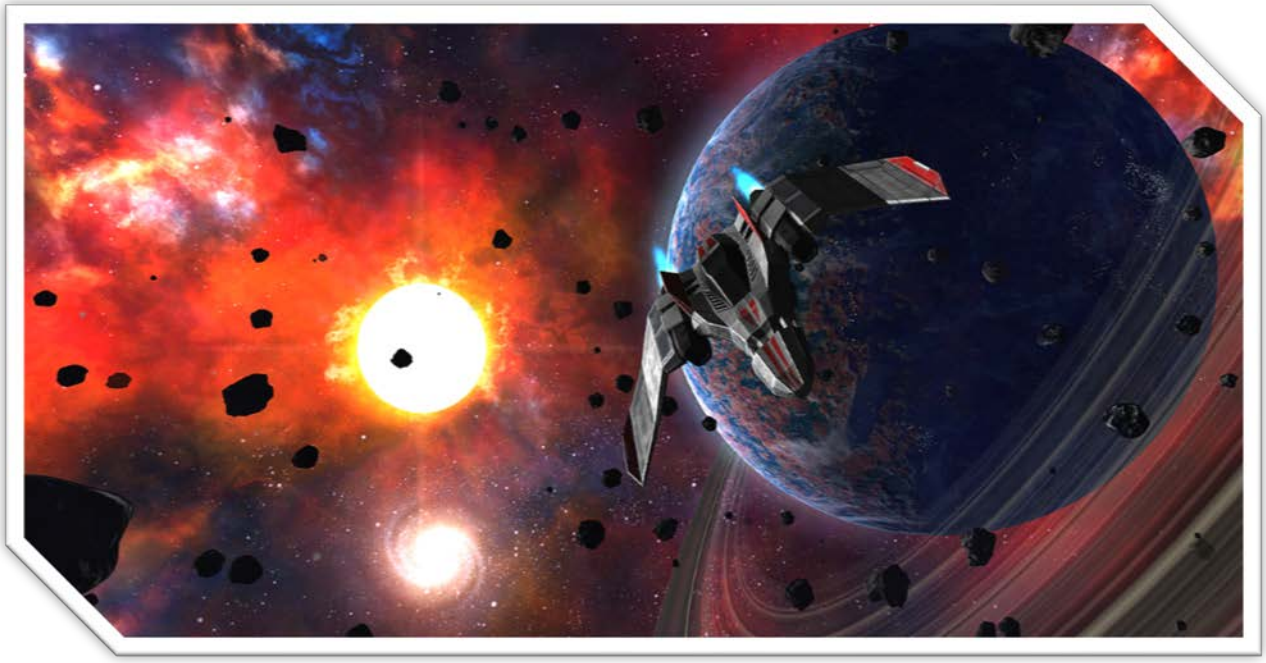
SPACE FOR UNITY

SPACE SCENE CONSTRUCTION KIT



Manual (version 1.5)

TABLE OF CONTENTS

INTRODUCTION

Thank you for purchasing *SPACE for Unity - Space Scene Construction Kit*. This Unity package enables you to quickly create custom space scenes for your games.

Traditionally, you would use a skybox to create a space background but that approach has a number of limitations which SPACE for Unity is designed to overcome.

Skyboxes use 6 fixed static textures which becomes a problem with growing distribution sizes and the need to create/buy a skybox for every single scene. SPACE for Unity uses a different approach. A space scene is generated dynamically based upon your choices within the Space Scene Construction Kit Unity 3D editor window. These space scenes consume less memory and they also offer much more flexibility compared to skyboxes. Individual textures can be smaller in size and they can also be reused in multiple scenes.

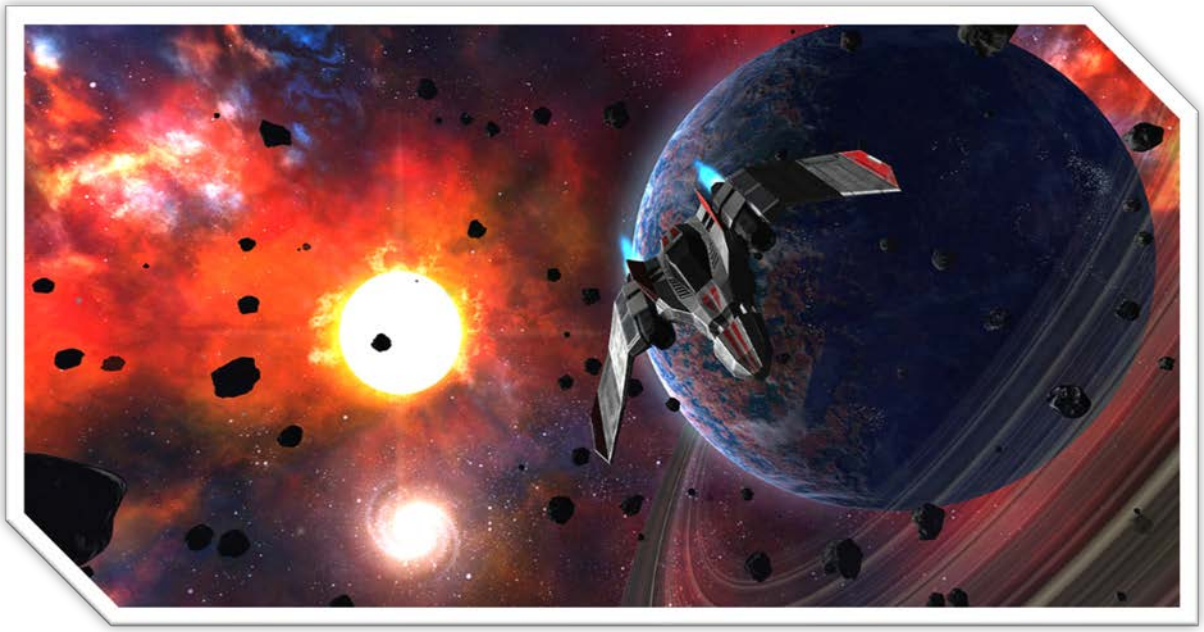


FIGURE 1 - SCREENSHOT

EXPANSION PACKS

Due to size limitations in the Unity Asset store, additional content for SPACE for Unity is provided as expansion packs.

How Does it Work?

SPACE for Unity works by adding a camera, SpaceCamera, which renders everything on Unity Layer 20 (named "DeepSpace".) Depth of SpaceCamera is set to -2 which renders *before* the Main Camera which is has a default depth of -1. Also, the Main Camera has Clear Flag set to "Depth Only" which allows everything rendered by SpaceCamera to remain as a background. The SpaceCamera monitors the orientation of the Main Camera and rotates simultaneously to keep them in sync.

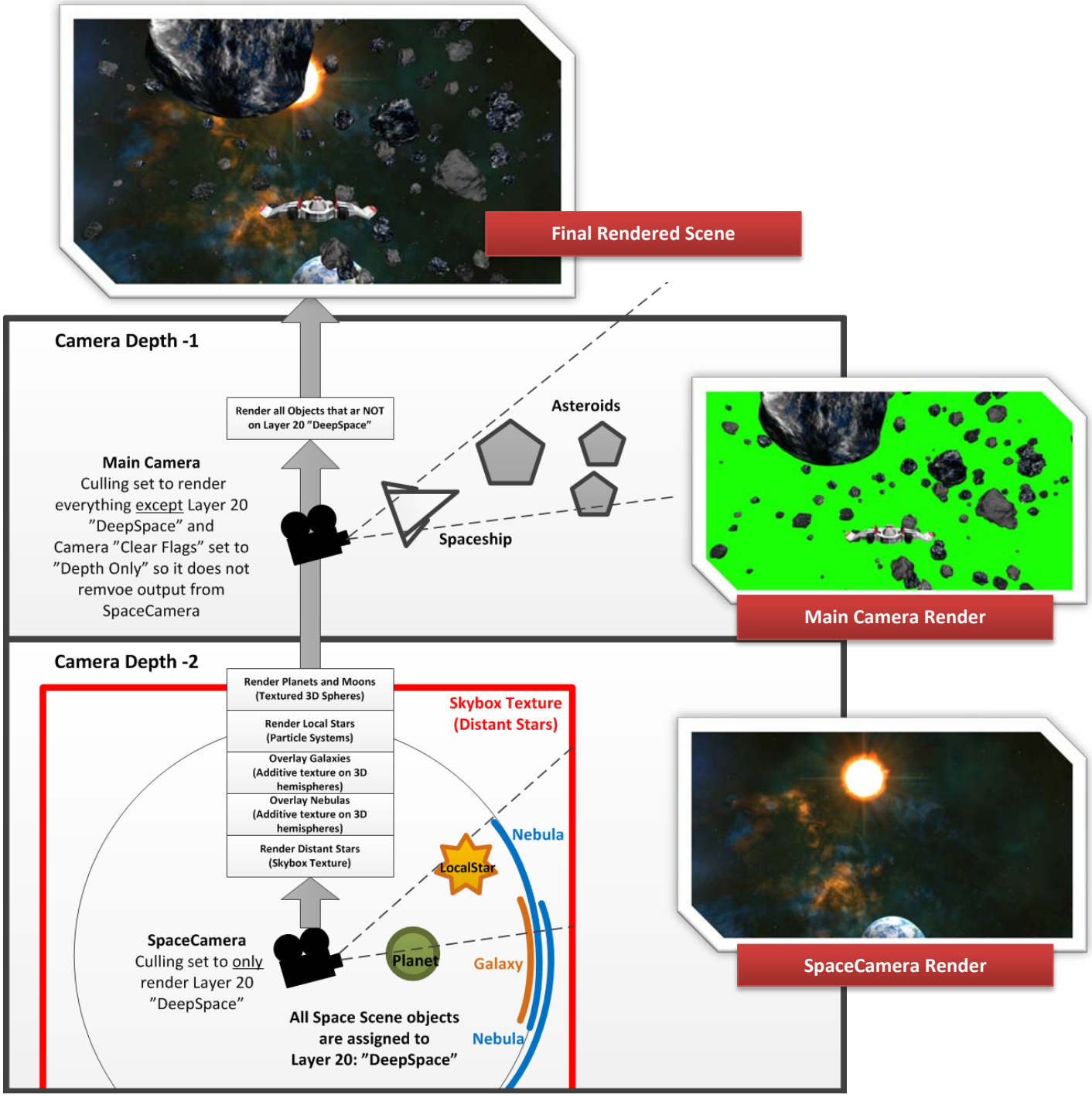


FIGURE 2 - SPACE FOR UNITY CONCEPT

UNITY EDITOR EXTENSION

Once you have imported the SPACE for Unity package in unity, you will get a new menu option under Window named “Space Scene Construction Kit” (*Figure 4 - Editor Extension Window.*)

You can customize all the filters to create a space scene in a particular mood and with different memory use properties.

The new window allows you to quickly create a space scene with millions of stars, nebulas, galaxies, planets, moons and local stars.

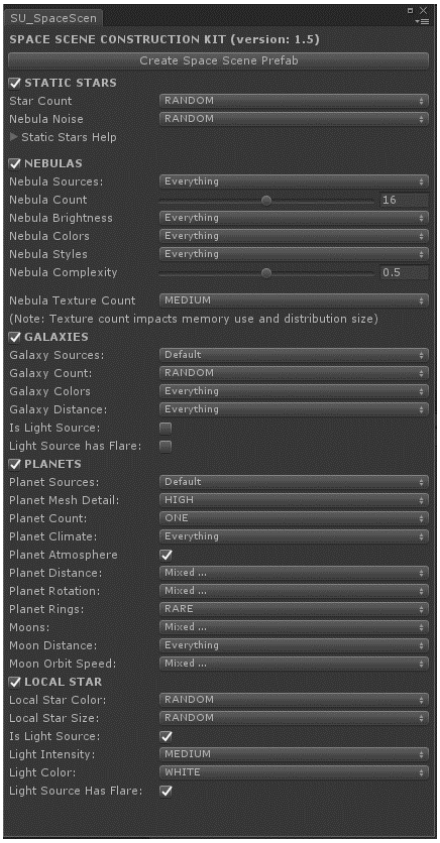


FIGURE 3 - EDITOR EXTENSION WINDOW

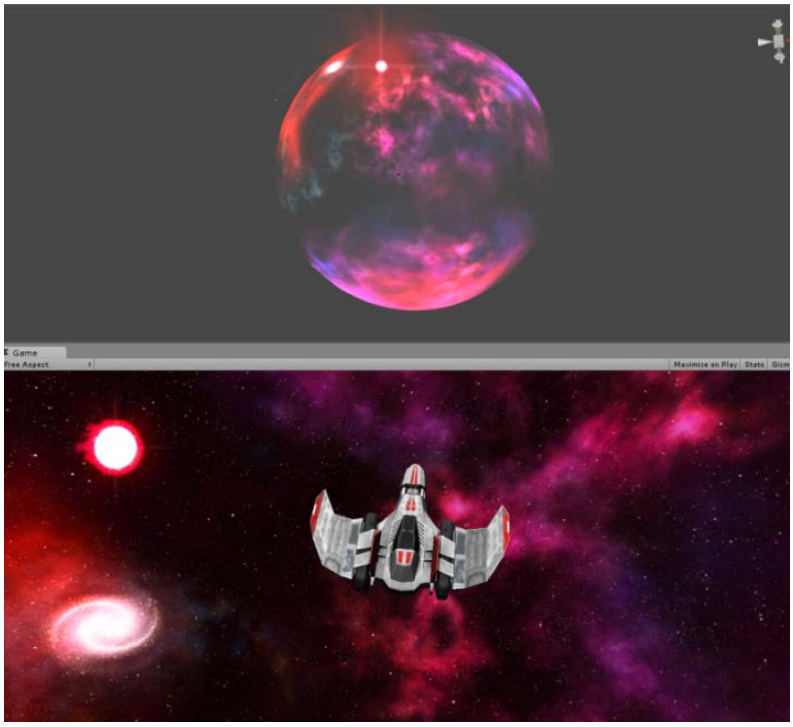


FIGURE 4 - SCENE AND GAME VIEW

SPACE SPHERE AS BACKGROUND

SPACE for Unity creates a “space sphere” (*Figure 5 - Scene and Game View*) containing stars, nebulas, galaxies, planets, moons, and other objects. The ***space sphere has its own camera that is independent from your main camera*** and the entire sphere is ***rendered as a background*** to your main camera.

Another advantage of using this approach is that you can optionally enable relative speed between your game object and the background so you move slightly within the space scene making it come alive even more.

GETTING STARTED

IMPORTING THE SPACE UNITY PACKAGE

When you purchase Space Unity form the Unity Asset store the package will be downloaded. Please allow some time for this as the high resolution graphics consume a large amount of space.

Once downloaded it is strongly **recommended** that you **import the Space Unity package into a new Unity project**. Use the new project and **dedicate it to creating space scenes** for your games that you export/import into your other projects. See for *Recommended Workflow* on page 23 for more details.

SET THE CORRECT NAME FOR USER LAYER 20

When you import packages in Unity, layer names are not imported. This is because there are a fixed number of layers in Unity (compared to labels, for example, which are included in imported packages because you can have unlimited number of labels.)

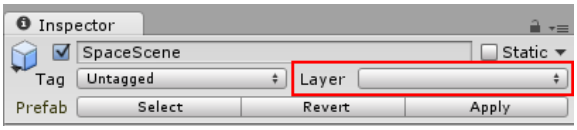


FIGURE 5 - LAYER NAME BLANK

You manually have to set the name of Layer 20 to "DeepSpace". If this step is not performed the layers of space scene objects will appear to be blank. The layer name should be configured for any project using assets from Space Unity.

CONFIGURE THE LAYER NAME

1. Click on "**Layer**" in the inspector (if you have a Space Unity object selected it may be blank (as seen in *Figure 6 - Layer Name Blank*) or if you have another asset selected it may say "Default."
2. Select "Add Layer" (*Figure 7 - Add Layer)*
3. Click on Layer 20 and name it DeepSpace (*Figure 8 - Layer 20 DeepSpace*)

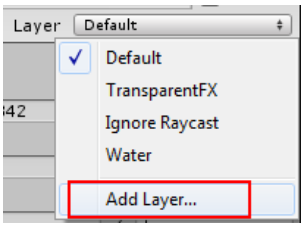
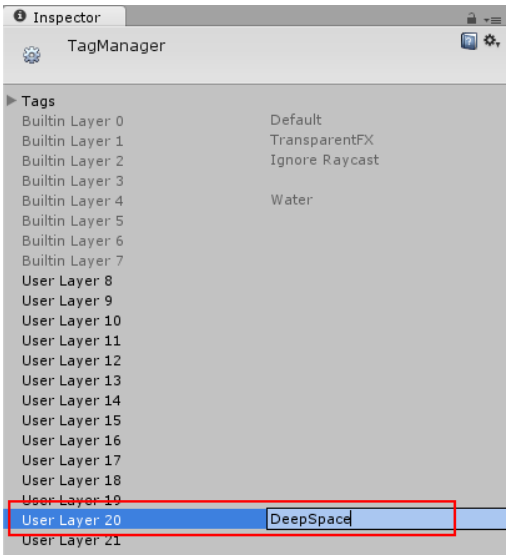


FIGURE 6 - ADD LAYER

Note 1: *Even if the name is not configured, and provided that you don't already use Layer 20 for anything else, the Space Scenes will still work. It will, however, look much more understandable if the layer is named properly.*

Note 2: *This process will have to be performed in any Unity project to which you import scenes created by Space Unity because export/import does not include layer names.*

Note 3: *If you are already using User Layer 20 in your game, you may get unwanted behavior as the Main Camera, for example, does not render object in layer 20. If this is the case, you will manually have to change the layer for either your previous content or SpaceUnity content.*



IMPORTANT NOTES

FIGURE 7 - LAYER 20 DEEPSPACE

TEXTURE RESOLUTIONS

The resolutions of Planets and Star background textures have their ”Max Size” **set to 4096x4096**. You may want to reduce the texture size, for example to **1024x1024**, to reduce size of games and to support older hardware. Pick a resolution that suits your need.

You can set the texture resolutions by selecting texture(s) in the Project Folder (e.g. SpaceUnity/Textures/Planets/Earth (Diffuse 4096) and changing “Max Size” to desired max resolution.)

Be aware: *Unity may crash if you try to select multiple high resolution textures and increasing them to 4096x4096 all at once.*

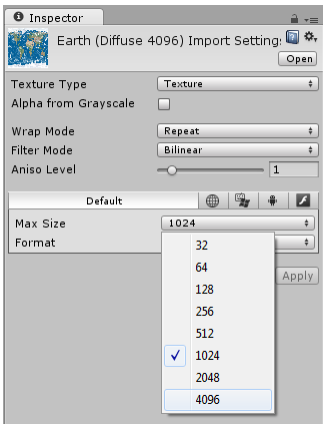


FIGURE 8 - TEXTURE MAX SIZE

LIGHTS

The main source of light in your scene should preferably be created by a Local Star. This automatically creates a “**Point Light**” which uses a Culling Mask to **only light up the space scene**, and a “**Directional Light**” which lights up **everything except the space scene**. The reason for having two lights is that the point light will provide incorrect lighting angles for your actual game objects which operate in a much smaller scale than the space scene and you would fly around the point light even if it looks like it’s far away.

The **planet shader** only supports **one main point light** and you may experience strange graphical behavior on planets when you add more point lights to your scene. To avoid this graphical behavior (flickering atmosphere / night side) you can configure your point light settings to “**Not Important**” which should then be ignored by the planet shader.

WHEN ADDING LIGHTS TO YOUR SCENE

When you add lights to your scene ensure to set **Culling Mask for the light to exclude Layer 20 “Deep Space”** - Otherwise you your planets may light up incorrectly.

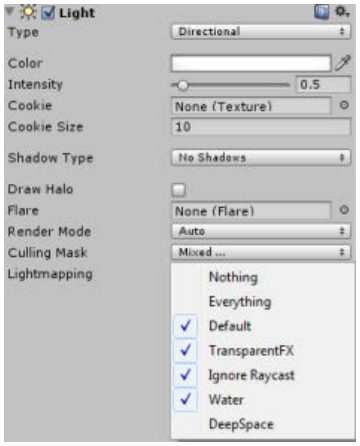


FIGURE 9 - LIGHT CULLING MASK

MOVING AND ROTATING PLANETS

It is important that when you move a planet – move the parent object for the planet, for example “PlanetHighPoly.” When you rotate a planet – it is important that you rotate the child object of the planet, “PlanetObject.” This is because the atmosphere shader only works properly if the atmosphere object has a rotation of 0,0,0.

CREATING SPACE SCENES

There are a number of ways that you can create space scenes with SPACE for Unity. The simplest method is to load the Unity Editor Extension Window and "Template" scene.

ALTERNATIVE 1 (EASY) - CREATE A NEW SPACE SCENE USING TEMPLATE SCENE

1. Load the scene: **SpaceUnity/_Scenes/Template**
2. Click menu **Window | Space Scene Construction Kit**
3. Click the button **Create Space Scene Prefab**
   (*Figure 11 - Create Space Scene Prefab*)
   Optional: Repeat step 3 step as many times as you wish until you get a space scene that you are satisfied with.
4. Go to **File | Save Scene As**…
5. Enter an appropriate name and click **Save**

FIGURE 10 - CREATE SPACE SCENE PREFAB

Once the space scene has been created, you can press play and have a look at the new scene by flying the spaceship in the template scene (use arrow keys and configured fire buttons.)

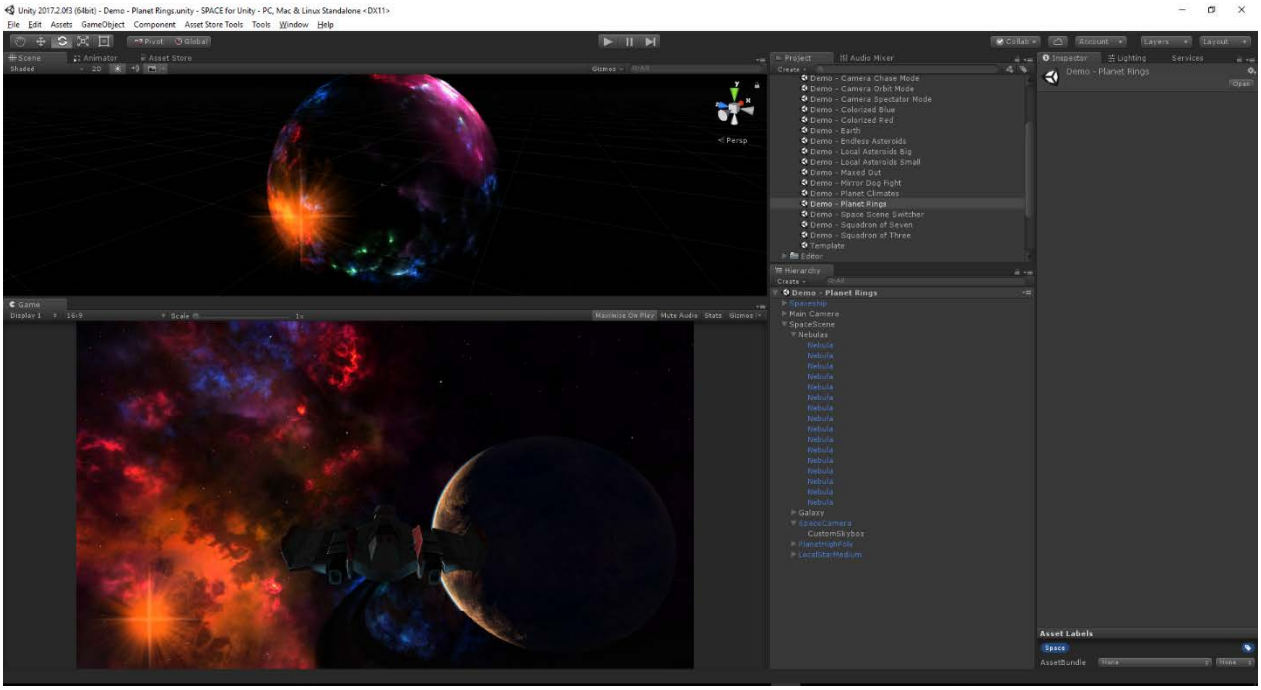


FIGURE 11 - NEWLY CREATED SPACE SCENE

Alternative 2 - Create a New Space Scene Using an Empty New Scene

1. Create a new scene **File | New Scene**

Modify Main Camera

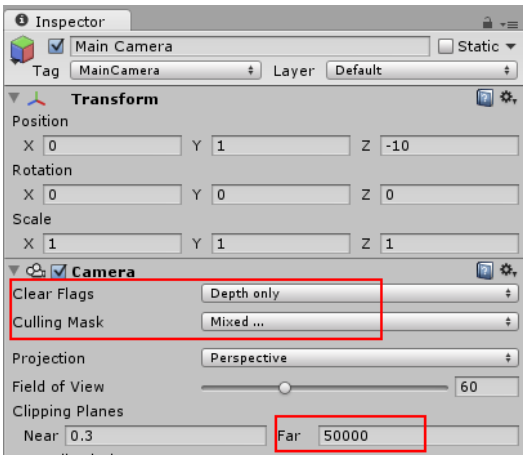


FIGURE 12 - MAIN CAMERA CONFIGURATION

2. Click on the **Main Camera** in the **Hierarchy Window** (*Figure 13 - Main Camera Configuration*)
3. Change **Clear Flags** from Skybox to **Depth Only**
4. Change **Culling Mask** and exclude **DeepSpace**
   Note: *If the "DeepSpace" layer is not visible, follow procedure: "Set the Correct Name for User Layer 20" on page 8*
5. Change the **Far Clipping Plane** from 1000 to **50000** (The purpose being that camera effects and asteroid fields, if used, are further away than 1000)

Create the Space Scene

6. Go to **Window | Space Scene Construction Kit**
7. Click the button **Create Space Scene Prefab** (*Figure 11 - Create Space Scene Prefab*)

Optional - Adding the Spaceship

8. Drag the prefab **SpaceUnity/_Demo/Prefabs/Spaceship** to the **Hierarchy**

Optional - Configure Camera Follow Script

9. Drag the script **SpaceUnity/_Demo/Scripts/SU_CameraFollow** to the **Main Camera**
10. Click on Main Camera in the Hierarchy Window
11. Drag the **Spaceship** object from your hierarchy to "**Target**" value of the Camera Follow script

Optional - Adding Space Particles and Space Fog

12. Drag the prefab **SpaceUnity/Prefabs/CameraEffects/SpaceFog** from the Project Window onto the **Main Camera** in the Hierarchy Window making it a child of the main camera
13. Drag the prefab **SpaceUnity/Prefabs/CameraEffects/SpaceParticles** from the Project Window onto the **Main Camera** in the Hierarchy Window making it a child of the main camera

Optional - Adding Endless Asteroid Field

14. Drag the prefab **SpaceUnity/Prefabs/Objects/AsteroidField** onto the **Spaceship** in the Hierarchy Window making it a child to the Spaceship

Once the space scene has been created, you can press play and have a look at the new scene by flying the spaceship in the template scene (use arrow keys and configure fire buttons.)

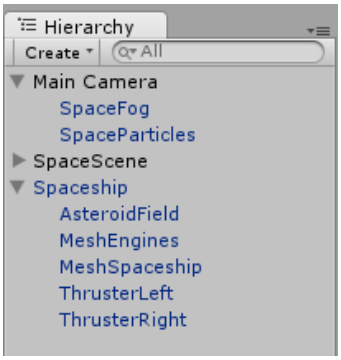


FIGURE 13 - CAMERA EFFECTS AND ASTEROIDS

USING CUSTOM FILTERS

When you create a Space Scene you can specity custom filters for the Space Scene Construction Kit editor extension to consider when randomly selecting assets. Default options in the filter tables are marked in **Bold**.

FILTER OPTIONS – STATIC STARS

Distant stars are created using a single skybox texture. The textures have been labeled with "star count". As you change the star count and nebula noise the editor script will only select skybox textures with the defined star count and nebula noise color.

FIGURE 14 - STARS FILTER

TABLE 1 - FILTER OPTIONS - STARS

Filter	Options	Selection Type	Description
Star Count	**Random** / Low / Medium / High	Single Value	Skybox textures are categorized by the number of stars they contain.
Nebula Noise	**Random** / Black / Blue / Orange / Green / Red / Purple / Gray / Cyan	Single Value	An additional texture can be selected that contains nebula noise that tiles in any direction. The nebula noise can be tinted to a color (also customizable).

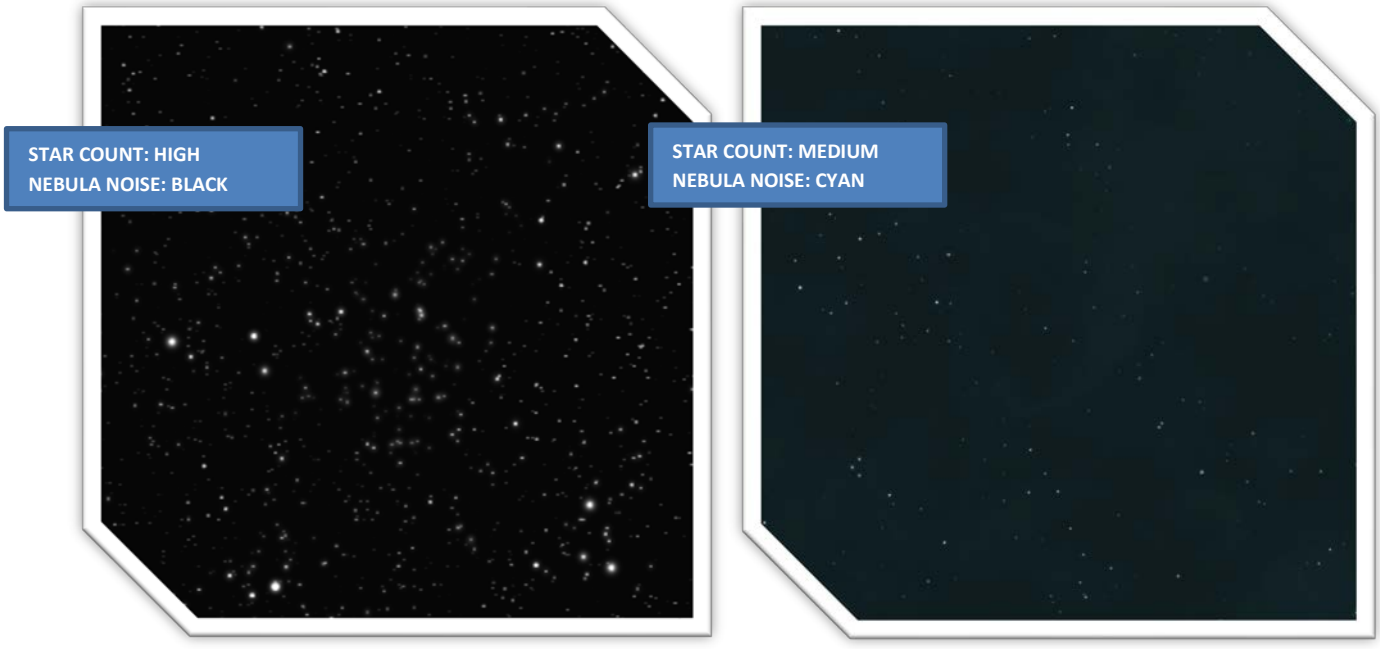


FIGURE 15 - STARS EXAMPLES

FILTER OPTIONS – NEBULAS

Nebulas are created by mapping textures onto a hemisphere mesh (like a satellite dish =) which can be rotated in any direction around three axis.

Nebula filters are bitmasks which means that you can select multiple filter options, e.g. only choose nebulas that are "Very Dark" AND/OR "Dark."

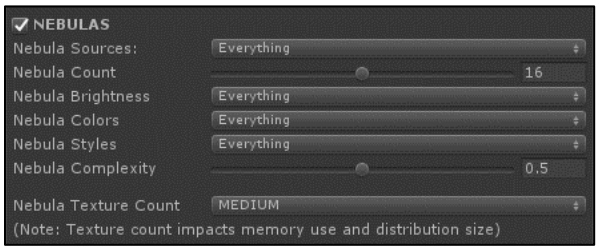
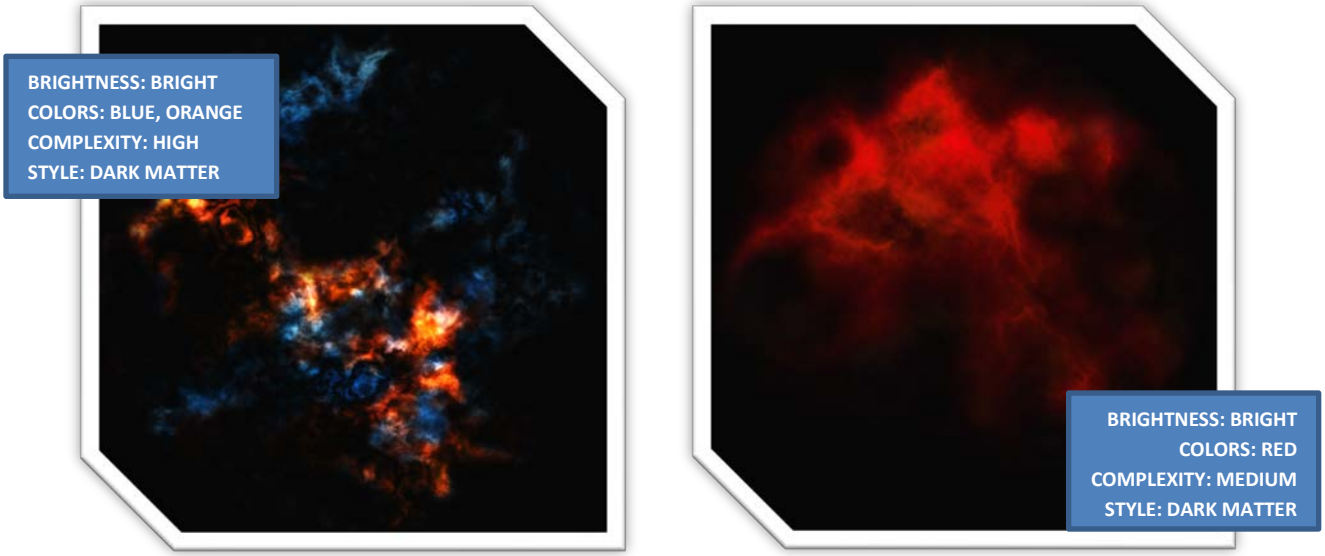


FIGURE 16 - NEBULA FILTERS

TABLE 2 - FILTER OPTIONS - NEBULAS

Filter	Options	Selection Type	Description
Nebula Sources	Subfolders of the Nebula folder	Dropdown (Multiple values)	When new nebulas are added, or if custom nebulas are created, they can be put in a separate subfolder under the SpaceUnity/Materials/Nebulas folder and the folder can be used as a filter option here.
Nebula Count	0 – 32 **(16)**	(int) Slider	Number of nebulas to create in the scene. This is the number of textured hemispheres to be created. Each hemisphere has a 224 triangles polygon with a single additive texture map.
Nebula Brightness	**Very Dark** / **Dark** / **Medium** / **Bright** / **Very Bright**	Dropdown (Multiple Values)	Only nebula materials labeled with your brightness selection will be used when creating the space scene.
Nebula Colors	**Blue** / **Pink** / **Purple** / **Green** / **Yellow** / **Orange** / **Red**	Dropdown (Multiple Values)	Only nebula materials labeled with your color selection will be used when creating the space scene. If you select Blue and Pink, textures that are labeled with additional colors that are NOT Blue and Pink will not be included.
Nebula Styles	**Cloudy** / **Streaky** / **Glittery** / **Dark Matter**	Dropdown (Multiple Values)	Only nebula materials labeled with your selection will be used when creating the space scene.
Nebula Complexity	0.0 Low → **0.5** → 1.0 High	(float) Slider 0.0 (Low) → 1.0 (High)	Use slider to set the distribution of graphic complexity from low (smooth) to high (detailed.) At 0.0, only low complexity will be used. At 0.5 an even mix between Low/Medium/High complexities will be used. At 1.0 only high complexity nebulas will be used.
Nebula Texture Count	Very Low / Low / **Medium** / High / Very High	Dropdown (Single Value)	Very Low = Maximum of 2 unique materials Low = Maximum of 4 unique materials Medium = Maximum of 8 unique materials High = Maximum of 16 unique materials Very High = Maximum of 32 unique materials (Multiply texture resolution (1024x1024 default) by number of unique materials to get video memory usage)

FIGURE 17 - NEBULA EXAMPLES



FILTER OPTIONS – GALAXIES

Galaxies are created by mapping textures onto a hemisphere mesh which can be rotated in any direction around three axis.

Galaxies can also be lightsources in the scene with an optional flare effect.

FIGURE 18 - GALAXY FILTERS

TABLE 3 - FILTER OPTIONS - GALAXIES

Filter	Options	Selection Type	Description
Galaxy Source	Subfolders of the Galaxy folder	Dropdown (Multiple values)	When new galaxies are added, or if custom galaxies are created, they can be put in a separate subfolder under the SpaceUnity/Materials/Galaxies folder and the folder can be used as a filter option here.
Galaxy Count	**Random** / None / One / Two / Three	Dropdown (Single Value)	Number of galaxies to create in the scene. This is the number of textured hemispheres to be created. Each hemisphere has a 160 triangles polygon with a single additive texture map.
Galaxy Colors	**White** / **Blue** / **Yellow** / **Purple** / **Green** / **Orange**	Dropdown (Multiple Values)	Only galaxy materials labeled with your color selection will be used when creating the space scene.
Galaxy Distance	**Very Close** / **Close** / **Distant** / **Very Distant**	Dropdown (Multiple Values)	The selected distances will determine the randomly selected size/scale of the galaxy.
Galaxy Is Light Source	True / **False**	Toggle	Whether or not a point light with infinite range should be created at the galaxy position.
Galaxy Light Has Flare	True / **False**	Toggle	If Galaxy has a light source, this selection determines whether or not a lens flare should be used for the light.

FIGURE 19 - GALAXY EXAMPLES

FILTER OPTIONS – PLANETS

Planets are spheres objects mapped with materials containing Diffuse Map, Bump Map and optionally also Night Lights and Illumination maps. You have control over the amount of planets, how far away they should be located, and the climates of the planets. Planets also have volumetric atmospheres.

Planets may have planetary rings (separate textured mesh) and moons (separate sphere mesh) depending on your selections.

FIGURE 20 - PLANET FILTERS

TABLE 4 - FILTER OPTIONS - PLANETS

Filter	Options	Selection Type	Description
Planet Source	Subfolders of the Planets folder	Dropdown (Multiple values)	When new planets are added they can be put in a separate subfolder under the SpaceUnity/Materials/Planets folder and the folder can be used as a filter option here.
Planet Mesh Detail	Low / Medium / **High**	Dropdown (Single Value)	Mesh quality of the planet. See triangle count for the meshes in the content details section (REF_XXX). High detail is only advisable for very close planets and high quality games – otherwise, save polygons by selecting a lower mesh quality.
Planet Count	Random / **None** / **One** / Two / Three / Four / Five	Dropdown (Single Value)	Number of planets to be created in the scene.
Planet Distance	**Very Close** / **Close** / **Distant** / Very Distant	Dropdown (Multiple Values)	The selected distances will determine the randomly selected distance of the planets. (There can only be one planet that is Very Close, and one planet that is Close.)
Planet Climate	**Earth Like** / **Ice** / **Desert** / **Gas** / **Molten** / **Alien**	Dropdown (Multiple Values)	Planets are categorized with different climates. Your selection will determine which random climates are created for the planets in the scene.
Planet Atmosphere	**True** / False	Toggle	Whether or not planets should have atmospheres. Atmosphere is a separate object rendered with a specific atmosphere shader.
Planet Rotation	**None** / **Slow** / Medium / Fast	Dropdown (Multiple Values)	Random rotational speed around the planet axis (Note: Anything other than None or Slow may look unrealistic)
Moons	**None** / **One** / Two	Dropdown (Multiple Values)	Number of moons to randomly add to planets.
Moon Distance	**Very Close** / **Close** / **Distant** / **Very Distant**	Dropdown (Multiple Values)	Distance from planet to randomly place moons.
Moon Orbit Speed	**Stationary** / **Slow** / Medium / Fast	Dropdown (Multiple Values)	Orbit speed of moon around planet (Note: Anything other than Stationary or Slow may look unrealistic)
Planet Rings	Never / Very Rare / **Rare** / Coin Flip / Common / Very Common / Always	Dropdown (Single Value)	Odds that planet rings should be created for planets in a scene.

FIGURE 21 - PLANET EXAMPLES

Filter Options – Local Stars

Local stars are created using Shuriken Particle Effects and also a point light which most commonly acts as the main light source in a scene.

The script will only create one local star at most (but you can manually add as many stars as you want.)

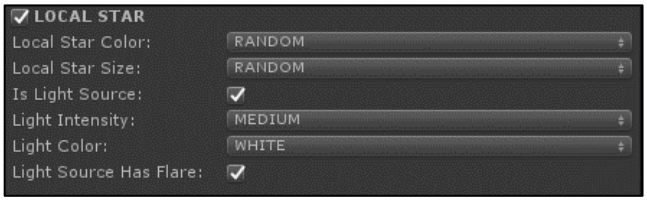


FIGURE 22 - LOCAL STAR FILTERS

TABLE 5 - FILTER OPTIONS - LOCAL STARS

Filter	Options	Selection Type	Description
Color	**Random** / Yellow / Red / Orange / Blue	Dropdown (Single Value)	Color of the local star to be created.
Size	**Random** / Small / Medium / Large	Dropdown (Single Value)	Particle Systems, and in particular light flares, don't scale very well in Unity so three fixed sizes for local stars can be created.
is Light Source	**True** / False	Toggle	Whether or not a point light with infinite range should be created at the position of the local star (this Is usually true and the star is the main light source in the scene)
Light has Flare	**True** / False	Toggle	Whether or not the light should have a flare. Visually it is strongly recommended to use the flare as it adds substantially to the look of the star.
Light Intensity	Random / Very Low / Low / **Medium** / High / Very High	Dropdown (Single Value)	Light intensity of the local star. See light intensities in the content details section (REF_XXX.)

FIGURE 23 - LOCAL STAR EXAMPLES

CREATING SPECIFIC SPACE SCENE ELEMENTS

You can choose to select only specific elements to be created. By enabling or disabling the toggle box for each element category (Stars, Nebulas, Galaxies, Planets, and Local Stars) as seen in *Figure 25 - Create Specific Elements* only the enabled elements will be created.

When a category has its toggle box disabled, the script will not create or remove any objects of that category.

When a category has its toggle box enabled, the script will search the current scene for objects with the category specific Unity tag and remove the object and all its children.

TABLE 6 - SPACE SCENE ELEMENT TAGS

GameObject	Unity Tag	Comments
Nebula	SpaceScene_Nebula	
Galaxy	SpaceScene_Galaxy	
Planet	SpaceScene_Planet	Moons and planetary rings are children of planets. They will get removed if a planet is removed.
Local Star	SpaceScene_LocalStar	
Stars	SpaceScene_Camera	Distant stars are skybox materials on the SpaceScene camera so the space scene camera is tagged. The script will detect if a material is assigned to the skybox component and warn if it is about to be overwritten.

Tip: *If you are happy with a scene but you just want to change the nebulas; uncheck all categories apart from Nebulas and click "Create Space Scene Prefab" repeatedly until you get a nebula layout you are happy with.*

FIGURE 24 - CREATE SPECIFIC ELEMENTS

OVERWRITING SPACE SCENE ELEMENTS

When a Space Scene already exists in the current scene and you click the "**Crete Space Scene Prefab**" button, you will be notified that the components you have selected will be overwritten (*Figure 26 - Overwrite Warning*).

If you choose "Ok – Overwrite", all elements of that type will be removed, and new elements will be added by the editor. For example, your scene contains stars, nebulas, galaxies, planets and moons and you select to only create new planet(s) – all existing planets (and moons since they are children of planets!) will first be removed before new planet(s) are created.

If you choose "Cancel" – the warning will disappear, and no changes are made to your Space Scene.

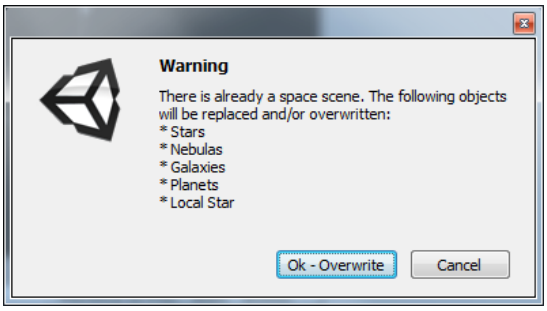


FIGURE 25 - OVERWRITE WARNING

MANUALLY MODIFYING SPACE SCENE ELEMENTS

The space scene consists of instantiated prefabs which can manually be modified once created. You may want to remove or duplicate elements, exchange some materials, or reposition elements in the scene.

Expand the "**SpaceScene**" game object in the **Hierarchy** to reveal the Space Scene elements (*Figure 27 - Manual Modification Of Space Scene Elements*) such as nebulas, galaxies, planets, moons and local stars. Click to select the elements and modify them as necessary using the **Inspector**. Most likely your modifications may entail rotating nebulas and galaxies, repositioning planets and stars, replacing materials for planets, nebulas, and galaxies, etc.

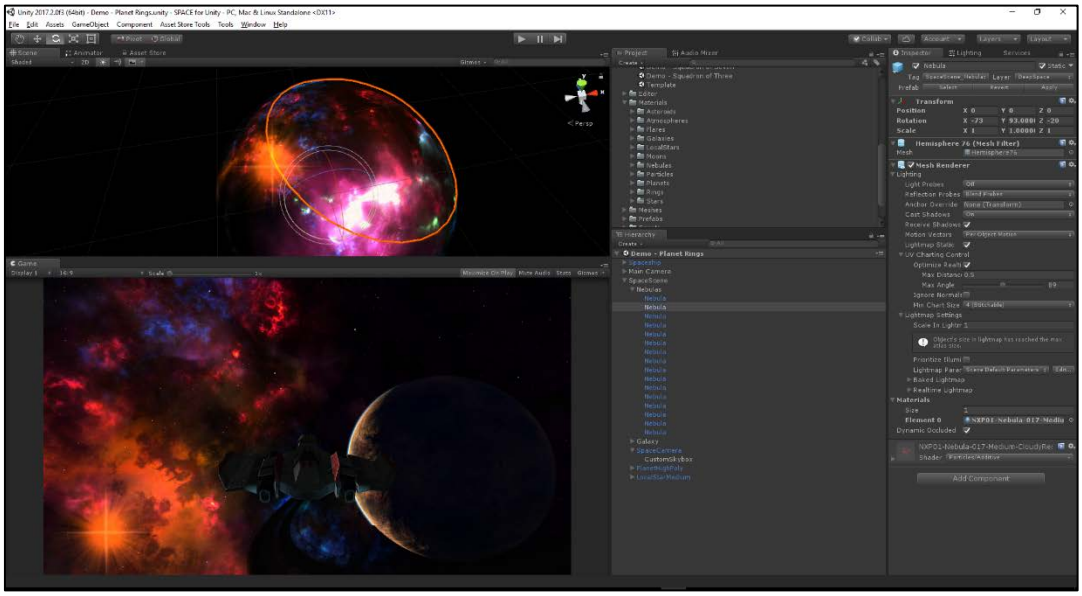


FIGURE 26 - MANUAL MODIFICATION OF SPACE SCENE ELEMENTS

Caution: *It is not recommended to rescale nebulas and local stars. Nebulas will move further away if rescaled since the pivot point of the nebula mesh is at the center of the imagined sphere that the nebula hemisphere is a part of. By trying to rescale the nebula it would move further away resulting in no visual difference to the spectator. Local stars depend on particle systems and lens flares which do not scale very well.*

MANUALLY CREATING SPACE SCENE ELEMENTS

Space Scene element prefabs located in "**SpaceUnity/Prefabs/SpaceSceneElements**" can be manually dragged from the Project structure to the Hierarchy or scene view to create elements.

Once a prefab has been added to the scene it can be repositioned and, if applicable, materials can be replaced for nebulas, galaxies, planets, and moons. Add as many or as few object as you desire and customize the look of a scene to suit your specific needs by manually building your scene.

MEMORY AND PERFORMANCE CONSIDERATIONS

VIDEO MEMORY USAGE

The amount of video memory your space scene will consume depends on the number of elements and materials you choose to have in your scene.

Memory Usage has been determined through verifying the Stats window (*Figure 28 - Memory Usage*) for each type of Space Scene elements at various texture resolutions. If you need to reduce the resolution of a texture, see *Texture Resolutions* on page 9.

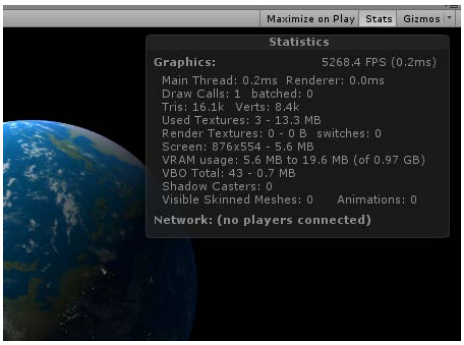


FIGURE 27 - MEMORY USAGE

TABLE 7 - VIDEO MEMORY USAGE

Space Scene Elements	Number of Textures for Material	Texture Resolutions	Texture Memory
Static Stars	1 Diffuse for Skybox – one texture for all 6 skybox sides (Double amount is used if nebula noise is enabled)	**4096x4096 (Default & Max)** 2048x2048 1024x1024 512x512 256x256 128x128	**64.0 MB** 16.0 MB 4.0 MB 1.0 MB 0.25 MB 0.08 MB
Nebula	1 (Additive)	**1024x1024 (Default & Max)** 512x512 256x256 128x128	**4.0 MB** 1.0 MB 0.25 MB 0.08 MB
Galaxy	1 (Additive)	**1024x1024 (Default & Max)** 512x512 256x256 128x128	**4.0 MB** 1.0 MB 0.25 MB 0.08 MB
Planets **(all planets)**	2 (Diffuse + Normals)	**4096x4096 (Default. Max)** 2048x2048 1024x1024 512x512 256x256 128x128	**112.36 MB** 37.2 MB 9.3 MB 2.3 MB 0.6 MB 0.15 MB
Planets **(additional memory for night city lights or illumination maps)**	1 (City Lights / Illumination)	**4096x4096 (Default. Max)** 2048x2048 1024x1024 512x512 256x256 128x128	**64.0 MB** 16.0 MB 4.0 MB 1.0 MB 0.25 MB 0.08 MB
Moons	2 (Diffuse + Normals)	1024x1024 (Max) **512x512 (Default)** 256x256 128x128	9.3 MB **2.3 MB** 0.6 MB 0.15 MB
Planet Rings	1 (Transparent)	**512x512 (Default & Max)** 256x256 128x128	**1.0 MB** 0.25 MB 0.08 MB
Local Stars	4 (Additive Particles)	**128x128 + 512x512 + 512x512 + 1024x1024 (flare)** **(Default & Max)**	**6.4 MB**

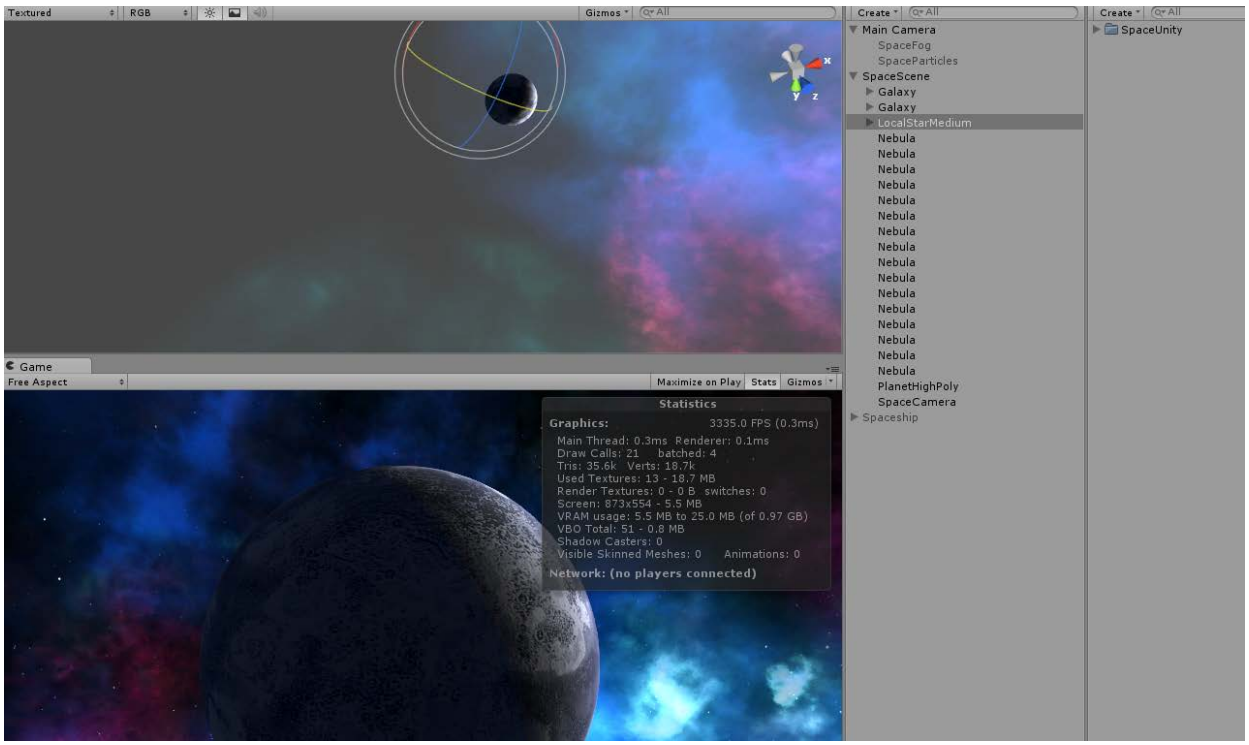
Note: *Since the space camera is in the center of the sphere, all textures will not be visible simultaneously. Less actual memory will be used during game play due to Unity's ability to manage VRAM (Video Random Access Memory.)*

TEXTURE MEMORY USAGE FOR A SPACE SCENE WITH DEFAULT SETTINGS

Using the default settings when creating a Space Scene (excluding SpaceParticles, SpaceFog, and Asteroids) varies depending on how many elements that are randomly generated.

A Space Scene was created using the default settings: *Figure 29 - Default Settings Space Scene*

FIGURE 28 - DEFAULT SETTINGS SPACE SCENE



ACTUAL MEMORY VS. CALCULATED (THEORETICAL) MEMORY USAGE

To demonstrate the **dynamic memory management** in Unity, note that with the planet in Unity reports 13 textures used with a total size of **18.7 MB** (*Figure 29 - Default Settings Space Scene*) , yet the theoretical memory calculated for the scene in the table below is 59.6 MB.

TABLE 8 - ACTUAL MEMORY VS CALCULATED MEMORY USAGE

Space Scene Element	Comments	Number of Textures	Resolution	Memory
Stars	Skybox using 1 texture	1	1024x1024	4 MB
Galaxies	Randomly, 2 galaxies were created	2	1024x1024	8 MB
Local Star Medium	Contains several particle textures at various resolutions	4	128x128 (Corona) 512x512 (Disc) 512x512 (Prominence) 1024x1024 Cropped (Flare)	6.3 MB
Nebulas	16 nebulas were created using “Nebula Texture Count” “Medium” = 8 unique textures	16	1024x1024	32 MB
Planet	Ice planet with Diffuse Map + Bump Map	2	1024x1024	9.3 MB
			Total Theoretical (not actual) Texture Memory Usage:	**59.6 MB**
			Actual Texture Memory Usage:	**18.7 MB**

DISTRIBUTION SIZE

The distribution size depends on how many space elements and textures *all* your space scenes in your game uses. If you make clever use of reusing Stars skybox backgrounds, and also try to reuse a number of nebula and galaxy textures in multiple scenes, you can reduce the size of your distribution yet still having very different looking space scenes.

This is one of the main advantages of using SPACE for Unity compared to static skyboxes which would always occupy the same amount of space in a distribution for each static skybox since no elements can be reused.

The distribution size depends on which compression settings you use for textures. The general rule of thumb to keep your distribution size down is, however;

- Use as few materials as you need
- Use the best compromise between details / performance / size for your target audience

There is no "magic bullet" here to get the maximum visual quality, best performance, and smallest distribution size. It's all about finding the best compromise.

TABLE 9 - DISTRIBUTION SIZE

Space Scene Elements	Compression	Texture Resolutions	Compressed Texture Sizes
Stars	DXT1 Compression (Diffuse for Skybox – one texture for all 6 skybox sides)	**4096x4096 (Default & Max)** 2048x2048 1024x1024 512x512 256x256 128x128	**11.2 MB** 2.8 MB 0.7 MB 0.17 MB 0.04 MB 0.01 MB
Nebula	DXT1 Compression	**1024x1024 (Default & Max)** 512x512 256x256 128x128	**0.7 MB** 0.17 MB 0.04 MB 0.01 MB
Galaxy	DXT1 Compression	**1024x1024 (Default & Max)** 512x512 256x256 128x128	**0.7 MB** 0.17 MB 0.04 MB 0.01 MB
Planets **(all planets)**	Diffuse DXT1 Compression Normal DXTnm Compression	**4096x4096 (Default & Max)** 2048x2048 1024x1024 512x512 256x256 128x128	**32.0 MB** 8.0 MB 2.0 MB 0.5 MB 0.13 MB 0.03 MB
Planets **(additional memory for night city lights or illumination maps)**	City Lights DXT1 Compression / Illumination DXT5 Compression	**4096x4096 (Default & Max)** 2048x2048 1024x1024 512x512 256x256 128x128	**11.2 MB / 20.8 MB** 2.8 MB / 5.2 MB 0.7 MB / 1.3 MB 0.17 MB / 0.33 MB 0.04 MB / 0.08 MB 0.01 MB / 0.02 MB
Moons	Diffuse DXT1 Compression Normal DXTnm Compression	1024x1024 (Max) **512x512 (Default)** 256x256 128x128	2.0 MB **0.5 MB** 0.13 MB 0.03 MB
Planet Rings	ARGB 32 BIT	**512x2 (Default & Max)** 256x2 128x2	**6.0 KB** very small =) very small =)
Local Stars	2xDXT1 + 2xDXT5 (Additive Particles)	**128x128 + 512x512 + 512x512 + 1024x1024 (Default & Max)**	**1.2 MB**

Note: *Sizes will vary on different platforms. The best way to determine the size of your distribution is to compile it and have a look at the actual file sizes.*

PERFORMANCE

Performance differs greatly between hardware which makes it difficult to predict the performance impact. Modern computers and consoles should have no problems running the Space Scenes in highest quality. Older computers may struggle if the graphic cards have poor support for additive textures (which all the nebulas, local stars, and galaxies build upon.)

Mobile platforms will not render Space Scenes as fast as computers, especially with high quality settings and resolutions. More testing will be performed and shaders and alternative mobile materials will be designed in future updates of SPACE for Unity.

Make sure you test the performance of the Space Scenes running with your additional game logic on a range of devices for your target platform and audience. If you find that the performance is great, consider adding more detail though additional elements and increased resolutions. If you find that the performance is poor, reduce the number of elements, lower resolution and consider removing bump maps and illumination maps from planets.

IMPROVING PERFORMANCE

Tips to improve performance if necessary:

- Use fewer space scene elements
- Reduce resolution for space scene materials
- Remove bump/normal and illumination maps from planets
- Re-use the same materials for nebulas and galaxies within a scene (nebulas using the same materials will be batch rendered)
- Make sure galaxies do not have light sources, only use one main light source
- Avoid planets with night illumination and night city lights

IMPROVE PERFORMANCE WITH CAMERA EFFECTS AND SPACE OBJECTS

There are also objects within Space Unity that are not part of the Space Scene that impact performance is used. Camera Effects and Objects (such as Asteroid fields) will likely require more rendering time than the entire Space Scene itself.

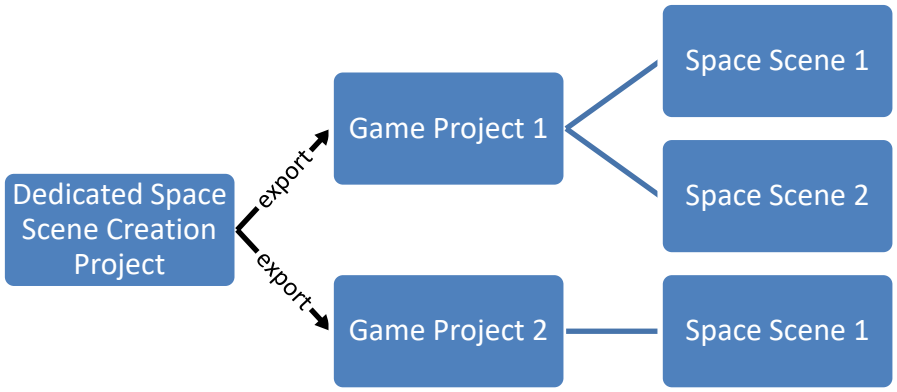
- **If you use SpaceParticles Prefab (More details, see *SpaceParticles* on page 25)**
  - Reduce the number of particles
  - Disable Fade Particles
  - Consider removing space particles all together for poorly performing devices (since it is only a visual camera effect)
- **If you use SpaceFog Prefab (More details, see *SpaceFog* on page 27)**
  - Reduce the number of fog particles
  - Consider removing space fog all together for poorly performing devices (since it is only a visual camera effect)
- **If you use AsteroidField Prefab (More details, see *AsteroidField* on page 28)**
  - Reduce the number of asteroids in the field (you can make the range smaller if you still want to keep the density of the asteroid field)
  - Disable “Fade asteroids” (should only be little if any change since fading is done in GPU shader)

RECOMMENDED WORKFLOW

The recommended method to work with SPACE for Unity is to keep a dedicated Unity project for creating space scenes which are then exported and imported into your game.

The reasons why it is advisable to use a dedicated project to create scenes are, for example:

- Avoid having the entire SPACE for Unity asset library in multiple copies for each game taking up gigabytes of disk space
- If you want to change texture resolutions in a game it does not affect the entire Space Unity project and hundreds of textures, just the once that are in use for your game
- You avoid your game project become cluttered with parts and assets of Space Unity that you do not use



EXPORTING SPACE SCENES

Exporting a space scene requires that you first create a space scene (See *Creating Space Scenes* on page 109)

1. Go to **File | Save Scene as…** and give the scene a unique name, e.g. "SpaceScene1"
2. **Right-click** on the scene in the Project window and select "**Export Package…**"
3. Ensure all assets (including dependencies) are selected and click "**Export…**"
4. Choose an appropriate path and name that you remember, click "**Save**"
5. Repeat Steps 1-4 for each scene that you plan to import to your game

IMPORTING SPACE SCENES

Once you have followed the steps to export your Space Scene(s) (see above) you are ready to import them into your game project.

1. Load your Unity game project
2. Go to **Assets | Import Package | Custom Package…**
3. Select the package(s) containing the Space Scenes previously exported
4. Import the package(s) to your scene

Note: *If you have not yet named Layer 20 in your game project, do so by following the procedure in* ***Set the Correct Name for User Layer 20*** *on page 8.*

USING MULTIPLE SPACE SCENES IN ONE UNITY SCENE

There may be times when you want to change your Space Scene, but not load a new Unity scene. This is possible by creating Prefabs of each Space Scene object that you import to your game project.

SPACESCENESWITCHER

The prefab **SpaceUnity/Prefabs/Tools/SpaceSceneSwitcher** using the script (**SpaceUnity/Scripts/SU_SpaceSceneSwitcher.cs**) will assist you with this method of switching between scenes quickly.

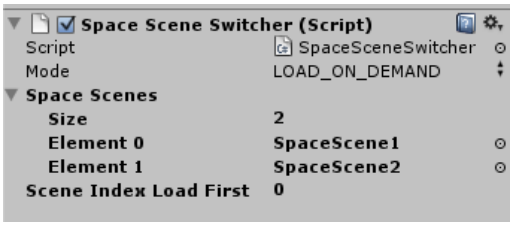


FIGURE 29 - SPACE SCENE SWITCHER

USING THE SPACESCENESWITCHER PREFAB

1. **Create prefab of each SpaceScene** that you wish to be able to quickly switch between (do this by dragging the SpaceScene object from the Hierarchy to your Project window which makes them into Prefabs)
2. **Name the prefabs** in the Project Window appropriately, e.g. SpaceScene1, SpaceScene2, or DarkUniverse, HomeSystem, etc.
3. Drag **SpaceUnity/Prefabs/Tools/SpaceSceneSwitcher** into anywhere in your scene
4. Click on the SpaceSceneSwitcher in the Hierarchy (so you can edit values in the Inspector)
5. **Drag** all your named **Space Scene Prefabs** from the Project window onto the **Space Scenes Array**
6. Select Mode "**LOAD_ALL_AT_STARTUP**" or "**LOAD_ON_DEMAND**"
    a. "LOAD_ALL_AT_STARTUP" will instantiate all space scenes when the scene is loaded and disable all but the active space scene. This method requires more initial load time and may occupy more video memory.
    b. "LOAD_ON_DEMAND" instantiates the space scene each time it is needed to reduce initial load time and conserve video memory
7. Set "**Scene Index Load Fist**" to the index value of the scene in the array you wish to activate upon start
8. Since SpaceSceneSwitcher is a static function, you can now switch between Space Scenes from any script using the call: SU_`SpaceSceneSwitcher.Switch("SCENE NAME");`
   For example: e.g. SU_`SpaceSceneSwitcher.Switch("SpaceScene2");`
   You can also switch scenes using array index, e.g. SU_`SpaceSceneSwitcher.Switch(3)` for the fourth Space Scene in the array (0 is the first scene.)

FIGURE 30 - SPACE SCENE SWITCHER DEMO

Try it: *Load the demo scene (Figure 31 - Space Scene Switcher DEMO) "SpaceUnity/_Scenes/Demo – Space Scene Switcher" to see how the switching works as it also comes with an additional GUI and hotkey script to switch between scenes.*

CAMERA EFFECTS

In order to improve the visual appearance, and in order to introduce a sense of speed when flying though space, a number of camera effect prefabs (and scripts) are available. These camera effects do not interact with a player or other game objects so they can be disabled to improve performance if necessary.

SPACEPARTICLES

The SpaceParticles prefab consists of a Shuriken Particle System which is controlled by the SU_SpaceParticles.cs script.

The script spawns a customizable number of particles in a sphere around its transform. The particles are fixed in world space (well not entirely true, they move slightly, but they are not in local space so if the parent transform moves the particles will not follow.)

SpaceParticles is then set as a child of another transform, usually the Main Camera (but it could also be another camera or a spaceship for example.) As the SpaceParticles transform moves through space (as a child of another transform, e.g. Main Camera) it will calculate the distance to the particles within the system. If a particle becomes out of range (distance is greater than the range parameter of the script) the script will "respawn" the particle somewhere in range (but out of sight) creating a seemingly infinite amount of space particles that never appear to die.

HOW TO USE SPACEPARTICLES

1. Drag the **SpaceParticles** Prefab from **SpaceUnity/Prefabs/CameraEffects/SpaceParticles** and drop it onto **Main Camera** to make it a child (or on whichever object should be in the center of the particle field)
2. That's it – press Play =)

CUSTOMIZE SPACEPARTICLES

The SpaceParticles prefab uses a generic script for space particles: **SpaceUnity/Scripts/SU_SpaceParticles.cs.**

Editor script **SpaceUnity/Editor/SU_SpaceParticlesEditor.cs** automatically overrides the GUI function of the Inspector to create a custom view (*Figure 32 - Space Particles Inspector*) for configuring SpaceParticles.

Detailed parameter description for SpaceParticles is available in *Table 10 - Space Particles Parameters* on page 26.

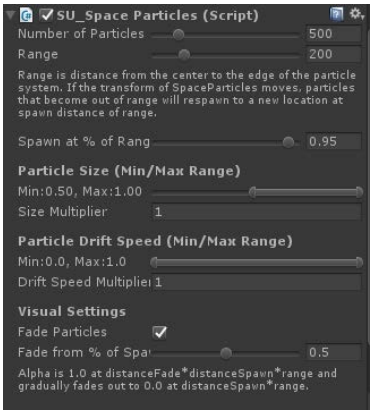


FIGURE 31 - SPACE PARTICLES INSPECTOR

TABLE 10 - SPACE PARTICLES PARAMETERS

Parameter (script variable)	**Values** **(Default)**	**Description**
Number of Particles (maxParticles)	Integer Slider 1 – 3000 **(Default: 500)**	The number of particles in the system. Reduce to improve performance or increase for added visual effect.
Range (range)	Float Slider 1 – 1000 **(Default: 200)**	The size the particle field. Particles that become out of range will re-spawn at the configured Spawn Distance. Keep the field as small as possible to increase density of particles. The speed of your object and size of particles will determine what range is suitable to ensure particles spawn out of sight.
Spawn at % of Range (distanceSpawn)	Float Slider 0.0 – 1.0 **(Default: 0.95)**	Percentile of Range at which distance out of range particles will re-spawn. E.g. if Range is 1000 and Spawn Distance is 0.95, the particles will spawn at 950 units from the center. Avoid using 1.0 (because particles may be out of distance on the frame right after spawning) and avoid too low values which will spawn particles in view. If you have a high value but particles still spawn in view, increase the Range and Max Particles values.
Particle Size (Min/Max Range) (minParticleSize) (maxParticleSize)	Min Max Float Slider 0.01 – 1.0 **(Default Min 0.50 Max 1.0)**	The size range of particles that are instantiated in the particle field. The particles will spawn with a random size between the Min and Max value (and then multiplied by the Size Multiplier value)
Drift Speed Min/Max (minParticleDriftSpeed) (maxParticleDriftSpeed)	Min Max Float Slider 0.0 – 1.0 **(Default Min 0.0 Max 1.0)**	The min/max range for drift/movement speed of particles. This is multiplied by the driftSpeedMultiplier to get the final value.
Drift Speed Multiplier (driftSpeedMultiplier)	Float field **(Default 1.0)**	The drift speed multiplier is multiplied by the Particle Drift Speed for spawning particles.
Fade Particles (fadeParticles)	**True (Default)** / False	Whether or not particles should fade in and out when close to spawn distance (disable to improve performance.)
Fade from % of Spawn (distanceFade)	Float Slider 0.0 – 1.0 **(Default: 0.5)**	Percentile of spawn distance from where particles should begin to alpha fade. E.g. if Range is 1000 and Spawn Distance is 0.95, the particles will spawn at 950 units from the center and if Fading Distance is also set to 0.95 particles will fade from 902.5 (950 x 0.95) where the particle's alpha value is multiplied by 1.0. The particle is faded completely to 0 at spawn distance, e.g. 950 according to the example.

Hint: *You can also customize the Shuriken Particle System parameters of SpaceParticles game object to change colors, particle texture, and other properties.*

SPACEFOG

The SpaceFog prefab consists of a Shuriken Particle System which is controlled by the SU_SpaceParticles script, just like the SpaceParticles prefab described above.

The difference with the SpaceFog and the SpaceParticles effect is that there are fewer particles creating the space fog and the particles are much larger in size with a fog-like appearance. The fog effect nicely adds to the visual appearance both when stationary as it overlays nebulas and planets and also in movement as it flies past the camera.

HOW TO USE SPACEFOG

1. Drag the Prefab S**paceUnity/Prefabs/CameraEffects/SpaceFog** and drop it onto **Main Camera** to make it a child (or whichever object should be in the center of the particle field)

CUSTOMIZE SPACEFOG

The SpaceFog prefab uses a generic script for space particles: **SpaceUnity/Scripts/SU_SpaceParticles.cs.**

Editor script **SpaceUnity/Editor/SU_SpaceParticlesEditor.cs** automatically overrides the GUI function of the Inspector to create a custom view (*Figure 33 - Space Fog Inspector*) for configuring SpaceFog.

For a detailed explanation of the parameters of the Space Particles script, see *Table 10 - Space Particles Parameters* on page 26.

Tip: *The main parameter you may want to change for SpaceFog is the brightness and color of the particle in the Shuriken Particle System. If you lower the brightness of the particle the effect will become more subtle, and if you increase the brightness of the particle it will appear more clearly. Remember: It is the color brightness you should configure, and not the alpha value of the particle you modify since it is an additive effect.*

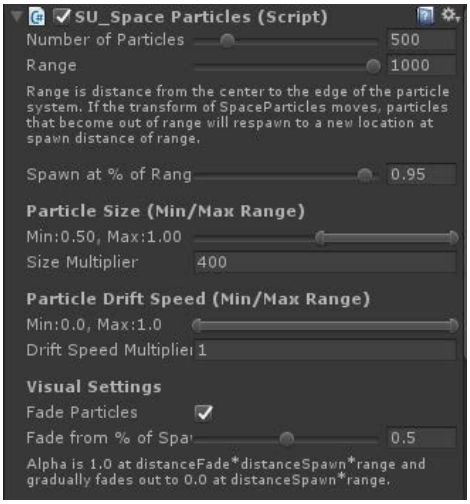


FIGURE 32 - SPACE FOG INSPECTOR

OBJECTS

ASTEROIDFIELD

The AsteroidField prefab (using the SU_AsteroidField script) adds asteroids to your game. The field can either be a local field (if the AsteroidField prefab is a standalone object in the Hierarchy), or an infinite field if set as a child of, for example, the player spaceship.

AsteroidField spawns Asteroid prefabs that can be either rigidbody or non-rigidbody objects which in turn the player can interact with for collisions.

The script spawns a customizable number of asteroids in a sphere around itself. The asteroids are instantiated as independent objects in world space so they will not be children of the AsteroidField game object nor will they relocate if the AsteroidField transform moves.

FIGURE 33 - ASTEROID FIELD

LOCAL ASTEROID FIELDS

If the AsteroidField transform does not have a parent and asteroid movement (either Drift speed parameter for non-rigidbody, or Velocity for rigidbody asteroids is set to 0) the asteroids will never re-spawn or relocate. The asteroid field will be of fixed size and the player will be able to fly in and out of the asteroid field.

CREATING A LOCAL ASTEROID FIELD

- Drag the Prefab **SpaceUnity/Prefabs/Objects/AsteroidField** and drop it in your scene (make sure it is **not created as a child** of an object that will potentially move)

See *Table 11 - AsteroidField Parameters* on page 29 to configure the asteroid field parameters further.

INFINITE ASTEROID FIELDS

If the AsteroidField moves through space, e.g. as a child of a SpaceShip, it will determine which asteroids become out of range and re-spawn them at a new position which is in range but out of sight. This creates a seemingly infinite asteroid field that a player can fly though forever. I suggest eating every now and then when testing out whether or not it really is infinite =)

CREATING AN INFINITE ASTEROID FIELD

- Drag the Prefab **SpaceUnity/Prefabs/Objects/AsteroidField** and drop it onto the object (usually a player's spaceship) that you want to experience the infinite asteroid field. Ensure it becomes a child of the moving object.

TABLE 11 - ASTEROIDFIELD PARAMETERS

Inspector GUI (script variable)	**Values** **(Default)**	**Description**
Number of Asteroids (maxAsteroids)	Integer Slider 10 – 5000 **(Default: 1000)**	The number of asteroids in the field. If performance becomes an issue, lower the amount of asteroids and reduce the range of the asteroid field to keep the same density.
Respawn if Destroyed (respawnDestroyedAsteroids)	**True (Default)** / False	If an asteroid is destroyed this decides whether it should be re-spawned or not.
Range (range)	Integer Slider 10 – 100000 (Default: 20000)	The size the asteroid field. Asteroids that become out of range will re-spawn at the configured Spawn Distance. Since asteroids are large objects you will probably need a large range. It is advisable to increase the range of your game camera (usually Main Camera) so the asteroids will not be drawn in a sliced fashion.
Spawn at % of Range (distanceSpawn)	Float Slider 0.0 – 1.0 **(Default: 0.95)**	Percentile of Range at which distance out of range particles will re-spawn. E.g. if Range is 1000 and Spawn Distance is 0.95, the particles will spawn at 950 units from the center. Avoid using 1.0 (because particles may be out of distance on the frame right after spawning) and avoid too low values which will spawn particles in view. If you have a high value but particles still spawn in view, increase the Range and Max Particles values.
Asteroid Scale (Min/Max Range) (minAsteroidScale) (maxAsteroidScale)	Min Max Float Slider 0.1 – 1.0 **(Default Min 0.1 Max 1.0)**	The scale range of asteroids that are instantiated in the asteroid field. The asteroids will spawn at a random scale between the Min and Max value (and then multiplied by the Scale Multiplier value)
Scale Multiplier (scaleMultiplier)	Float field **(Default 1.0)**	The scale multiplier is multiplied by the Asteroid Scale for spawning asteroids to accommodate games of any scale.
Is Rigidbody (isRigidbody)	True / **False (Default)**	Whether or not the spawned asteroids should be rigidbody objects. (If true, this requires that the asteroid prefabs configured blow have rigidbody objects components.)
Mass [rigidbody only] (mass)	Float Field **(Default: 1000)**	Base mass of the asteroid if it is a rigidbody (isRigidbody = true.) This mass value is multiplied by the random random for the asteroid (mass * <asteroidScale> * scaleMultiplier)
Angular Velocity Min/Max [rigidbody only] (minAsteroidAngularVelocity) (maxAsteroidAngularVelocity)	Min Max Float Slider 0.1 – 1.0 **(Default Min 0.1 Max 1.0)**	The min/max range for angular velocity (rotational speed of rigidbody asteroids) that should be applied to spawned rigidbody asteroids. This is multiplied by the angularVelocityMultiplier parameter to get the final value.
Angular Velocity Multiplier [rigidbody only] (angularVelocityMultiplier)	Float field **(Default 1.0)**	The angular velocity multiplier is multiplied by the Asteroid Angular Velocity for spawning asteroids.
Velocity Min/Max [rigidbody only] (minAsteroidVelocity) (maxAsteroidVelocity)	Min Max Float Slider 0.0 – 1.0 **(Default Min 0.0 Max 1.0)**	The min/max range for velocity (drift/movement speed of rigidbody asteroids) that should be applied to spawned rigidbody asteroids. This is multiplied by t
Velocity Multiplier [rigidbody only] (velocityMultiplier)	Float field **(Default 1.0)**	The velocity multiplier is multiplied by the Asteroid Velocity for spawning asteroids. This is accommodate for games of any scale.
Rotation Speed Min/Max [non-rigidbody only] (minAsteroidRotationSpeed) (maxAsteroidRotationSpeed)	Min Max Float Slider 0.1 – 1.0 **(Default Min 0.1 Max 1.0)**	The min/max range for rotational speed of non-rigidbody asteroids. This is multiplied by the rotationSpeedMultiplier parameter to get the final value.

Rotation Speed Multiplier [non-rigidbody only] (rotationSpeedMultiplier)	Float field **(Default 1.0)**	The rotation speed multiplier is multiplied by the Asteroid Rotation Speed for spawning asteroids.
Drift Speed Min/Max [non-rigidbody only] (minAsteroidDriftSpeed) (maxAsteroidDriftSpeed)	Min Max Float Slider 0.0 – 1.0 **(Default Min 0.0 Max 1.0)**	The min/max range for drift/movement speed of non-rigidbody asteroids. This is multiplied by the driftSpeedMultiplier to get the final value.
Drift Speed Multiplier [non-rigidbody only] (driftSpeedMultiplier)	Float field **(Default 1.0)**	The drift speed multiplier is multiplied by the Asteroid Drift Speed for spawning asteroids. This is accommodate for games of any scale.
Fade Asteroids (fadeAsteroids)	**True (Default)** / False	Whether or not asteroids should fade in and out when close to spawn distance (disable to improve performance.) Enabling this option will create an additional transparent texture for each asteroid material. The script will automatically replace the material of a fading asteroid to the transparent material and adjust the alpha value. Once an asteroid is within distance the material will be replaced with the original non-transparent material again.
Fade from % of Spawn (distanceFade)	Float Slider 0.5 – 0.98 **(Default: 0.7)**	Percentile of spawn distance from where asteroids should begin to alpha fade. E.g. if Range is 20000 and Spawn Distance is 0.95, the asteroids will spawn at 19000 units from the center and if Fading Distance is set to 0.7 asteroid will fade from 13300 (19000 x 0.7) where the asteroids scale is multiplied by 1.0. The asteroid is faded/scaled completely to 0 at spawn distance, e.g. 19000 according to the example.
Poly Count (polyCount)	Enum Popup Low / Medium / **High**	The polygon count / quality of asteroids in the field. This requires that the Asteroid Prefabs used in the field have three levels of quality and that they use the Asteroid script.
Poly Count Collider (polyCountCollider)	Enum Popup **Low** / Medium / High	The polygon count / quality of asteroids' colliders in the field. This requires that the Asteroid Prefabs used in the field have three levels of quality and that they use the Asteroid script. **It is recommended to keep the collider quality at Low as complex colliders will have a great impact on performance.**
Asteroid Prefabs (prefabAsteroids)	Array of Prefabs	The Asteroid Prefabs that to randomly choose from when spawning asteroids. This is only the mesh shape of the asteroid (and not the materials.)
Asteroid Materials (materialsVeryCommon) (materialsCommon) (materialsRare) (materialsVeryRare)	4 Arrays of Materials	The Asteroid Materials to randomly choose from when spawning asteroids. There are four different arrays with different probabilities that a material from that array will be chosen. "Very Common" Materials are used 50% of the time "Common" Materials are used 30% of the time "Rare" Materials are used 15% of the time "Very Rare" Materials are used 5% of the time.

SPACE SCENE ELEMENTS

This section lists the various elements the Space Scene Construction Kit will create for you.

STATIC STAR

This is a single skybox texture with millions of distant stars and the same skybox texture is used on all 6 skybox cube faces to reduce memory use, rendering time, and distribution size. A particle system is not used for distant stars as millions of particles would be much too CPU-intensive.

TABLE 12 - STARS DETAILS

Asset Details	
Prefabs	n/a
Meshes	n/a
Texture Count	10 star textures (“8 sided” tileable in any direction) 4 nebula noise textures (“8 sided” tileable in any direction)
Material Count	None – material is generated by SU_StaticStars.cs
Resolution	4096 x 4096
Filters Available	Star Count, Color (Background Noise)
Textures Path	SpaceUnity/Textures/StaticStars
Materials Path	n/a
Scripts	n/a
Lights	n/a

NEBULAS

Nebulas are textures that are projected on hemispheres with random rotation to create a distant sphere. All the nebula materials have been categorized with brightness, colors, styles, and complexity enabling you to customize your space scene with a specific mood (e.g. dark and hazardous vs. bright and colorful.)

TABLE 13 - NEBULAS DETAILS

Asset Details	
Prefabs	SpaceUnity/Prefabs/SpaceSceneElements/Nebula
Meshes	SpaceUnity/Meshes/NebulaHemisphere (224 triangles)
Texture Count	339 (211 new textures added in v1.5)
Material Count	339 (Particle/Additive)
Resolution	1024 x 1024
Filters Available	Brightness (5 levels), Color, Complexity (3 levels), Style
Textures Path	SpaceUnity/Textures/Nebulas
Materials Path	SpaceUnity/Materials/Nebulas
Scripts	n/a
Lights	n/a

GALAXIES

Galaxies are textures that are projected on smaller hemispheres much like the nebulas. Galaxies can also vary in distance (size) and they can optionally also be light sources with lens flares.

TABLE 14 - GALAXIES DETAILS

Asset Details	
Prefabs	SpaceUnity/Prefabs/SpaceSceneElements/Galaxy
Meshes	SpaceUnity/Meshes/GalaxyHemisphere (160 tirangles)
Texture Count	14
Material Count	14(Particle/Additive)
Resolution	1024 x 1024
Filters Available	Color
Textures Path	SpaceUnity/Textures/Galaxies
Materials Path	SpaceUnity/Materials/Galaxies
Scripts	n/a
Lights	Optional (with optional light flare)

PLANETS

Planets are 3D spheres with a customizable level of mesh detail. The editor window enables you to add up to five planets in your scene at various distances. Planets are categorized in terms of climates (Alien, Desert, Earth-Like, Ice, and Molten).

Some planets have night lights (Earth-like) and molten planets have additional illumination maps. All textures have a resolution of 4096x4096.

TABLE 15 - PLANETS DETAILS

Asset Details	
Prefabs	SpaceUnity/Prefabs/SpaceSceneElements/PlanetHighPoly SpaceUnity/Prefabs/SpaceSceneElements/PlanetMediumPoly SpaceUnity/Prefabs/SpaceSceneElements/PlanetLowPoly
Meshes	SpaceUnity/Meshes/SphereHighPoly (16128 tiangles) SpaceUnity/Meshes/SphereMediumPoly (3968 tiangles) SpaceUnity/Meshes/SphereLowPoly (960 tiangles)
Texture Count	12 (Diffuse, Normals, Illumination[1], Lights[2])
Material Count	12
Resolution	4096 x 4096
Filters Available	Climate (Alien, Desert, Earth-Like, Gas, Ice, Molten)
Textures Path	SpaceUnity/Textures/Planets
Materials Path	SpaceUnity/Materials/Planets
Scripts	SpaceUnity/Scripts/SU_Planet.cs (C#) *(Used for rotation of planet around its own axis)*
Lights	n/a

PLANET ATMOSPHERES

[1] Only available for planets that light up without a light source, e.g. lava and molten rock planets
[2] Only available for some inhabited planets

Planets have atmospheres which are separate objects with an atmosphere shader and material.

TABLE 16 - PLANET ATMOSPHERE DETAILS

Asset Details	
Texture Count	n/a (Color only)
Material Count	20 (Custom Shader: SpaceUnity/Shaders/PlanetRings)
Materials Path	SpaceUnity/Materials/Atmospheres
Shader	SpaceUnity/Shaders/PlanetAtmosphere

PLANET RINGS

Planets can also have rings (like Saturn's rings) of varying widths, details, and textures.

TABLE 17 - PLANET RINGS DETAILS

Asset Details	
Texture Count	20 (RGBA)
Material Count	20 (Custom Shader: SpaceUnity/Shaders/PlanetRings)
Resolution	512 x 1
Filters Available	n/a
Textures Path	SpaceUnity/Textures/Rings
Materials Path	SpaceUnity/Materials/Rings
Scripts	n/a
Lights	n/a

MOONS

Planets can also have stationary or orbiting moons with customizable orbit and rotational speed.

TABLE 18 - MOONS DETAILS

Asset Details	
Prefabs	SpaceUnity/Prefabs/SpaceSceneElements/MoonHighPoly SpaceUnity/Prefabs/SpaceSceneElements/MoonMediumPoly SpaceUnity/Prefabs/SpaceSceneElements/MoonLowPoly
Meshes	SpaceUnity/Meshes/SphereHighPoly (16128 tiangles) SpaceUnity/Meshes/SphereMediumPoly (3968 tiangles) SpaceUnity/Meshes/SphereLowPoly (960 tiangles)
Texture Count	5 (Diffuse, Normals)
Material Count	5
Resolution	1024 x 1024
Filters Available	n/a
Textures Path	SpaceUnity/Textures/Moons
Materials Path	SpaceUnity/Materials/Moons
Scripts	SpaceUnity/Scripts/SU_Moon.cs (C#) *(Used for rotation of planet around its own axis and for orbiting the parent planet)*
Lights	n/a

LOCAL STARS

A local star is usually the main source of light in the space scene. It is constructed with particle effects and a point light with lens flare. The stars are animated with cascades of fire (referred to as prominence.) Local stars come in three different sizes and in four different colors (Yellow, Orange, Blue, and Red.)

Since particle effects don't scale well in Unity, the local stars come in three fixed sizes: Large, Medium and Small.

TABLE 19 - LOCAL STARS DETAILS

Asset Details	
Prefabs	SpaceUnity/Prefabs/SpaceSceneElements/LocalStarLarge SpaceUnity/Prefabs/SpaceSceneElements/LocalStarMedium SpaceUnity/Prefabs/SpaceSceneElements/LocalStarSmall
Meshes	n/a
Texture Count	13
Material Count	13 (Particle/Additive)
Resolution	Various
Filters Available	Color
Textures Path	SpaceUnity/Textures/LocalStars
Materials Path	SpaceUnity/Materials/LocalStars
Scripts	n/a
Lights	Point Light (with optional flare)

TRAVELWARP

The SU_TravelWarp component is used to travel very fast within a scene. Since the nebulas, stars, and planets are rendered by a dedicated background camera, they are normally static within your game. If you want to travel fast within a space scene, e.g. travel to a planet or around a star, you can use the SU_TravelWarp component attached to a game object and make it relatively move within a space scene and it also allows you to move any object around you to move rapidly away from you to create the illusion that the scene is much larger than supported by Unity.

FIGURE 34 – TRAVELWARP VISUAL EFFECT

HOW DOES IT WORK?

THE VISUAL EFFECT

The visual effect of traveling fast is achieved by the component creating a long inverted tube mesh which is cone-shaped at both ends. The mesh follows the rotation and position of the parent object, e.g. the spaceship. When the SU_TravelWarp public bool property Warp is set to enabled the tube will gradually be introduced and the shader (SU_WarpFXDistortion) and texture used on the tube is animated and it grabs the screen with a “Grab Pass” in the shader and offsets things around the spaceship by the amount of the warp texture. The warp texture is simply a noise texture that is tiled in the tube and it is scrolled along the tube by the shader.

- The speed of the scrolling is controlled by the visualTextureSpeed variable but the Y-tiling texture tiling and the maxSpeed also affects how fast the effect is. There is no clamping of the visualTextureSpeed, you can set it as fast or slow as you want. Balance the variables to achieve the look you want.
- The magnitude of the effect is controlled by the visualWarpEffectMagnitude variable but the speed also affects it so find a value that is appropriate, the value is clamped between 0.0 – 1.0.

If the object that SU_TravelWarp component is attached to has a rigidbody it will calculate the local angular velocity of the object and bend the “tube” to create a worm-hole like effect.

The SU_WarpFXDistortion shader will grab anything on the screen with a RenderQueue value of < 1990 and apply the distortion effect to it. It operates on the main camera so everything that was previously rendered by the SpaceScene camera will automatically be included and affected by the warp regardless of RenderQueue value. If you want objects to be affected by the warp effect you will either need to change the render queue of such objects to a value < 1990 or adjust the warp shader to have a higher render queue (but then you will need to modify objects that are actually warping in the tunnel to have a higher value so those objects are not included in the warp effect). The asteroids, for example, has a RenderQueue of 1900 so they are affected by the distortion effect.

THE MOVEMENT WITHIN A SCENE

Movement in a large space scene, millions of miles across, is difficult to achieve since the Unity scene itself only effectively is 100km in diameter or so (well, it's larger but precision becomes too poor at such distances from the origin of the scene that you don't want object that far away as they start to jitter and skip around, Google "Floating Point Origin" for more on this topic).

To achieve the effect that we are moving across vast distances in space we need to do relative movement. Relative to the stars, planets, and nebulas in the scene – but also relative to other objects around you like spaceships and asteroids.

The SU_TravelWarp component has a basic concept of achieving this effect. As you enable the Warp, two things happen:

- You begin to relatively move within the space scene. The component uses the **acceleration** to incrementally accelerate you to the **maxSpeed** variable (change both to higher/lower values as desired). The current speed is then passed to the new public Move(Vector3 _vector) method of SU_SpaceSceneCamera and the relative position within the space scene is modified this way. Your *actual movement in Unity is still quite small* but it's the relative movement to the SpaceSceneCamera that makes you appear to be traveling fast.
- Other objects need to *move away* from *you.* Unless they are moved away relatively to you, objects like small asteroids and spaceships would remain right next to you as you warp at super high speeds. This would not look natural. Imagine having an asteroid the same size as your spaceship right next to you, then you enter warp speed and start moving towards a star at a high speed, but the asteroid still sits right next to you. This is why we need to move other objects away from you at an incredible speed so you appear to warp away from them. This is done by the **surroundingObjectMultiplier** which is default set to 20000. As you begin to warp, objects are translated (moved) away from you at that high rate so they appear to just disappear in a split second. You don't want *all* objects to be moved away relatively from you – some things need to be excluded. There are a few exclude options available and some of them are enabled by default.

CONFIGURABLE VARIABLES

- **warpTexture** – a texture with RGB noise that can be tiled
- **textureTiling** – default Vector2 XY is 1,10 – you can change this for custom stretching to your liking.
- **excludeDeepSpaceLayer** – you probably want this enabled (excluded) so all objects on layer 20 are not moved away from you.
- **excludeCameras** – you probably want this enabled (excluded) so cameras are not moved away from you.

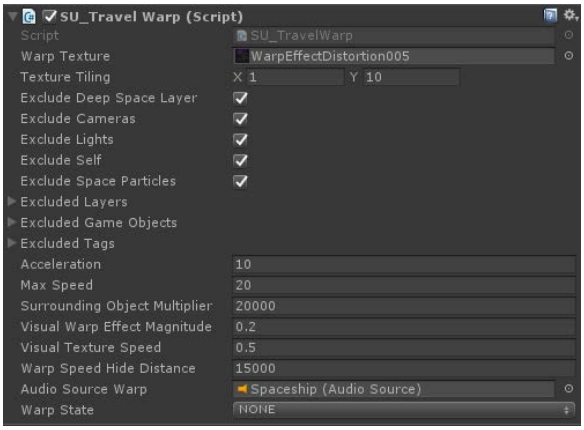


FIGURE 35 - SU_TRAVELWARP.CS

- **excludeLights** – you don't want the star lights to move away from you, but if you have other lights in your scene you may want to disable this and then add the exclusions manually in the excludedGameObjects array.
- **excludeSelf** – you probably want this enabled (excluded) since you don't want to warp away from yourself.
- **excludeSpaceParticles** – exclude space/particles yes or no.
- **excludedLayers (array)** – add integers of layers here that you want to be excluded from the relative warp movement.
- **excludedGameObjects (array)** – add gameobjects to this array that you want to be excluded from the relative warp movement. For example, in demo scenes where there are multiple spaceships that warp in formation together, the other spaceships (and all their child objects) have been included in this array. Importantly the other spaceships do not have the TravelWarp effect – you only want one instance of that.
- **excludedTags (array)** – add tags to this array that you want to be excluded from the relative warp movement.
- **acceleration** – How fast the object should be accelerated
- **maxSpeed** – The top speed of the warp. The actual speed will gradually be incremented by acceleration (multiplied by Time.deltaTime) to reach maxSpeed. The speed affects how fast you move relatively within the space scene and partially also the speed of the visual effect. There is no scientific accuracy to speed or acceleration in terms of meters per second or speed of light.
- **surroundingObjectMultiplier** – How fast surrounding objects, like asteroids and other spaceships, should be moved away from you when you warp. Set a value that achieves the look/speed you want, default is 20000.
- **visualWarpEffectMagnitude** – How strong the visual warping effect should be. This value is clamped between 0.0 and 1.0.
- **visualTextureSpeed** – How fast the texture should be scrolled along the inverted tube for a faster/slower warp effect. The texture Y-tiling and current warp speed also affects the visual scrolling speed as they are multiplied by one another.
- **warpSpeedHideDistance** – At what distance objects that you have moved away from should be disabled. They are re-enabled once again as you stop warping. This is to stop asteroids and particles from being re-spawned during your warp.
- **audioSourceWarp** – an audio source attached to the gameobject that should have a looping sound for the warp assigned to it. Default volume should be set to 0 on the audio source and it should also be set to looping.

HOW TO CONFIGURE AND WARP?

1. Attach the SU_TravelWarp component to the spaceship that should be able to warp.
2. Configure the variables as desired (see above, and also look at the Spaceship object in the demo scenes).
3. Use **gameObject.GetComponent<SU_TravelWarp>().Warp = true;** (accelerates to maxSpeed) in a script on your spaceship to begin warping and use **gameObject.GetComponent<SU_TravelWarp>().Warp = false;** to stop warping (decelerates to 0).

SCRIPTS

TABLE 20 - SCRIPTS

Script (C#)	Path	Description
SU_Asteroid.cs	SpaceUnity/Scripts	This script handles an asteroid in terms of rotation and movement. (*See AsteroidField page 28*)
SU_AsteroidFadeOrigin.cs	SpaceUnity/Scripts	This script should be attached to a gameobject, normally the main camera, as it will designate the origin/center of an endless asteroid field. The distance for visibility and fading is calculated from this.
SU_AsteroidField.cs	SpaceUnity/Scripts	This script creates a localized asteroid field around itself. As the object moves the asteroids will optionally re-spawn out of range asteroids within range (but out of sight.) (*See AsteroidField page 28*)
SU_AsteroidFieldEditor.cs	SpaceUnity/Editor	Override Inspector to customize configuration of SU_AsteroidField.cs (*See AsteroidField page 28*)
SU_CameraFollow.cs	SpaceUnity/Demo/Scripts	Smooth camera follow script used to follow an object (Transform)
SU_Explosion.cs	SpaceUnity/Demo/Scripts	Simple script to destroy instantiated explosions after a delay.
SU_LaserImpact.cs	SpaceUnity/Demo/Scripts	Laser impact script for sound effect and impact effect.
SU_LaserShort.cs	SpaceUnity/Demo/Scripts	Script for firing laser weapon that spawns a bullet that can destroy objects that it hits.
SU_Moon.cs	SpaceUnity/Scripts	Script for the rotational and orbiting behaviours of moons.
SU_Planet.cs	SpaceUnity/Scripts	Script for the rotational behaviour of planets.
SU_SpaceParticles.cs	SpaceUnity/Scripts	Script spawns particles in a sphere around its parent. The particles live for an infinite period of time but they will be relocated when they are beyond "range." (*See SpaceParticles page 25*)
SU_SpaceSceneCamera.cs	SpaceUnity/Scripts	This script is attached to the Space Scene camera and it renders the space scene which is then used as a background to the main camera.
SU_SpaceSceneSwitcher.cs	SpaceUnity/Scripts	Switches between Space Scenes that have been saved as Prefabs in the Project window. (*See SpaceSceneSwitcher page24*)
SU_SpaceSceneSwitcherDemoGUI.cs	SpaceUnity/Demo/Scripts	Demo script on how to use SU_SpaceSceneSwticher.cs
SU_Spaceship.cs	SpaceUnity/Demo/Scripts	Demo script for controlling the Spaceship prefab
SU_StaticStars.cs	SpaceUnity/Scripts	Script that procedurally creates a cube named CustomSkybox and applies the static stars texture and optionally the nebula noise texture. It creates a skybox with 1 (or 2) draw calls since we don't use the Unity original skybox. The stars and nebula noise can be tinted by color values.
SU_Thruster.cs	SpaceUnity/Demo/Scripts	Demo script for adding thrusters to the Spaceship that apply force to the parent (spaceship) rigidbody.
SU_TravelWarp.cs	SpaceUnity/Scripts	This script procedurally creates an inverted "tube" with cone-shaped caps. An object that has this component added to it can set the public "Warp" property to true and the inverted tube will be enabled and a texture is scrolled on the inside of the tube to create the illusion of warping through space. Paramters can be configured for the desired look. This script also moves the gameobject relatively within the space scene and it also moves other objects relatively away from you to simulate a bigger scene. Unity only supports scenes of roughly up to 100x100km so an approach of moving objects relative to one another must be used to simulate larger distances.

SPACESHIP PREFAB

Included in the package you get a spaceship prefab fitted with engines and working thrusters (sound effect, particle effect affecting the rigidbody ship.) You can use the spaceship to test out your space scenes or use it in your game, it's up to you =)

- 3D Mesh (ship + engines)
- Textures
    - 1024x1024 Ship Diffuse map
    - 1024x1024 Ship Specular map
    - 1024x1024 Ship Normal maps
    - 512x512 Engine Diffuse map
    - 512x512 Engine Specular map
    - 512x512 Engine Normal Map
- Scripts
    - Smooth Camera Chase (C#)
    - Spaceship Control (C#)
    - Thruster Control (C#)
    - LaserShot (C#)
    - LaserImpact (C#)
- High quality audio
    - Seamless looping thruster effect
    - Laser shot effect
    - Explosion sound effect
- Particle Effects
    - Thruster Flames (Shuriken)
    - Explosion (Shuriken)

HOW TO USE THE SPACESHIP

1. Drag the prefab **SpaceUnity/_Demo/Prefabs/Spaceship** to your scene or hierarchy window
2. The spaceship is now ready to fly using configured Vertical / Horizontal axis controllers and fire buttons

TECHNICAL SUPPORT

For any technical issues, first refer to the Troubleshooting section on page 40. You can also have a look in the SPACE for Unity thread in the Unity Forum, or contact me on directly at stefan@imphenzia.com. Finally, additional information is also available on the official web site, www.imphenzia.com/space-for-unity

TROUBLESHOOTING

- **Planet atmospheres look wrong**
  You may have rotated the planet parent object which breaks the child atmosphere object. The proper way to rotate planets are to rotate the child object "PlanetObject" (and the proper way to move planets is to move the parent object, e.g. "PlanetHighPoly.")

- **Planets behave strange (changes color or night city lights go on and off)**
  - You may have added light sources to your scene without setting Culling Mask to exclude layer 20 ("Deep Space") (see *Lights*, page 9), or not marked the light as "Not Important" (see *Lights*, page 9)

- **Unity crashes with "Fatal Memory" error**
  This can happen often if you are changing several textures to high resolution (e.g. 4096x4096.) The bug has been reported to Unity and hopefully memory management in Unity should allow change of texture resolutions better in the future. Ensure that you work with texture resolution changes in your game project and not the space scene construction kit project (see *Recommended Workflow*, page 23)

TABLE OF FIGURES

TABLE OF TABLES

VERSION HISTORY

VERSION 1.5

- **Added SU_TravelWarp script and shader**
  - A new script component, SU_TravelWarp, can be added to objects that should travel fast within a space scene. The demo spaceship prefab now has this enabled in all the demo scenes so you can inspect how it is configured and how it behaves. Try pressing "Space" key in the demo scenes and the spaceship will accelerate into warp mode (if you then also press "Right Shift" key the demo ship also enters into an ultra fast warp).
  - The SU_TravelWarp component procedurally generates an inverted tube mesh which has cone-shaped caps on both ends. The tube is hidden by default and it moves and rotates with the object to which it is attached. When you set the public bool property "Warp" to true the tube becomes visible based on strength parameters configured in the inspector for SU_TravelWarp. The different variables will allow you to change the magnitude and speed of the visual effect. The component also moves the object relatively within the space scene by calling the "Move" method on SU_SpaceCamera. This way the warp allows you to travel within the space sphere of objects – but be aware, you can also exit the sphere entirely so make sure to control speed and bounds within your own game.
  - To simulate that you are not only moving in the space scene but also away from smaller objects around you, the SU_TravelWarp component will offset any other game objects in the scene (unless excluded) relative to the direction you travel. E.g. asteroids and surrounding spaceships are moved relatively away from you in the scene. It would look odd if they didn't move because if you have an asteroid next to you as you warp away you don't want it sitting right next to you as you fly at the speed of light =) You can excluded objects from being moved relatively by adding them to the Tag/Layer/GameObject arrays of the SU_TravelWarp component. There are also options to exclude, cameras, lights, etc.
  - If the object to which SU_TravelWarp is attached has a rigidbody the local angular momentum is calculated and used by the shader to bend the tube creating a worm-hole like effect.
  - **Note:** Using the SU_TravelWarp component can be tricky as it fundamentally affects your game with relative movement between objects. You will likely need a fair bit of programming skills to implement relative movement within a scene to suit your needs. Also search on the topic "Floating Origin" on the Internet. Representing a large universe within a small scene is quite tricky.
  - *See section TravelWarp on page 35 for details.*
- **"Static Stars" replaces "Stars"**
  - Ten new grayscale 4096x4096 textures and four grayscale 4096x4096 nebula noise textures was added. A custom script, SU_StaticStars, is added as a component to the SpaceCamera and the SU_StaticStars component procedurally generates a cube and creates a material that renders the stars and nebula noise. This uses only 1 draw call (optionally 2 if you enable nebula noise) and it also has the benefit of allowing custom tinting of stars and nebula noise. These new textures only use 15 MB of disk space and the number of colors are in the millions instead of fixed like before.
  - In previous versions, 60 large 4096x4096 textures were used where stars and nebula noise colors where baked into the same texture. This occupied over 200MB of disk space and affected the performance of working with the asset. It also required 6 draw calls since a traditional Unity skybox was used even though it was the same texture on all sides.
  - **Upgrade Notice:** Old scenes will not work since the original star textures have been removed. You will either need to recreate the "Static Stars" for the scene using the Space Scene Construction Kit editor window or you need to import textures from a previous version.

- **Added 211 new nebula textures**
  - There is now a total of 339 nebula textures instead of 128 for greater variety of scenes.
- **Added subfolder support to Nebulas, Planets, and Galaxies**
  - Materials can now be added into subfolders and used as a filter option when generating space scenes. This way it will be easier to only select new expansion packs or custom created content.
- **New asteroid materials**
  - New shaders created for asteroids to increase specular reflections.
- **New asteroid fading method**
  - Asteroids are now faded using scaling instead of alpha fading at the perimeter. It greatly increases performance since there is no need for performance heavy alpha transparent materials. It also looks better as asteroids that faded in front of a white background, like a star or galaxy, looked odd. The scaling is performed by the Vertex shader of the asteroid material so the mesh scaling is entirely handled by the GPU and does not impact performance.
- **Demo scenes updated**
  - Demo scenes updated and all the star textures were replaced by the new Static Stars component. SpaceShip now supports the travel warp if you press Space key (and additionally also Right Shift if you want to go really fast)

VERSION 1.07

- Added Unity 2017.2 package
- Removed Unity 3.x and 5.0x packages
- Updated URL to official website

VERSION 1.06

Only Unity version 5.5 supported!

- Removed deprecated code.

VERSION 1.05

Only Unity version 5.x supported!

FIXES

- Compiled for Unity 5.x
- Added pragma "keepalpha" for planet ring shader to allow transparency in Unity 5
- Removed compiler specific code for Unity version 3.x and 5.x
- Added interpolation to spaceship rigidbody and changed camera update mode to LATE_UPDATE for smoother movement

VERSION 1.03

FIXES

- Removed rewrote deprecated 4.x code in scripts
- Added compile dependent code to ensure Unity 4.1, 4.2, and 4.3 executes code
- Fixed problem with asteroids not being created after build if "fading" was enabled

    - (Added new shader SpaceUnity/AsteroidTransparent located in a Resources folder to ensure it is always included during the build process)
- Fixed flickering of atmospheres in Unity 4.1 and above
    - Fixed by changing point lights distance from 100000 to 20000 (precision issue)
- Fixed memory leak of explosions never being removed
    - Added SU_Explosion.cs script that destroys gameobject after delay

VERSION 1.02

FIXES

- Changed name to SPACE for Unity due to Unity Asset Store policy conflict.
- Prefixed all scripts with "SU_" to avoid naming conflicts.
- Prefab: SpaceUnity/Prefabs/SpaceSceneElements/LocalStarLarge, LocalStarMedium, LocalStarSmall
    - Flickering Planets in Unity 4
        - Changed Range of child Point light from 1E+10 to 100000 because it made the planet textures flicker in Unity v4. (Unity 3.5 supported "Infinity" for point lights which didn't work in 4.0 and apparently too high of a range causes flickering, presumably due to some floating point issues.) Note: If you have created Local Stars in your scene with broken prefab connection you will need to set these values manually as well on scenes you created.
    - Missing white star disc after disappearing out of view in Unity 4
        - The white star disc disappeared in Unity v4 after the local star had once been out of view. This was not the case in v3.5. On child ParticleSystem-Disc, changed Duration and Lifetime from Infinity to 100 and set particle system to looping to looping which keeps the disc visible at all times.
- Script: SpaceSceneSwitcher.cs
    - Added compiler version verification to use SetActive instead of deprecated command SetActiveRecursively in Unity v4
- Scene: "Demo - Planet Climates"
    - Enabled gameObjects MeshEngines and MeshSpaceship that were accidentally disabled before.
- Planet Material: "Planet-Desert-OrangeWithImpacts"
    - Added missing label "atmosphere-brown-medium" because planets of this type didn't receive an atmosphere when created due to the missing label on the asset.
- Scene: "Demo - Colorized Red"
    - Set background stars (which were missing) texture for SpaceScene camera.
- Prefab: _Demo/Prefabs/SpaceScene1 & SpaceScene2
    - Added background stars (which were missing) in SpaceScene1.
    - Replaced local stars which were not rendering prominence and cascades.

VERSION 1.01

First public release

Thank you…

…once again for purchasing SPACE for Unity. With your support, I'm able to continue to develop this product along with other assets for games.

https://www.imphenzia.com