8K Earth, Moon and Mars Shaders Asset V0.3.3

Documentation

Charles Pérois - 2015

Table des matières

1. Introduction

This asset aims to provide a simple and fast way to add quite realistic looking planets Earth, Moon and Mars in your project.

The asset contains:

- A demo scene to show how to use it.
- Shaders for the Earth surface (4 different shaders), clouds, clouds shadows, the atmospheric outer scattering effect, Moon Surface and Mars surface.
- A high polygon sphere.
- A set of textures.
- A sunlight example.
- A milky way skybox.
- Scripts, mainly for the demo scene.

The 4 different shaders for the earth surface are the following:

- *earthGound8k* : SM3.0 shader, it uses 4 textures and wrap them to result in an 8k textured planet. 8k algorithm is used on surface color, normal map and night lights textures.
- *earthGound8k_surfaceonly* : SM3.0 shader, it's the shader used in the previous version of the asset. It uses the 8k algorithm only on the surface color textures.
- *earthGround* : SM3.0 shader, it uses only single 4K textures.
- *earthGroundSM2.0* : compared to "earthGround", it loses the specular effect but is SM2.0 compatible.

2. Release Note

V0.3.3
- Add normal mapping on the clouds shader.
- Add 8K texturing on earth normal map.
- Add 8K texturing on the earth night lights.
- Add 8K texturing on clouds color.
- Add 8K texturing on clouds normal map.
- New script for the demo scene.

v0.3.1
- Cloud “fake” shadows shader added.
- New Mars textures and normal map: now craters do look like craters, and hills like hills.

v0.3
- Mars and the Moon added.
- Planet prefabs now works "out of the box" using tag to get the light source, and can be dynamically instantiated.
- New demo scene in which you can browse the planets.
- Updated documentation.

v0.2.1
- New Earth surface texture.
- New Clouds Texture.
- Various tweaks on the atmospheric scattering default setting to make it look better "out of the box".
- Sun Lens Flare.

v0.2
- New earth surface shader that combines the 8k texture tiling and the inner scattering effect.
- Added Specular Mapping on the earth surface shader, with shininess, color and brightness setting.

v0.1
Initial release

3. How to Use

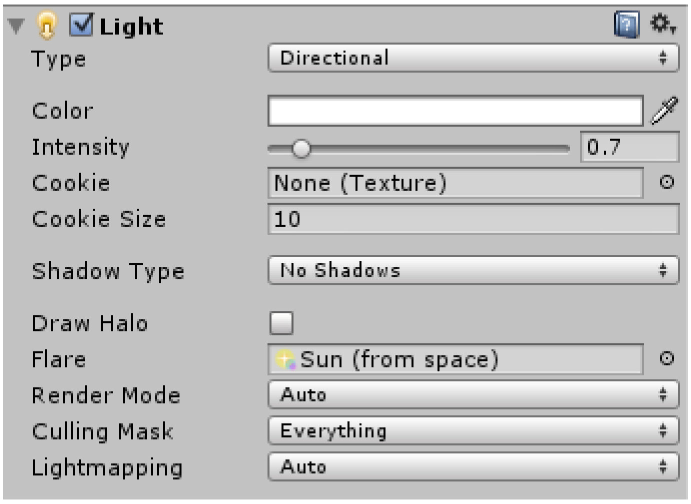
1. Set the scene

1. Set the scene with prefabs

First, copy all assets into your project.

Then, if your scene doesn't contain any sun object yet (or any object that emit a light), create it.

Add a light to the Sun object (Add component > Rendering > Light), make it directional, then add a lens flare to it (one is included in "Lens Flares/" folder).



☑ Light

Type: Directional

Color

Intensity: 0.7

Cookie: None (Texture)

Cookie Size: 10

Shadow Type: No Shadows

Draw Halo: □

Flare: Sun (from space)

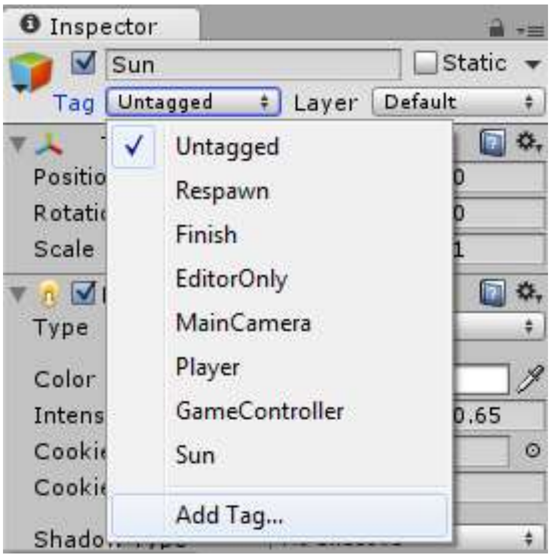
Render Mode: Auto

Culling Mask: Everything

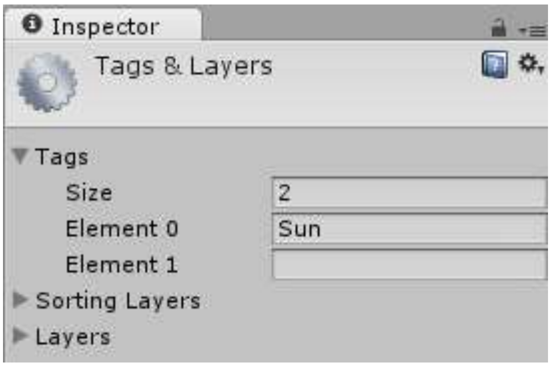
Lightmapping: Auto

Finally, add a tag "Sun" to the sun object (or the one that emit the light). To can do it this way:

Select « Add Tag… »



Add a tag named "Sun"



Inspector

Tags & Layers

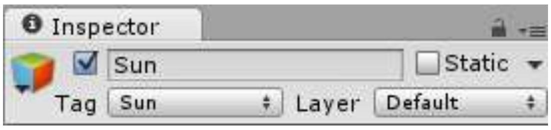
▼ Tags

Size	2
Element 0	Sun
Element 1	

▶ Sorting Layers

▶ Layers

And select it in the drop down list



Inspector

☑ Sun □ Static

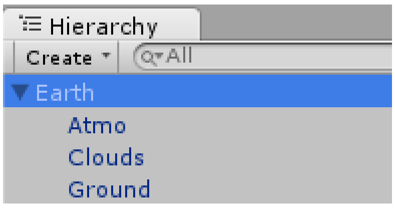
Tag Sun Layer Default

The tag have to be "Sun", this allows the planets prefabs to works without any setting, so that way you can dynamically instantiate them by code.

Now you just have to drag any of the planet prefabs to your scene. Those prefabs are contained in these folders:

Planet/Earth/Earth.prefab
Planet/Earth/Earth without shadows.prefab
Planet/Moon/Moon.prefab
Planet/Mars/Mars.prefab

Newly added prefabs should looks like this in the hierarchy view:



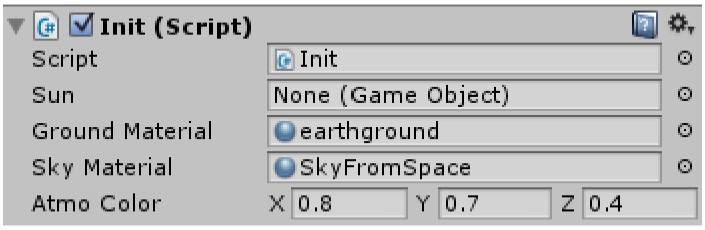
Hierarchy

Create All

▼ Earth
- Atmo
- Clouds
- Ground

Prefabs can be dragged anywhere in the hierarchy, whether on the root or inside another object.

If for some reason your Sun object can't have a tag named "Sun" on it, you can still manually assign your Sun object to the planet object by filling the **Sun** field in the Init component in the Inspector (just drag the Sun object from the hierarchy view):



▼ ☑ **Init (Script)**

Script	Init
Sun	None (Game Object)
Ground Material	earthground
Sky Material	SkyFromSpace
Atmo Color	X 0.8 Y 0.7 Z 0.4

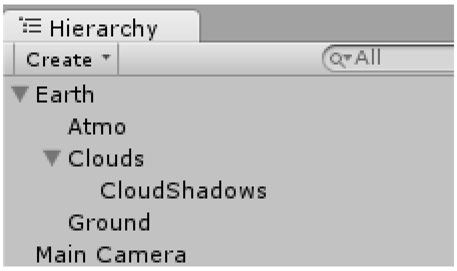
And that's it! If you want you can jump to the chapter *Outer atmospheric scattering effect*

2. Set the scene manually (Earth example)

If you want to set the scene manually (you may want to use spheres with less polygon for example), here's is how you can make this works:

First, copy all assets into your project.

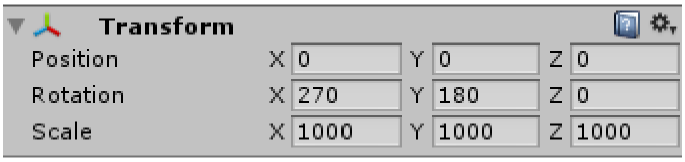
Then, to make the earth, create an empty object, and add 4 child empty object into it. Then add a sun empty object. Let's name them this way:



```
▼ Earth
    Atmo
  ▼ Clouds
      CloudShadows
    Ground
  Main Camera
```

The CloudShadows object is put inside the Cloud object to make them move together.

For floating precision sake, I would recommend you to set the "Earth" object scale to a bigger scale:



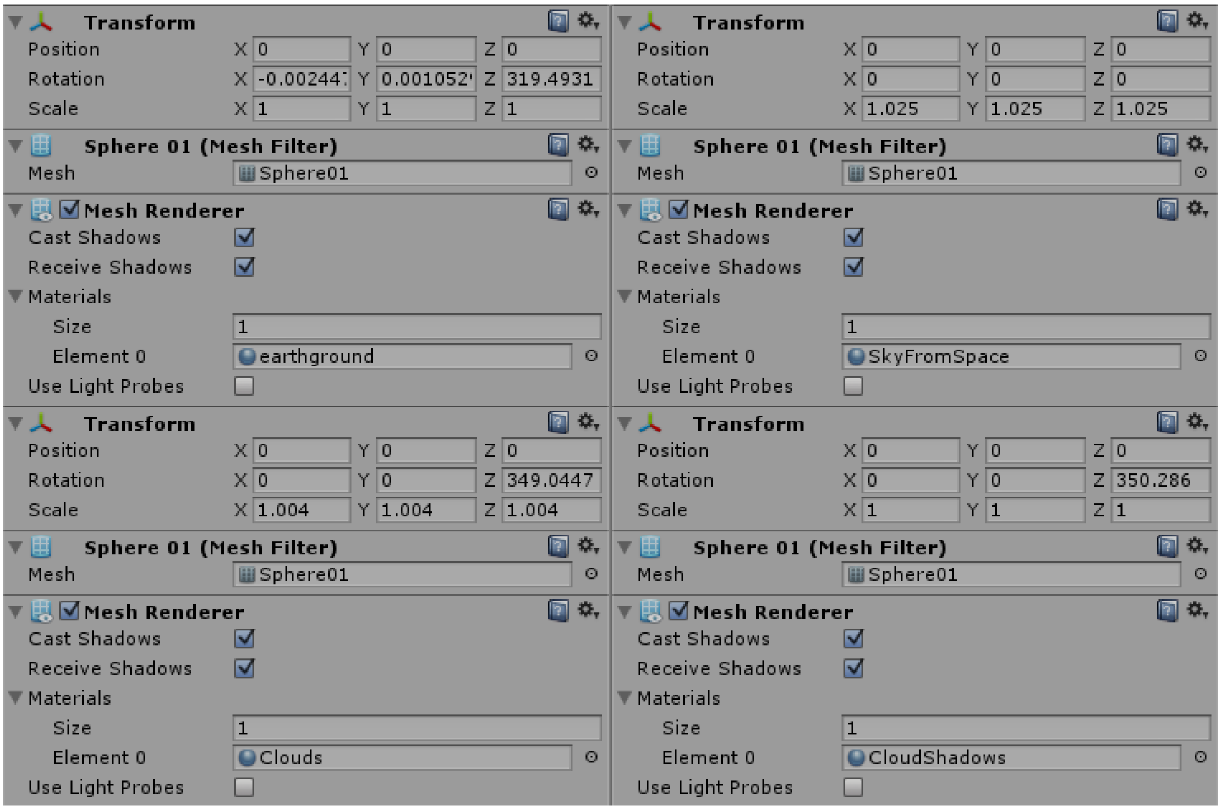
	X	Y	Z
Position	0	0	0
Rotation	270	180	0
Scale	1000	1000	1000

Now in inspector, add a mesh filter (Add Component > Mesh > Mesh Filter) on the 4 child objects, and select the mesh “sphere01”. Then add them a mesh renderer (Add Component > Mesh > Mesh Renderer).

In mesh renderer part, add the following materials:

- Atmo: SkyFromSpace.mat
- Clouds: Clouds.mat
- CloudShadows: CloudShadows.mat
- Ground: earthground.mat

Inspector should looks like this (respectively Ground, Atmo, Clouds and Cloud shadows):



Transform
Position X 0 Y 0 Z 0
Rotation X -0.00244 Y 0.001052 Z 319.4931
Scale X 1 Y 1 Z 1
Sphere 01 (Mesh Filter)
Mesh Sphere01
☑ **Mesh Renderer**
Cast Shadows ☑
Receive Shadows ☑
Materials
Size 1
Element 0 earthground
Use Light Probes ☐

Transform
Position X 0 Y 0 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1.025 Y 1.025 Z 1.025
Sphere 01 (Mesh Filter)
Mesh Sphere01
☑ **Mesh Renderer**
Cast Shadows ☑
Receive Shadows ☑
Materials
Size 1
Element 0 SkyFromSpace
Use Light Probes ☐

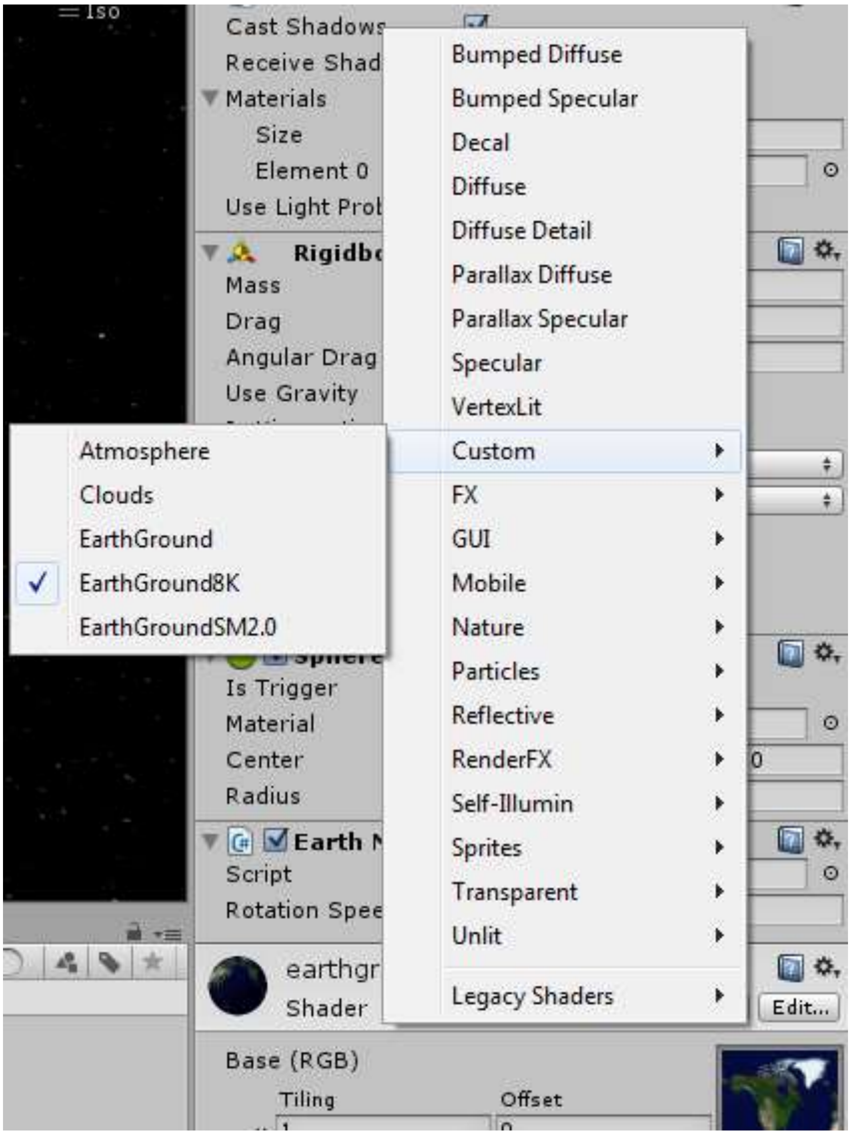
Transform
Position X 0 Y 0 Z 0
Rotation X 0 Y 0 Z 349.0447
Scale X 1.004 Y 1.004 Z 1.004
Sphere 01 (Mesh Filter)
Mesh Sphere01
☑ **Mesh Renderer**
Cast Shadows ☑
Receive Shadows ☑
Materials
Size 1
Element 0 Clouds
Use Light Probes ☐

Transform
Position X 0 Y 0 Z 0
Rotation X 0 Y 0 Z 350.286
Scale X 1 Y 1 Z 1
Sphere 01 (Mesh Filter)
Mesh Sphere01
☑ **Mesh Renderer**
Cast Shadows ☑
Receive Shadows ☑
Materials
Size 1
Element 0 CloudShadows
Use Light Probes ☐

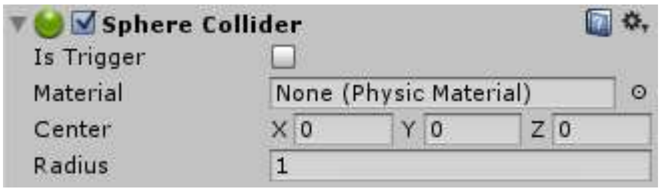
As you can notice, the “Atmo” object scale is at 1.025, it’s important for the atmospheric effect, the sphere has to be 2.5% bigger, so you have to change these values to exactly 1.025 too.

As for the cloud object, if you use the shadows, the scale also have to be a bit bigger than 1, here it’s at 1.004, but you can change it if you want the clouds to be further or closer than the default setting.

Materials should already be linked to the right shaders, but if they don't (that happens), you'll find them there:



Now, select the "Ground" object and add a sphere collider to it (Add Component > Physics > sphere collider). The default values should be good:



☑ Sphere Collider

Is Trigger □

Material None (Physic Material)

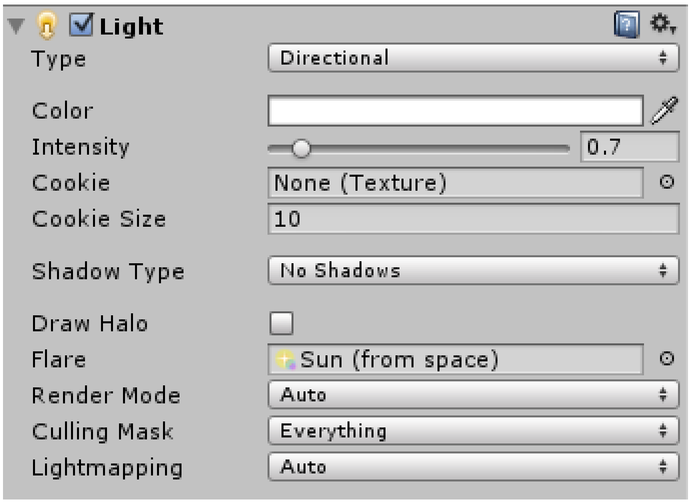
Center X 0 Y 0 Z 0

Radius 1

The purpose of this is, in addition to do add a collider, to hide the lens flare of the light when it is behind the planet.

You may also ask why not use the mesh collider using the previous high polygon sphere, it'll work but it would be a waste of resources, the default sphere seems sufficient for the most of purpose.

Finally, add a light to the Sun object (Add component > Rendering > Light), make it directional, then add a lens flare to it (one is included in "Lens Flares/" folder).



☑ Light

Type: Directional

Color:

Intensity: 0.7

Cookie: None (Texture)

Cookie Size: 10

Shadow Type: No Shadows

Draw Halo: ☐

Flare: Sun (from space)

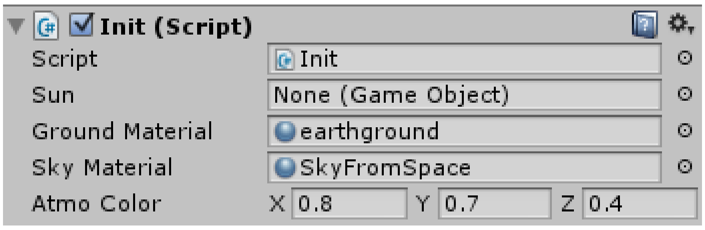
Render Mode: Auto

Culling Mask: Everything

Lightmapping: Auto

2. Outer atmospheric scattering effect

Attach the script named “Init.cs” (located in “Scripts/” folder) to the “Earth” object (Drag and drop it to the inspector, or in the inspector do Add Component > Scripts > Init. Then, fill the empty fields as follow:



☑ Init (Script)

Script: Init

Sun: None (Game Object)

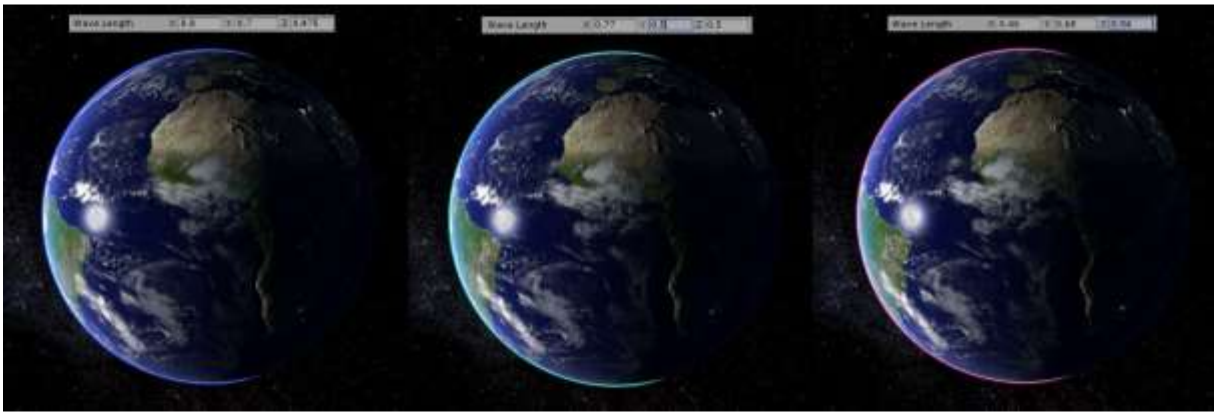
Ground Material: earthground

Sky Material: SkyFromSpace

Atmo Color: X 0.8 Y 0.7 Z 0.4

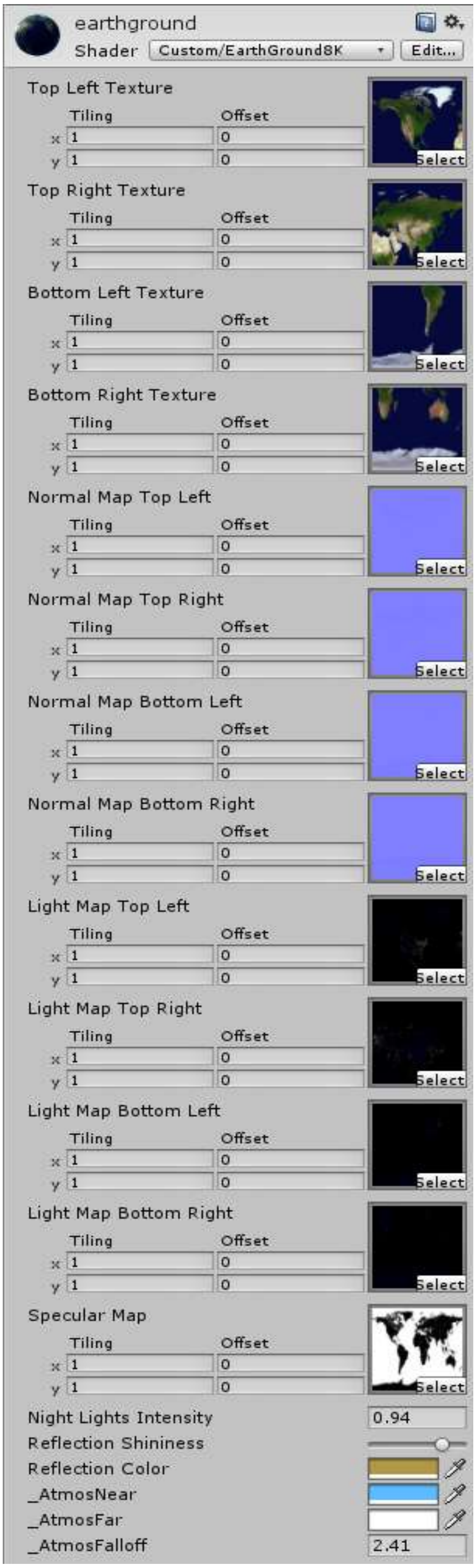
Sun: the sun object where the light is attached
Ground Material: the material that is used for your “Ground” object.
Sky Material: the material used for your “Atmo” object.
The “*Atmo Color*” parameters allow you to play with the color and aspect of the atmosphere.

Be warned, the atmospheric effect doesn’t appear in the editor view (or barely appears but looks like crap), you have to run the scene to see it. So you should play with the value while the scene is running to see effects. Examples:



3. Surface materials shaders

1. earthGround8K



earthground

Shader Custom/EarthGround8K

Top Left Texture

	Tiling	Offset
x	1	0
y	1	0

Top Right Texture

	Tiling	Offset
x	1	0
y	1	0

Bottom Left Texture

	Tiling	Offset
x	1	0
y	1	0

Bottom Right Texture

	Tiling	Offset
x	1	0
y	1	0

Normal Map Top Left

	Tiling	Offset
x	1	0
y	1	0

Normal Map Top Right

	Tiling	Offset
x	1	0
y	1	0

Normal Map Bottom Left

	Tiling	Offset
x	1	0
y	1	0

Normal Map Bottom Right

	Tiling	Offset
x	1	0
y	1	0

Light Map Top Left

	Tiling	Offset
x	1	0
y	1	0

Light Map Top Right

	Tiling	Offset
x	1	0
y	1	0

Light Map Bottom Left

	Tiling	Offset
x	1	0
y	1	0

Light Map Bottom Right

	Tiling	Offset
x	1	0
y	1	0

Specular Map

	Tiling	Offset
x	1	0
y	1	0

Night Lights Intensity 0.94

Reflection Shininess

Reflection Color

_AtmosNear

_AtmosFar

_AtmosFalloff 2.41

As you can see, most of textures are divided in 4 parts (except specular map). In this way we can have like an 8K texture on the sphere. Downward, you must attach your 4 texture parts like this:

Top Left Texture: put here the upper left part of your 8K texture
Top Right Texture: put here the upper right part of your 8K texture
Bottom Left Texture: put here the lower left part of your 8K texture
Bottom Right Texture: put here the lower right part of your 8K texture

Normal Map: the normal map textures to add a relief (bump mapping) effect. If you want to use your own height map textures (the gray ones), be careful, as Unity, when it transforms them into normal map textures (aka tangent spaced textures), adds a 1 pixel smoothing edge on each texture that is clearly visible and not wanted here. So I would suggest to prepare your own tangent spaced textures (the pink ones) that do not contain this 1px edge before importing them into Unity.

Light Map: the illuminating texture that allow the city light to be shown on the dark side of the earth.
Specular Map: the specular map texture to set the sun reflection area (water here)
Night Light Intensity: set here the intensity of the night lights.
Reflection Shininess: set here the diffusion of the sun reflection on the water. Higher = less diffusion.
Reflection Color: set here the color of the reflection. By playing with the darkness of the color, you can play with the intensity of the reflection.

Change Sun reflection's color and shininess

The next 3 parameters affect the atmospheric scattering gradient effect on the surface of the earth.

_*AtmosNear*: inner gradient color tint.
_*AtmosFar*: outer gradient color tint.
_*AtmosFalloff*: set here the thickness of the gradient. Lower value means a thinner atmosphering gradient ring.

If you want to use your own 8K texture, please note that you have to set their “wrap mode” on “clamp” (default is “repeat”)

2. earthGround8K_surfaceonly

It's the same as earthGround8K, but using the 4 texture tiling only for the surface color texture.

3. earthGround

It's the same as earthGround8K, but using a single main texture. You may want to use it if the 8K Shader is too demanding.

4. earthGroundSM2.0

This shader is here to ensure a shader model 2.0 compatibility for directX9.0 GPU, as the 2 previous shaders are compatible with DirectX 9.0c GPU only.

It's globally the same as earthGround, but without specular map (arbitrary choice, I think the specular mapping is less important as other effects).

Please also note that if a directX9.0 or lower GPU is used, you will also lose the outer atmospheric scattering effect.

4. 8k_normal_scattering

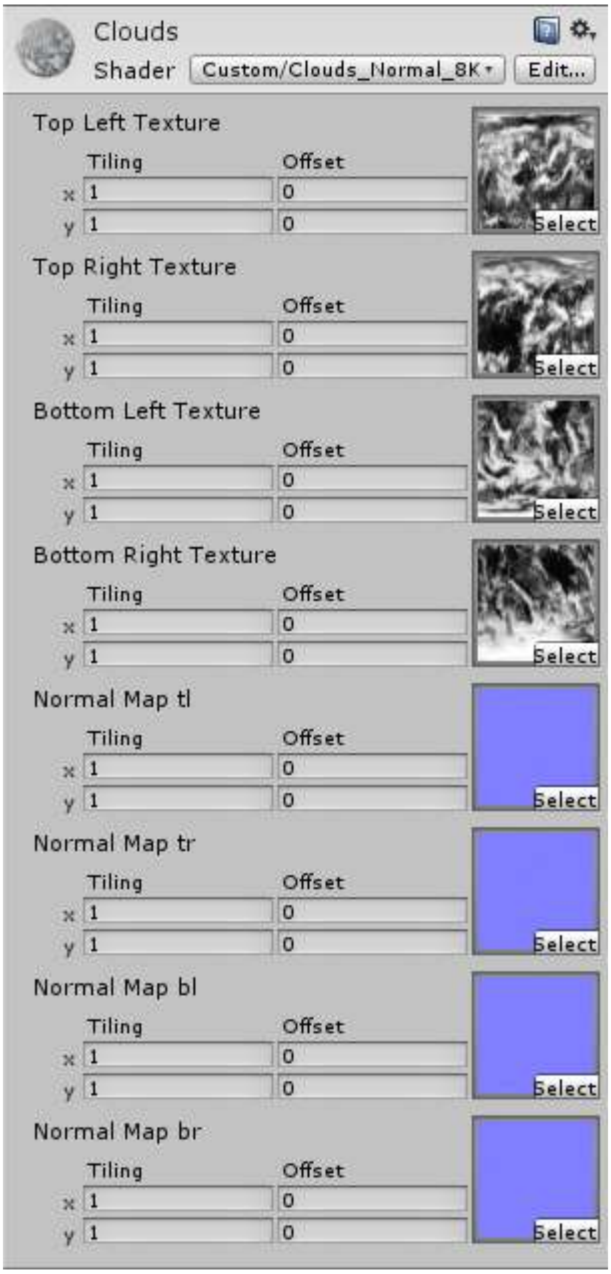
This shader used for the Mars surface, is basically a simplified version the "earthGround8K_surfaceonly" shader, since it doesn't have nightlight and specular effects.

5. 8k_normal

This shader, used for the Moon surface is even simpler as it doesn't have atmospheric scattering effect.

4. Clouds and cloud shadows material shader

1. Clouds_normals_8K



Clouds

Shader Custom/Clouds_Normal_8K ▾ Edit...

Top Left Texture

	Tiling	Offset
x	1	0
y	1	0

Select

Top Right Texture

	Tiling	Offset
x	1	0
y	1	0

Select

Bottom Left Texture

	Tiling	Offset
x	1	0
y	1	0

Select

Bottom Right Texture

	Tiling	Offset
x	1	0
y	1	0

Select

Normal Map tl

	Tiling	Offset
x	1	0
y	1	0

Select

Normal Map tr

	Tiling	Offset
x	1	0
y	1	0

Select

Normal Map bl

	Tiling	Offset
x	1	0
y	1	0

Select

Normal Map br

	Tiling	Offset
x	1	0
y	1	0

Select

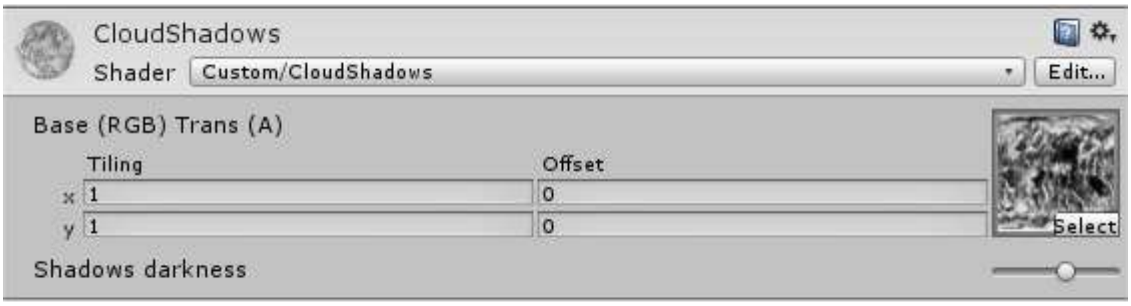
Well, just choose your cloud textures here.

The 4 textures tiling uses the same principle as the earthground8K shader.

2. Clouds

A simplified version of the cloud shader that uses a single texture.

3. Cloud Shadows



CloudShadows

Shader Custom/CloudShadows

Base (RGB) Trans (A)

Tiling x 1 y 1

Offset x 0 y 0

Shadows darkness

Here select the same texture that you use for the “real” clouds.

“Shadows darkness” setting here changes the opacity of the shadows from 0 to 1. Default is 0.5.

In both case just be sure that the texture have “alpha from grayscale” checked.

5. Demo Scripts

Except the atmospheric script Init.cs, all other script are here for a demo purpose, they aren’t necessary in your project. But still, here is a short summary of them:

- CameraZoom.cs: Allow to zoom-in and zoom-out usin the scroll wheel. Attach it to the camera.
- EarthMovement.cs: Allow any rigibody to automatically rotate. You can attach it on either “Ground” or “Clouds” objects, but be sure they do have a rigidbody attached.
- RotateEarth.cs: Allow users to rotate a planet with the right mouse button or the arrow keys. Attach it to any planet object.
- RotateSun.cs: Allow users to rotate the sun light with the left mouse button or the Q-D keys. Attach it to the “Sun” object.
- GUI.cs: This script allow to browse the different planets using buttons in the demo scene. You can see in it how to dynamically instantiate the planets.

3. Credits

Cloud and ground shaders are based on the work of Fragile Earth Studios:

http://fragileearthstudios.com/2012/07/03/code-snippets-realistic-earth-and-clouds-shader/

The atmospheric scatter shader is a part of “scrawk” work that you can find here:

http://scrawkblog.com/2013/04/13/gpu-gems-to-unity-atmospheric-scattering/

Earth textures came from NASA, have been modified to render better within the scene, but you’ll find originals here:

Earth surface : http://visibleearth.nasa.gov/view.php?id=57752

Night lights : http://visibleearth.nasa.gov/view.php?id=55167

Clouds : http://visibleearth.nasa.gov/view.php?id=57747

Normal map : http://visibleearth.nasa.gov/view.php?id=73934

Specular map : http://www.shadedrelief.com/natural3/pages/extra.html

Moon and Mars texture can be found on the Celestia Motherlode website:

Mars textures: http://www.celestiamotherlode.net/catalog/mars.php

Moon textures: http://www.celestiamotherlode.net/catalog/moon.php

Skymap textures uses a photograph from the European Southern Observatory (ESO) / S. Brunier :

http://commons.wikimedia.org/wiki/File:ESO_-_The_Milky_Way_panorama_(by).jpg