Overview

The Planet Texture Generator allows you to quickly create a wide variety of planets with little effort. Easily adjust terrain, water, clouds, city lights on the night side of your planets and add snow to your planets poles and mountain tops. When finished adjusting your planet, bake your textures at up to 8192 × 4096 resolution, which can later be edited in your favourite image editor.

Contents

1. Source code – contains all you need to create your planet textures while allowing you to make any changes to how textures are generated
2. Shaders for displaying city lights and clouds on your planet, one supports parallax mapping while the other cuts parallax mapping for better performance, both support specular highlights for water
3. Planet model
4. Example Planet Assets

Starting From Scratch

After importing the asset, either select Assets → Create → Planet Asset or right click in the project window and select Create → Planet Asset. Click on the new asset and in the inspector you'll see all settings available to adjust your planet, more information on these settings can be found below. By default, these are set to create an earth like planet.

At the bottom of the inspector, you'll see the preview button, click this to open the preview window were you can see the different images created. After adjusting any settings, click preview again to see your changes. Whenever you're ready, click bake to output your textures into the same folder as your Planet Asset.

At this point, you can create a material using either the HeyBlairGames/PTG/Planet Parallax Specular or HeyBlairGames/PTG/Planet Specular shaders and drag your textures into the correct slots. Now add the planet model into your scene and apply your material and finally add a directional light to light your planet.

Generated Textures

After your textures have been created, you'll have to select your NormalMap and CloudNormalMap and change their types from Texture to Normal map. Additionally, you may also experience texture filtering issues at the poles of your planet. To fix this, increase the Aniso Level of each texture, 9 should be fine.

1. GroundMap – contains the diffuse map in the rgb channels and the specular map in the a channel
2. NormalMap – contains normals for ground layer
3. HeightMap – optional, used for parallax effect, can be added to the a channel of the IlluminationMap
4. IlluminationMap – contains the illumination intensity in the rgb channels
5. CloudMap – contains the cloud map in the rgba channels
6. CloudNormalMap – optional, contains normals for cloud layer

Settings

Randomise All	Randomise all values
Randomise Surface Seed	Randomise seed for surface noise
Randomise Surface Noise	Randomise surface noise values
Surface Noise	Values that affect the noise generated for surface details
Randomise Land Seed	Randomise seed for land noise
Randomise Land Noise	Randomise land noise values
Land Noise	Values that affect the noise generated for land colours
Land Colour 0	First land colour
Land Colour 1	Second land colour
Land Colour 2	Third land colour
Land Colour 3	Fourth land colour
Randomise Land Colour 01 Seed	Randomise seed for first two land colours
Randomise Land Colour 01 Noise	Randomise noise values for first two land colours
Land Colour Noise 0	Values that affect the noise generated for the first two land colours
Randomise Land Colour 23 Seed	Randomise seed for second two land colours
Randomise Land Colour 23 Noise	Randomise noise values for second two land colours
Land Colour Noise 1	Values that affect the noise generated for the second two land colours
Water Colour 0	Colour of deep water
Water Colour 1	Colour of shallow water
Water level	How high the water level is, 0 – no water, 1 – all water
Water Specular	Strength of specular reflections
Water Falloff	Affects how quickly shallow water becomes deep water
Ice Colour	Colour of ice

Ice Reach	How far polar caps extend from poles
Ice Height	How high ice appears on mountain tops, 0 – everything is ice, 1 – no ice
Shadow Range	Determines at what point the difference in height between two points creates a shadow
Shadow Strength	How dark the shadow is, 0 – no shadow, 1 – completely black
Randomise City Seed	Randomise seed for city placement
City Reach	How far from the equator do cities reach
City Height	How high do cities reach, 0 – no cities, 1 – cities can be on highest mountains
City Colour	Colour of cities, this needs to be very dark to let the intensity below control how bright the cities appear
City Count	Count of starting (biggest) cities
City Multiplier	Multiplier for count of cities for next layer of cities
City Dropoff	How quickly city intensity and city spread drop with each layer
City Depth	How many layers of cities
City Spread	How big the cities are
City Intensity	How bright the cities are
Max City Intensity	Max brightness for cities
City Falloff	How quickly intensity drops from city centre
Normal Scale	Scale the normals in the NormalMap
Randomise Cloud Seed	Randomise seed for cloud noise
Randomise Cloud Noise	Randomise cloud noise values
Cloud Noise	Values that affect the noise generated for clouds
Cloud Colour 0	First cloud colour
Cloud Colour 1	Second cloud colour
Cloud Spin	Smears cloud colour horizontally, 1 is for no smear

Cloud Normal Scale	Scale the normals in the CloudNormalMap
Size	Width of generated textures, height is half of width
Bake Height Into Illumination	Add height data to alpha channel of IlluminationMap
Preview	Show preview window
Full Bake	Output all textures
Bake Ground Maps	Output ground, normal map, illumination and optionally height map textures
Bake Cloud Maps	Output cloud and cloud normal map textures

Source Files

PlanetAsset.cs	Contains all the data for the asset
PlanetComponentEditor.cs	Controls the component window view of the asset, clicking any of the bake buttons will call the PlanetTextureGenerator to create the textures and save them, clicking the preview button will also call the PlanetTextureGenerator to create preview versions of the textures and then call the PlanetPreviewWindow to display the textures
PlanetPreviewWindow.cs	Displays preview versions of the textures in an editor window
PlanetTextureGenerator.cs	Calls PlanetNoiseGenerator to create all the data needed to create the different textures and then either saves them or makes them available to the PlanetPreviewWindow
PlanetNoiseGenerator.cs	Uses noise data from PlanetNoisePerlin to create diffuse data (for ground colour, normals and height,) city data and cloud data for PlanetTextureGenerator to create textures from
PlanetNoiseGenerator.compute	A compute shader to accelerate diffuse and cloud data generation, used when available, see below for more details
PlanetNoisePerlin.cs	Creates noise data for PlanetNoiseGenerator to use

Compute Shader

The Planet Texture Generator includes a compute shader to accelerate parts of the generation process. This is used automatically when available and falls back to the standard CPU path when not. The following requirements must be met to use the compute shader:

1. Windows 7 or newer
2. DX11 or higher
3. DX11 capable GPU
4. Unity project using DX11

If you meet these requirements except for 4, it may be worthwhile to create a separate project to take advantage of compute support. Unfortunately Unity doesn't currently support compute shaders on Mac.

Depending on your graphics hardware, your graphics driver may crash while generating your textures. If this happens, you can go to PlanetNoiseGenerator.cs and change `maxThreadsPerDispatch` (line 25) to a lower value. If this doesn't work, in the same file, comment out line 6 `(#define USE_GPU)` to use your CPU instead.

Support

If you have any problems with this asset, please email the following address:

heyblairgames@gmail.com