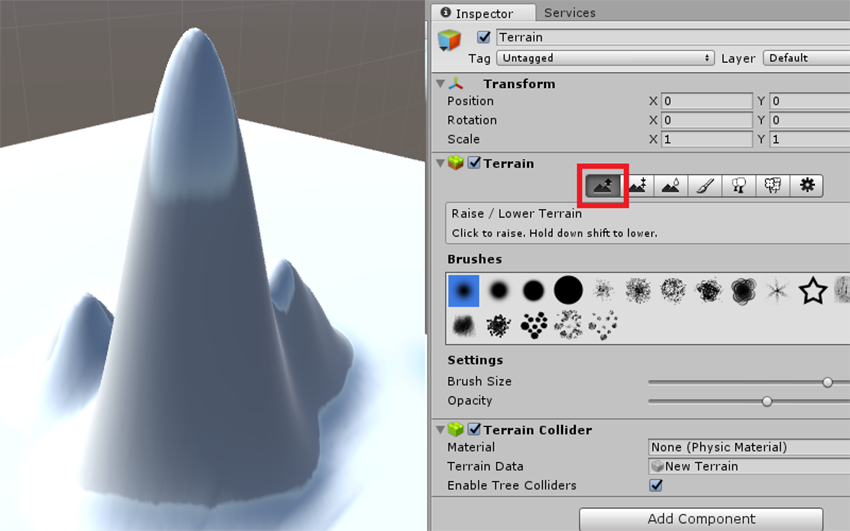
They provide a set of "brushes" with settings for **Brush Size** and **Opacity**; these tools are very similar to painting tools from any image editor. This allows you to paint detail in the terrain, as if you were painting an image.



If you move your cursor to the **Scene** tab, you will see a blue area projected on the terrain. This lets you know the area of the terrain that your brush will affect.

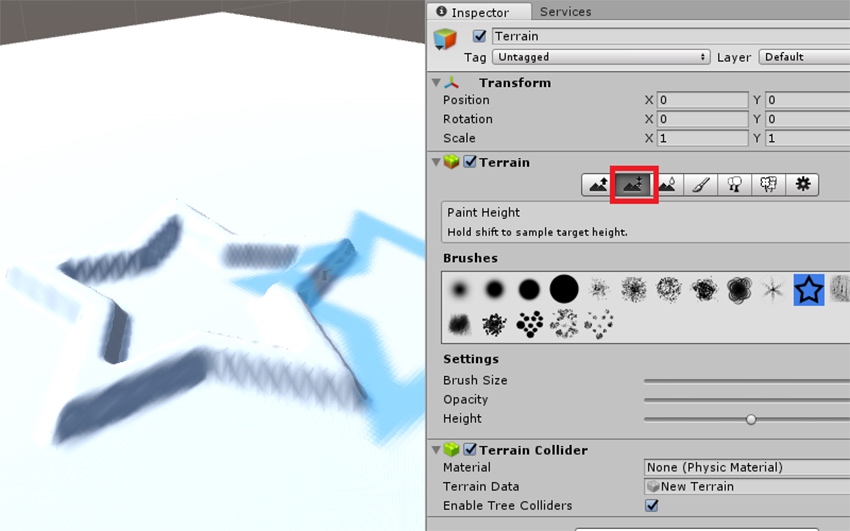


The first tool is **Raise / Lower Height**. When painting with this tool, the height of the terrain will be increased as you sweep the mouse over it. If you hold the mouse in one particular area, the height will accumulate. Such an effect can be visualized in the following image.

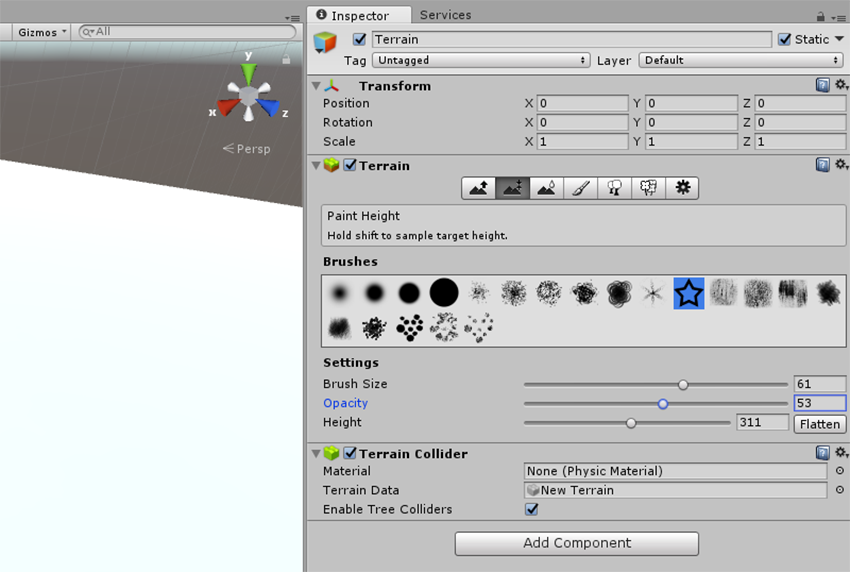


To lower the terrain, just hold the **Shift** key while painting with the tool, and this will lower the height of your terrain.

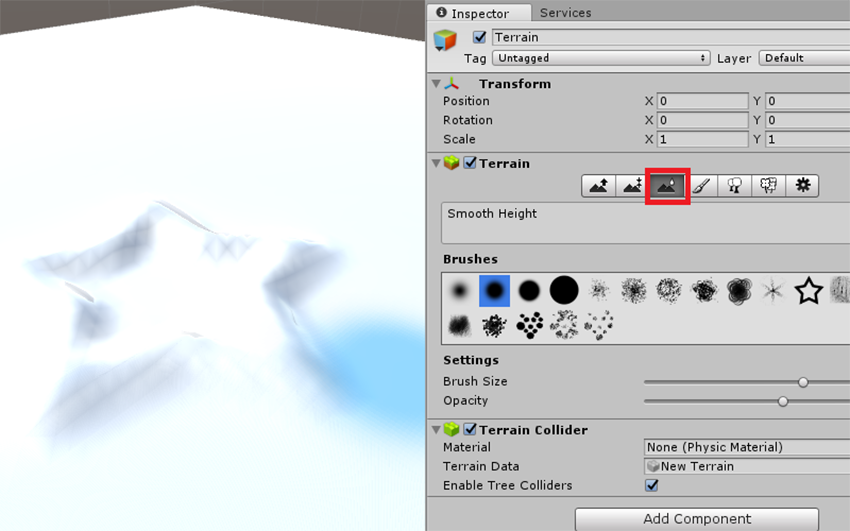
The second tool on the terrain editor is **Paint Height**. This tool is used to set a specific height for an area of the terrain.



It works in a similar way to the **Raise/Lower** tool, but it has an additional property to set the target **Height**. You can select the desired **Height** by changing the value of the **Height** parameter. The **Flatten** button next to the height property will level the whole terrain to the chosen **Height**.



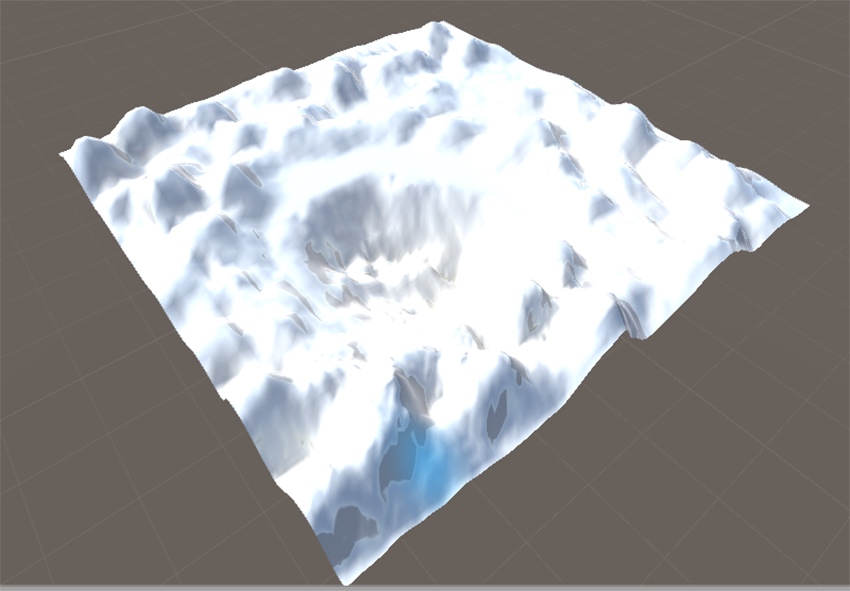
Finally, the third tool is **Smooth Height**. This tool will not drastically increase or decrease the height of the terrain; instead, it will soften the landscape and reduce the appearance of abrupt changes. This tool is particularly useful when you have painted detail using one of the noisier brushes, since these brushes tend to create sharp, jagged rocks in the landscape, and you can use this tool to soften them.



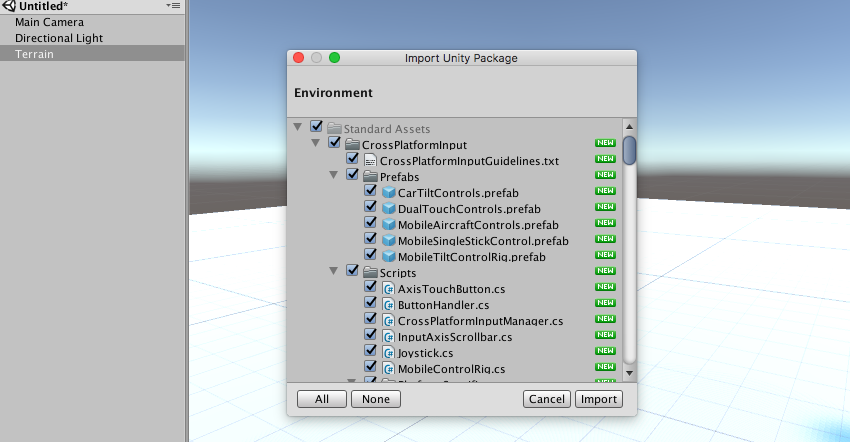
It is now time to take some time and create your own sculpted terrain. Try to use the aforementioned three tools to create a nice mountain terrain.

**Paint Texture Tool**

With the terrain sculpted, is now time to texture it.

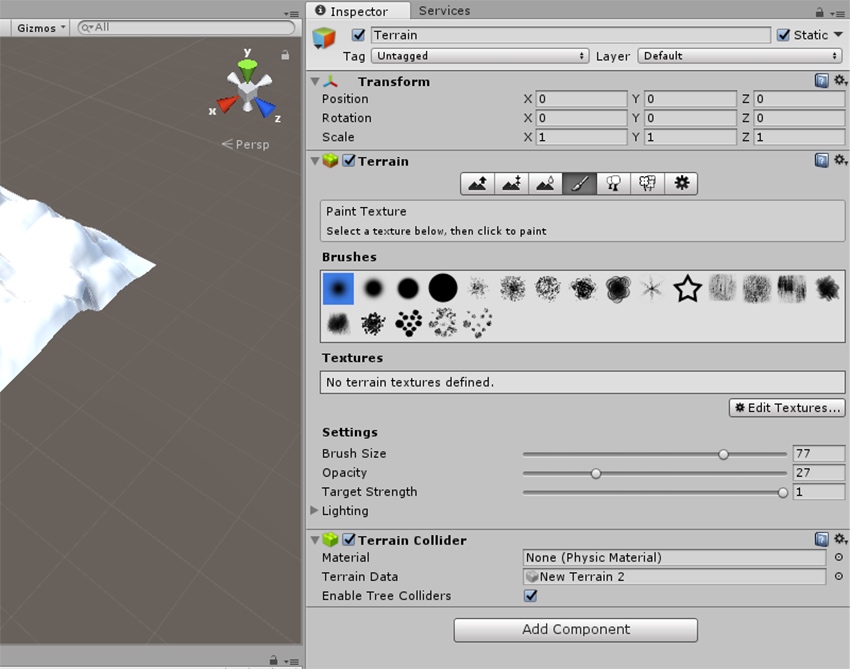


Before moving on, you need to import the environment assets that are available with Unity 5. To do that, select **Assets > Import Package**, and select **Environment**. Click **Import**to import all environmental assets.

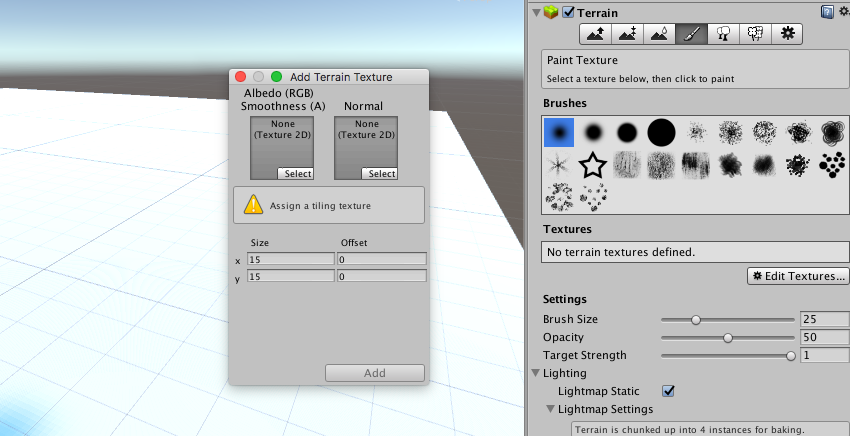


Unity allows you to paint your terrains with textures. You can paint your terrain with different textures in different areas. For example, you can have a texture for rock, another for sand, and another for grass. The textures can be applied with different transparency, which means that you can have smooth transitions between two textures. In order to achieve the best results, you should paint your terrain with seamless textures.

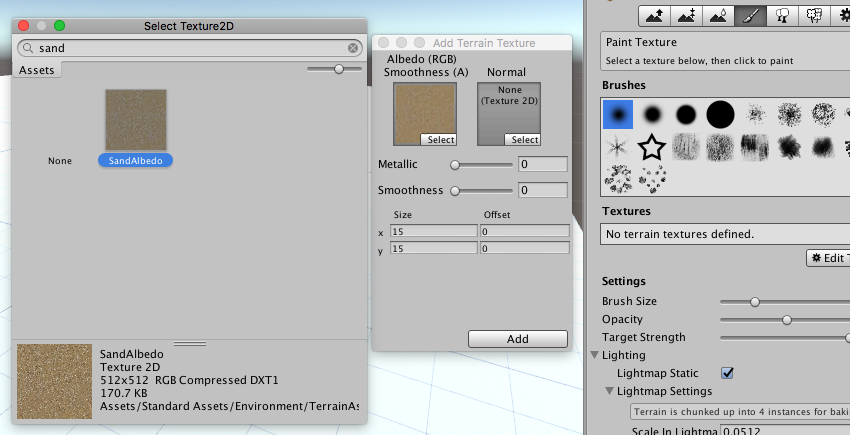
As you may have noticed on the **Inspector** tab, the **Paint Texture** tool has several brushes, and lets you change the **Brush Size**, the **Opacity**, and the **Target Strength**.



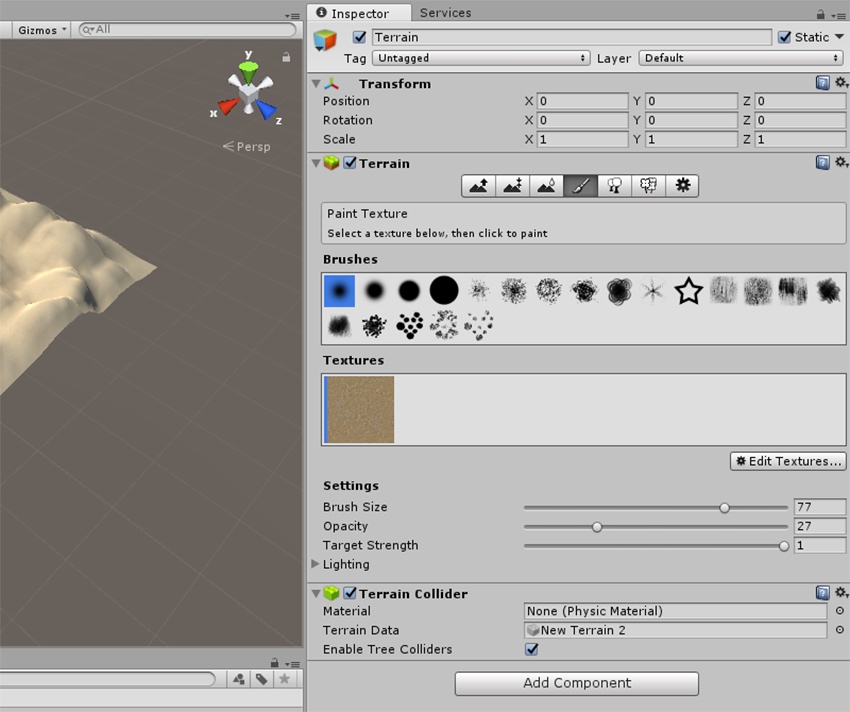
The first time you use the paint **Texture Tool**, there won’t be any textures to choose from. The first thing you need to do is to add a new texture into your pallet. To do this, click on the **Edit Textures** button and select **Add Texture**. A new window will appear.



As you can see, on this window you can **Select** the texture you want to use and its normal map, if you have one. Click on the square under texture and select your texture. A window will pop up to let you select the texture. Select the **SandAlbedo** texture and click **Add**.



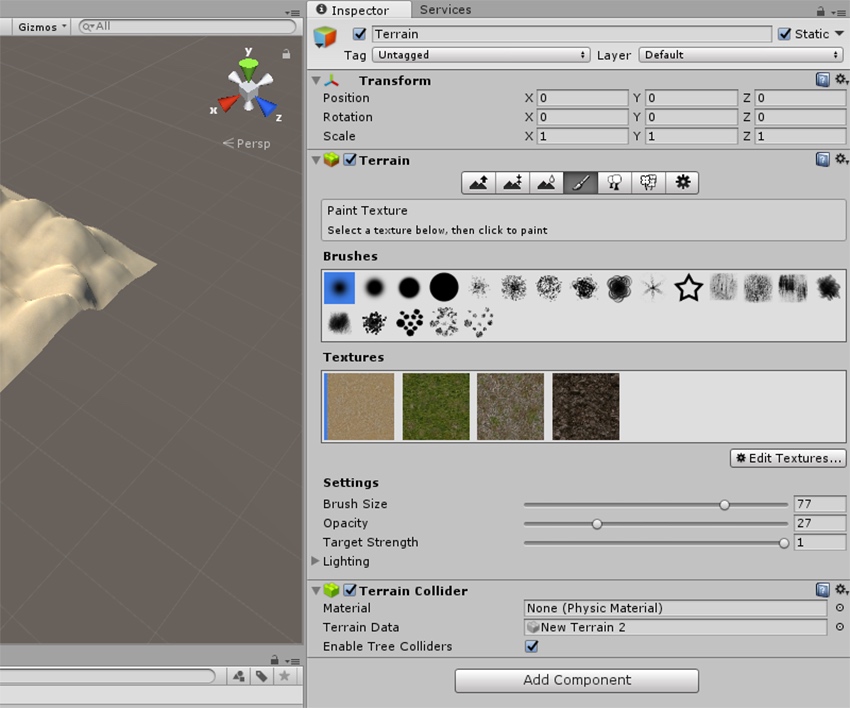
Your terrain should now have a sand look. Note that you can always modify it on the edit texture option, or remove it.



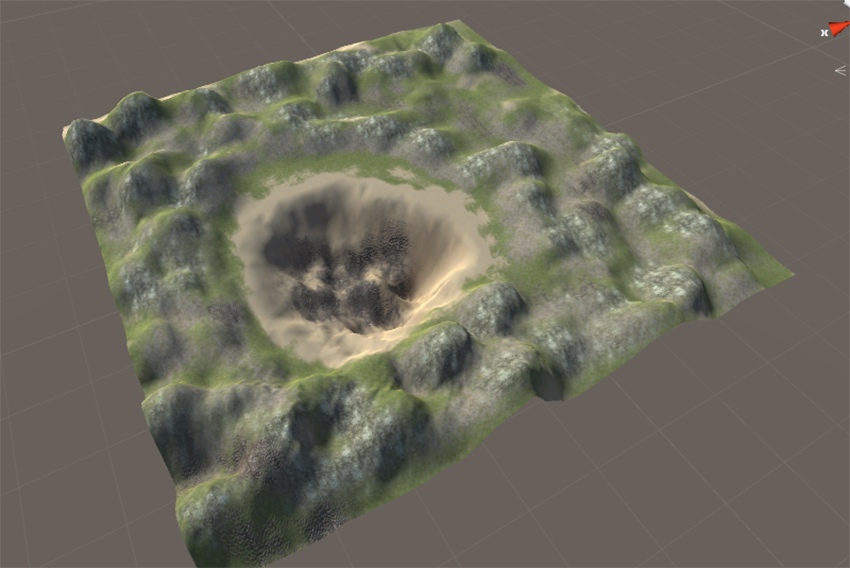
The first texture you create should always be the base texture for the terrain. The complete list of textures used in this example are:

1. SandAlbedo
2. GrassHillAlbedo
3. GrassRockAlbedo
4. MudRockyAlbedo
5. CliffAlbedo

All of them are part of the environmental imported assets and can be found in the same way as the **SandAbedo** texture. Go ahead and import them all.



When you have all the textures imported, it is now time to texture your terrain as you want. Select a texture and paint the desired environment. An example of a textured map is the following.

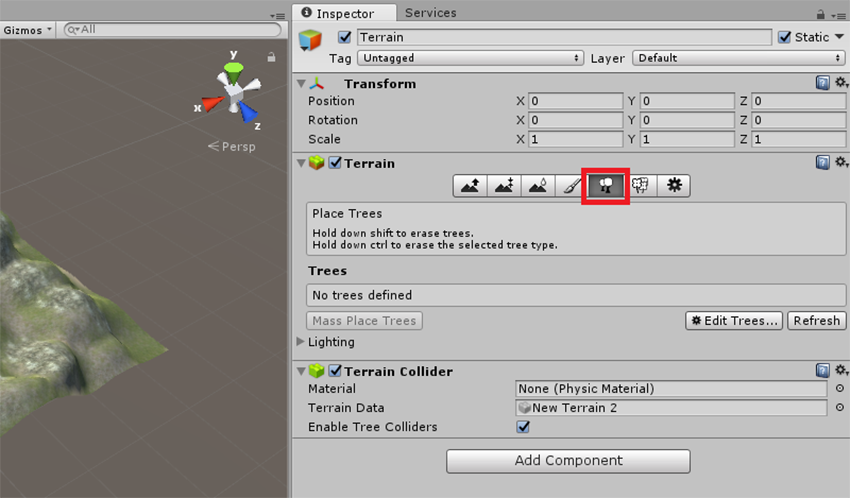


When you are satisfied with the result, move on to the next section.

**Paint Tree Tool**

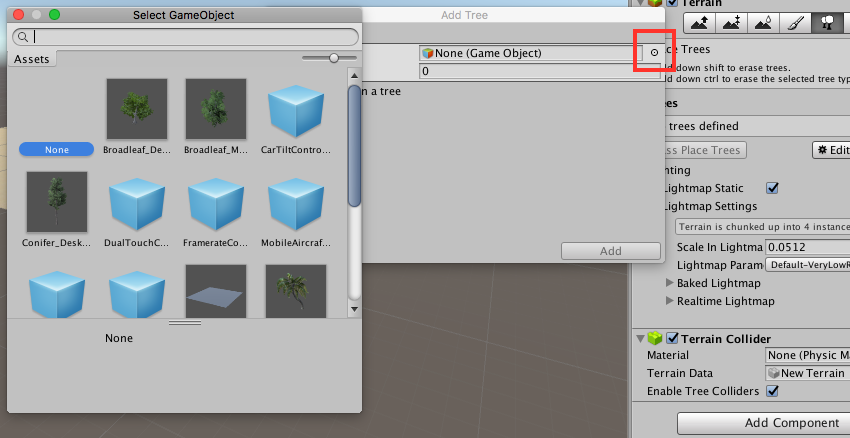
Now that you have your terrain painted, it’s time to add some trees to it. Unity 3D allows you to put trees on your terrain in the same way as you painted the textures. Using billboarding settings for distant trees, Unity maintains a good rendering performance, allowing you to have dense forests with thousands of trees.

To start painting trees, select your **Terrain** from the **Hierarchy** tab, and on the **Inspector** tab, select the fifth button.

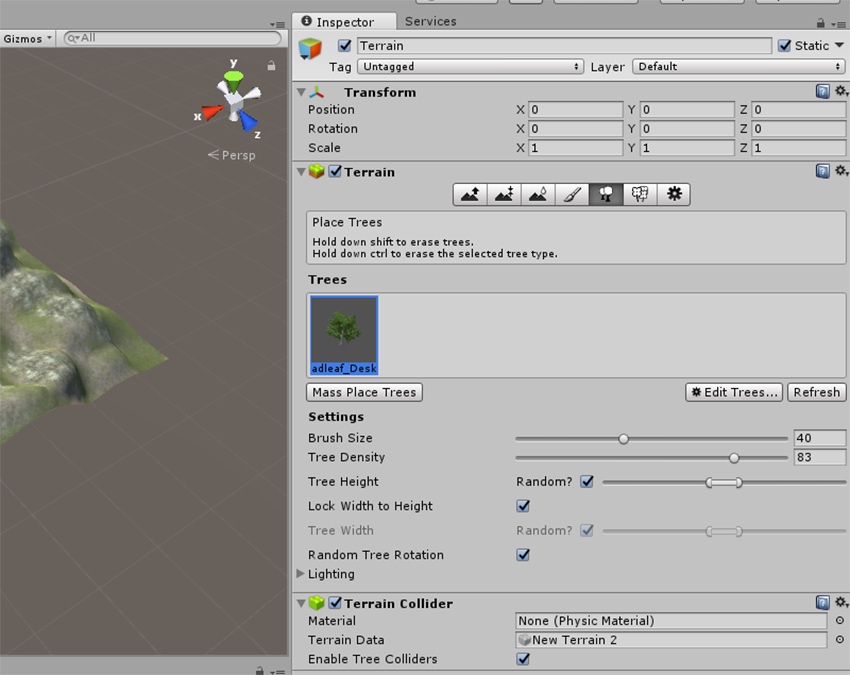


Similar to textures, you also need to add the trees. Click on **Edit Trees > Add Tree**.

Unity will ask you for the prefab of the tree you want to use. Click on the small circle, and another window will show up. This time, it will show you all the prefabs you have in your project.



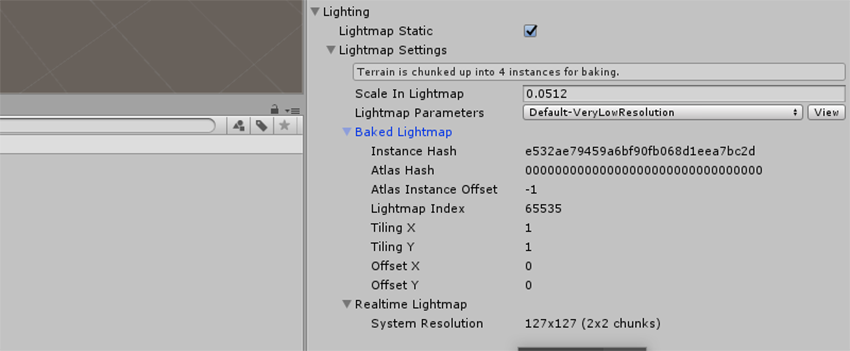
Select the **Broadleaf\_desktop** tree prefab and click **Add**. This will close the prefab window and add the selected tree into the project.



You can now select the palm tree to paint it onto your terrain.

Under the **Settings**, you will find several parameters that you can adjust while you are placing trees in your scene. **Brush Size** defines the area you will paint, while **Tree Density** defines the number of trees you will have in the painted area. High values will mean more trees. **Tree Height**will allow you to define how the height changes. You can set it to random and use an interval of values, or you can set a specific value. You can **Lock Width to Height** of the trees, in order to make them have the same aspect ratio. You can also randomize the tree rotation orientation using the **Random Tree Rotation** option. Also, the tree rotation can be set to random.

Inside the **Lighting** section you can set how the lightmaps are generated for your trees.



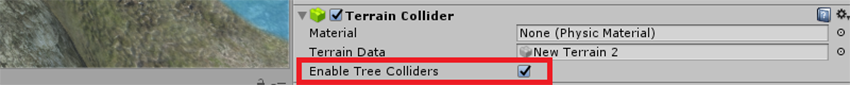
In order to paint the trees in the terrain, move the mouse over into your scene view. The blue area represents the brush. With a tree type selected, press the **left mouse** **button** on the terrain and start creating trees.



To erase trees from a certain area, hold**Shift** and use the **left mouse button**. This will clear all trees in the brush area. If you just want to erase a certain type of tree, hold **Control** instead of Shift. This will clear just the selected tree type.

You can always change or delete a type of tree by selecting it and clicking on the edit tree button. To modify a tree, click on edit, and to delete it, click on remove.

Finally, you will notice that inside the **Terrain Collider** component, you have a field named **Enable Tree Colliders**. By selection this option, you are enabling collisions between other game objects and the trees.



If you don’t want to paint the trees manually, you can use the **Mass Place Trees** button. If you click there, a new window will pop up. Unity will then ask you for a number of trees to create. Set the **Number of Trees** and click on **Place**. Unity will then randomly place the trees in your terrain.



The next step is being creative and populating your terrain with trees. In order to create different trees, you can also add new trees like the **Palm\_Desktop**, **Broadleaf\_Mobile** and**Conifer\_Desktop**. For that, use the option **Edit Trees** under the **Trees** menu (inside **Inspector**).

You can create something like the following image (or something completely different).

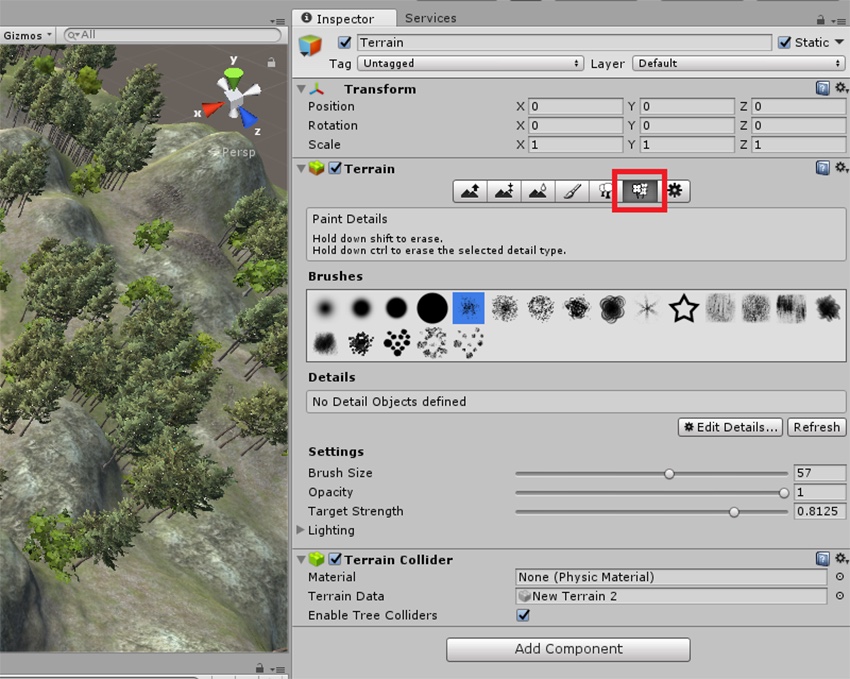


When you are ready, you can move on to the next step.

**Paint Details Tool**

Unity allows the terrains to have grass clumps and other small objects, such as rocks, covering its surface. The grass is rendered using 2D images, while other objects are usually standard meshes.

To add details to the terrain, select your terrain object from the hierarchy. Inside the **Inspector** tab, click on the sixth button, the one with the flowers.



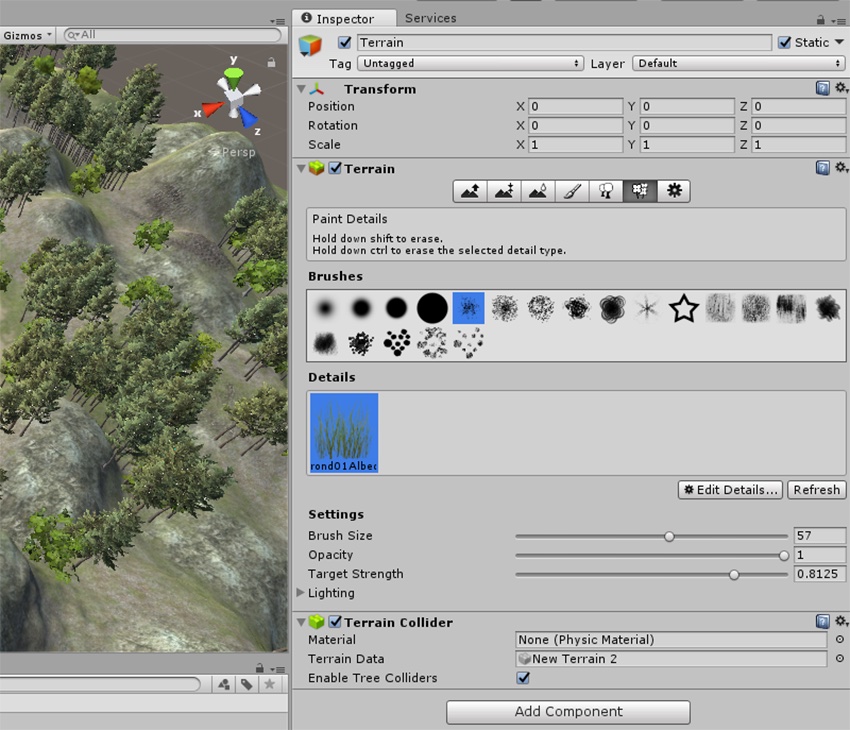
As you can see, you now have a similar interface to the **Height Tools**. Here you also have a set of brushes that you can choose from in order to paint a list of details.

Since you don't have any object details selected, let's add them by clicking on **Edit Details > Add Grass Texture**.

This will open a window where you can set several properties for your grass. **Detail Texture** lets you define the texture that you will use for your grass. You can also define several properties like **Max Width**, **Min Width**, **Max Height**, **Min Height, Noise Spread**, **Healthy Color**, **Dry Color**, and **Billboard**. All these properties will define how your grass will look.

For this tutorial, you can leave the default options or change them at your will.

Click on the circle in front of the **Detail Texture** field. This will open a new window. Search for the **GrassFrond01AlbedoAlpha**, select it, and click **Add**.



Just like for the trees, you can use several brushes to paint the grass. You can set different **Brush Sizes** and change the brush **Opacity** and the **Target Strength**.

It is now time to be creative and paint grass onto the terrain.



If you want to add a mesh to paint detail on your terrain, for example rocks, you need the repeat the same process you did with importing the grass. However, this time, when you click on**Edit Details**, choose **Add Detail Mesh**. This will open a window where you can specify several parameters of the detail mesh, very similar to the one you used to import the grass.

Unfortunately, the environment assets do not include any rock prefab, so you will not be able to paint rocks directly. However, you can search for some rocks over at the [Unity Asset Store](https://www.assetstore.unity3d.com/en/#!/), since it is a great place to find free assets.

**Conclusion**

This concludes the tutorial about Unity Terrain Tools. You learned about several terrain tools and configurations. With this knowledge, you can now create, modify and improve current or new terrains for your next cutting-edge game or application.

Unity has an active economy. There are many other products that help you build out your project. The nature of the platform also makes it a great option from which you can improve your skills. Whatever the case, you can see everything we have available [in the Envato Marketplace](https://codecanyon.net/category/unity).

If you have further questions or comments, as always, feel free to drop a line in the comments section.