JUPITER

USER GUIDE

(v1.2.10)

Procedural Sky for Unity

PROLOGUE

Jupiter is a simple and lightweight procedural skybox controller for Unity. This is a must-have asset which saves you hundreds hours of work and tons of effort, give you more time and resources to focus on other aspects of your game.

This User Guide is created to help you get started with the tool, including the most basic information about its features and workflow, as well as best practices, tips and tricks from its creator and showcasing amazing work from other successful developers.

To see release log, please visit this page.

For business, please contact: hello@pinwheel.studio

For support, report a bug or request a feature, please contact: support@pinwheel.studio

RELEASE LOG

1.2.10

NEW FEATURES

- Adding Remap filter for Overhead Cloud

FIXES

- Minor fixes

===

V1.2.9

NEW FEATURES

- Adding Overhead Cloud Shadow Casting as an experimental feature.

- Adjusting Overhead Cloud opacity.

===

V1.2.8.1

FIXES

- Minor DNC fixes.

===

V1.2.8

IMPROVEMENTS

- Adding HDR mode for sky color.

- DNC now become non-destructive (not modifying the Sky Profile directly).

- Optimizing material setup.

- Display a message to indicate that a property is overridden by DNC.

===

V1.2.7

FIXES

- Fix rendering issue in VR (Single Pass Instanced).

===

V1.2.6

FIXES

- Further fix for DNC environment reflection issue on mobile.

===

V1.2.5

FIXES

- Fix DNC environment reflection issue.

===

V1.2.4

IMPROVEMENTS

- Adding support for Unity 2020.

===

V1.2.3

IMPROVEMENTS

- DNC: render a custom cubemap for environment reflection.

- DNC: allow sun and moon to orbit around different pivots.

===

V1.2.2

IMPROVEMENTS

- Better stars placement.

===

V1.2.1

FIXES

- Fix Fog Sync issue in URP

===

V1.2.0

NEW FEATURES

- Adding Fog Sync option to sync fog with sky color, or animate overtime using day night cycle.

IMPROVEMENTS

- New button for adding DNC component.

- DNC animated properties will have default value instead of zero when added.

FIXES

- Default sky profile cannot be animated with DNC anymore.

===

V1.1.0

IMPORTANT!

- Please BACK UP your project and perform a clean update due to major package structure changes.

IMPROVEMENTS

- Compile Runtime and Editor code into their own assembly.

- Adding some test cases to detecting package mis-configuration.

===

V1.0.1

IMPROVEMENTS

- Adding HDR color picker for sun, moon and star.

===

V1.0.0

Initial release

INTRODUCTION

Jupiter is a user friendly sky system dedicated to help you create dynamic skybox, deeply focused on low-poly and stylized scenes, that can run well on both Mobile and Desktop applications, save you a lot of time and effort!

Jupiter is carefully designed to give you the fastest and easiest way to create beautiful sky, with many elements such as shining sun, twinkling stars, flowing cloud, etc. Even if you are a beginner or an advanced creator, it is just the right tool for you!

Jupiter also comes with sample assets and example scenes for you to play with. You can even use them in your commercial projects. Have fun!

FREQUENTLY USED EDITOR MENUS

You can easily find Jupiter functionalities in these editor menus:

- **Assets>Create>Jupiter>…**: Create specific assets like Sky Profile, Day Night Cycle Profile, etc.
- **GameObject>Create>3D Object>Jupiter Sky>...**: Create sky controller object in the scene.
- **Window>Jupiter>...**: Open additional editor window or configure Jupiter global settings.

CREATE AND CUSTOMIZE THE SKY

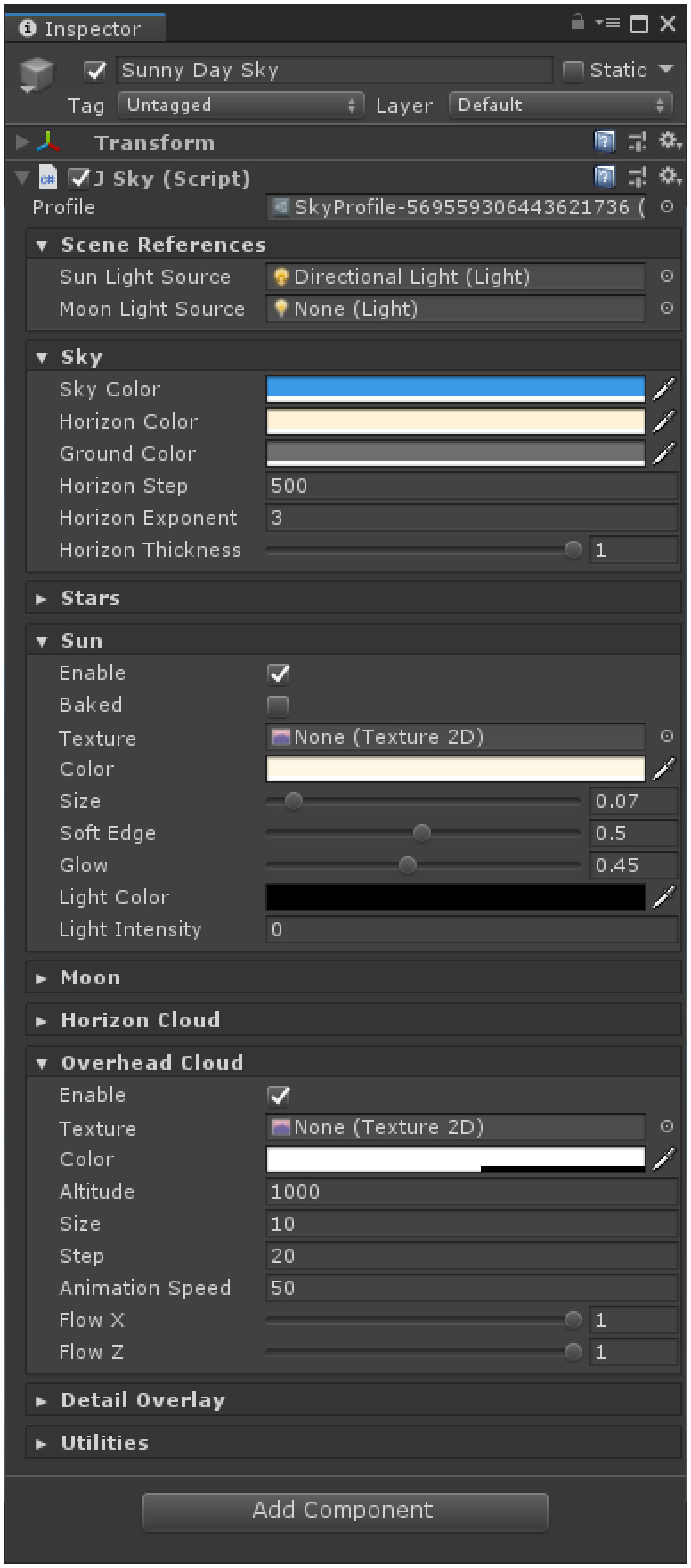
Create Sky object

To create a new sky object, go to **GameObject>3D Object>Jupiter>...** and select a template depend on your need, such as Sunny Day. You can customize its look later.

A new game object will be spawned into the scene.

A new Sky Profile asset will be created in Assets/ folder. You should rename and store it in a different directory for ease of management.

Select the sky game object, in the Inspector, you can see a various of settings for customize it.



Inspector

Sunny Day Sky — Static

Tag: Untagged — Layer: Default

Transform

J Sky (Script)

Profile	SkyProfile-569559306443621736 (

Scene References

Sun Light Source	Directional Light (Light)
Moon Light Source	None (Light)

Sky

Sky Color	
Horizon Color	
Ground Color	
Horizon Step	500
Horizon Exponent	3
Horizon Thickness	1

Stars

Sun

Enable	✓
Baked	
Texture	None (Texture 2D)
Color	
Size	0.07
Soft Edge	0.5
Glow	0.45
Light Color	
Light Intensity	0

Moon

Horizon Cloud

Overhead Cloud

Enable	✓
Texture	None (Texture 2D)
Color	
Altitude	1000
Size	10
Step	20
Animation Speed	50
Flow X	1
Flow Z	1

Detail Overlay

Utilities

Add Component

Scene References

Hold references to scene object such as Sun and Moon light source. These light sources are used to control Sun & Moon position, and should be **directional light**.

Sky

Control sky background gradient.

- Sky Color: Color of the upper sky.
- Horizon Color: Color of the horizon.
- Ground Color: Color of the lower sky.
- Horizon Step: Control the step/banding effect of the horizon.
- Horizon Exponent: Control the transition between horizon and upper/lower sky.
- Horizon Thickness: Thickness of the horizon.
- Fog Sync: Whether to synchronize fog color with sky/horizon/ground/custom color or not.
- Fog Color: Color of scene fog. Only available when Fog Sync set to Custom Color.

Stars

Control stars effect.

- Enable: Enable the effect or not.
- Baked: Whether to use baked stars (from cubemap) or render them procedurally.
- Cubemap: The stars cubemap to sample from.
- Twinkle Map: A grayscale map to animate the stars. A noise texture is good enough.
- Start/End: Visible range of the stars, on Y axis.
- Opacity: Overall opacity, for fading in/out between day and night time.
- Layers: Number of star layers.

For each star layer:

- Color: Color of the stars.
- Density: Density of the starfield.
- Size: Size of the stars.

- Glow: Stars brightness.
- Twinkle: How fast the stars to blink.

This is a heavy effect, each layer are calculate separately, more layer cost more time to compute.

Star layers with the same density produce the same star position. Consider to use different density value.

On mobile device, it’s better to use a cubemap for this effect.

Sun/Moon

Sun and Moon are separated effects with similar settings.

- Enable: Enable the effect or not.
- Baked: Whether to use baked sun/moon (from cubemaps) or render them procedurally.
- Cubemap: The cubemap to sample from.
- Texture: A custom texture for sun/moon.
- Color: Tint color.
- Size: Size of the sun/moon.
- Soft Edge: Slightly fade out the edge of the sun/moon.
- Glow: Control the sun/moon brightness and the halo.
- Light Color: Color of the sun/moon light source.
- Light Intensity: Intensity of the sun/moon light source.

Horizon Cloud

Animated cloud effect at the horizon.

- Enable: Enable the effect or not.
- Texture: Custom cloud texture. This will be shared with Overhead Cloud effect. Leave empty to use default cloud.
- Color: Color of the cloud.
- Start/End: Visible range of the cloud, on Y axis.
- Size: Size of the cloud.

- Step: Control the step/banding effect.
- Animation Speed: How fast the cloud animate.

Overhead Cloud

Animated cloud effect high above.

- Enable: Enable the effect or not.
- Texture: Custom cloud texture. This will be shared with Horizon Cloud effect.
- Color: Color of the cloud.
- Altitude: Height of the cloud layer.
- Size: Size of the cloud.
- Step: Control the step/banding effect.
- Animation Speed: How fast to animate the cloud.
- Flow X/ Flow Z: Flow direction on X/Z axis.
- Remap Min/Max: Apply remap filter for cloud opacity.

For performance reason, this effect will simulate 2 cloud layers on PC and 1 layer on Mobile.

Detail Overlay

Draw additional detail layer from a cube map, with a specific render order.

- Color: Tint color.
- Cubemap: The cubemap to sample from.
- Layer: the order to render the cubemap.
- Rotation Speed: How fast it rotate around Y-axis.

Utilities

Contains some utility settings:

- Allow Step Effect: Check to enable step/banding effect for some module. Step effect is heavy on Mobile devices since it use modulo (%) operation.

DAY NIGHT CYCLE

Create Day Night Cycle profile

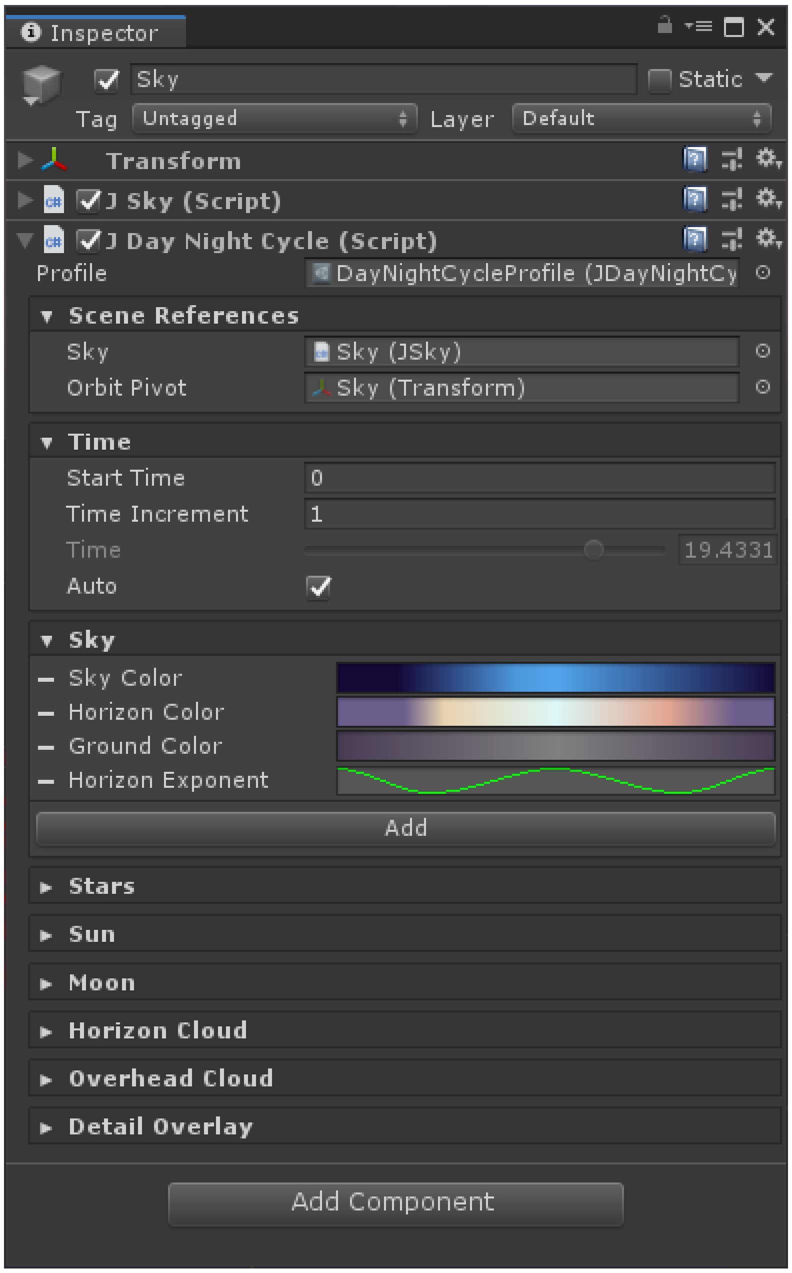
Day Night Cycle Profile is an asset type to store information about animated sky property, where you can override/animate them selectively depend on your need. This profile can be shared across many scenes.

To create a Day Night Cycle Profile, right click on the Project window and select **Create>Jupiter>Day Night Cycle Profile**. A new asset will be created, rename it and store in an appropriate folder.

Create Day Night Cycle controller

To animate the sky over time, you need to add a **JDayNightCycle** component to a game object in the scene, it's recommended to add to the Sky object for ease of management.

In the Inspector, drag and drop your Day Night Cycle Profile into the Profile slot, then its module will appear for you to customize:


Inspector

Sky ✓ — Static

Tag: Untagged — Layer: Default

Transform

✓ J Sky (Script)

✓ J Day Night Cycle (Script)

Profile: DayNightCycleProfile (JDayNightCy

Scene References
- Sky: Sky (JSky)
- Orbit Pivot: Sky (Transform)

Time
- Start Time: 0
- Time Increment: 1
- Time: 19.4331
- Auto: ✓

Sky
- Sky Color
- Horizon Color
- Ground Color
- Horizon Exponent

Add

Stars

Sun

Moon

Horizon Cloud

Overhead Cloud

Detail Overlay

Add Component

Scene References

Hold references to some objects in the scene.

- Sky: The sky object.
- Orbit Pivot: An anchor transform for orbiting sun and moon. It’s better to use the Transform of the same game object.

Time

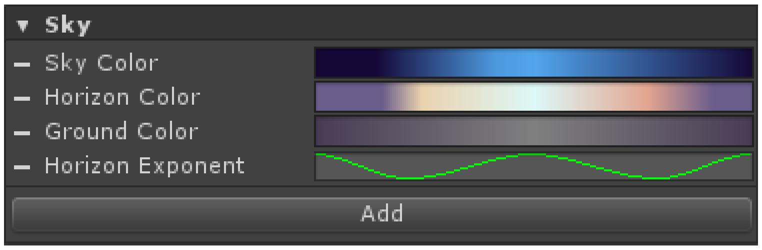
Contains timing settings.

- Start Time: The time when the game start.
- Time Increment: A small interval to increase each frame.
- Time: The current time.
- Auto: Should it use auto time increment or not. Disable this to set time manually.

Animate Sky Properties

The following modules are similar to Sky Profile, where you can define which properties should be animate overtime.

Click on **"Add"** button to add a new property, click on the **"-"** button to remove an added one.



A color property will be overridden by a gradient, while a number will be overridden by a curve.

For each gradient or curve, the evaluation time at 0.0 and 1.0 equal to 00:00 midnight (AM), and 0.5 equal to 12:00 noon (PM), adjust them for your desired effect.

See the demo scene for a sample setup.

Environment Reflection

The DNC component will spawn an internal Reflection Probe to capture the sky for environment reflection.

- Enable: Should it update environment reflection or not.
- Resolution: Size of the captured cubemap.

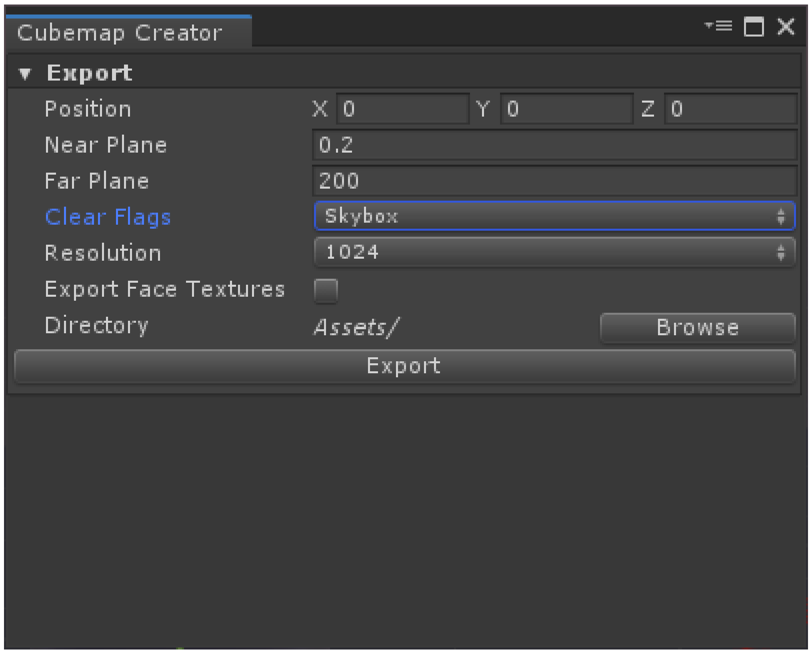
- Time Slicing: Render all faces at once or split to several frames.

BAKING SKY TO CUBEMAP

You can render your sky into a cubemap an use it statically, or render a specific effect of the sky material to use in Baked mode and save some performance on Mobile.

Cubemap Creator

To open the Cubemap Creator, go to **Window>Jupiter>Tools>Cubemap Creator.**


Cubemap Creator

Export

Position X 0 Y 0 Z 0

Near Plane 0.2

Far Plane 200

Clear Flags Skybox

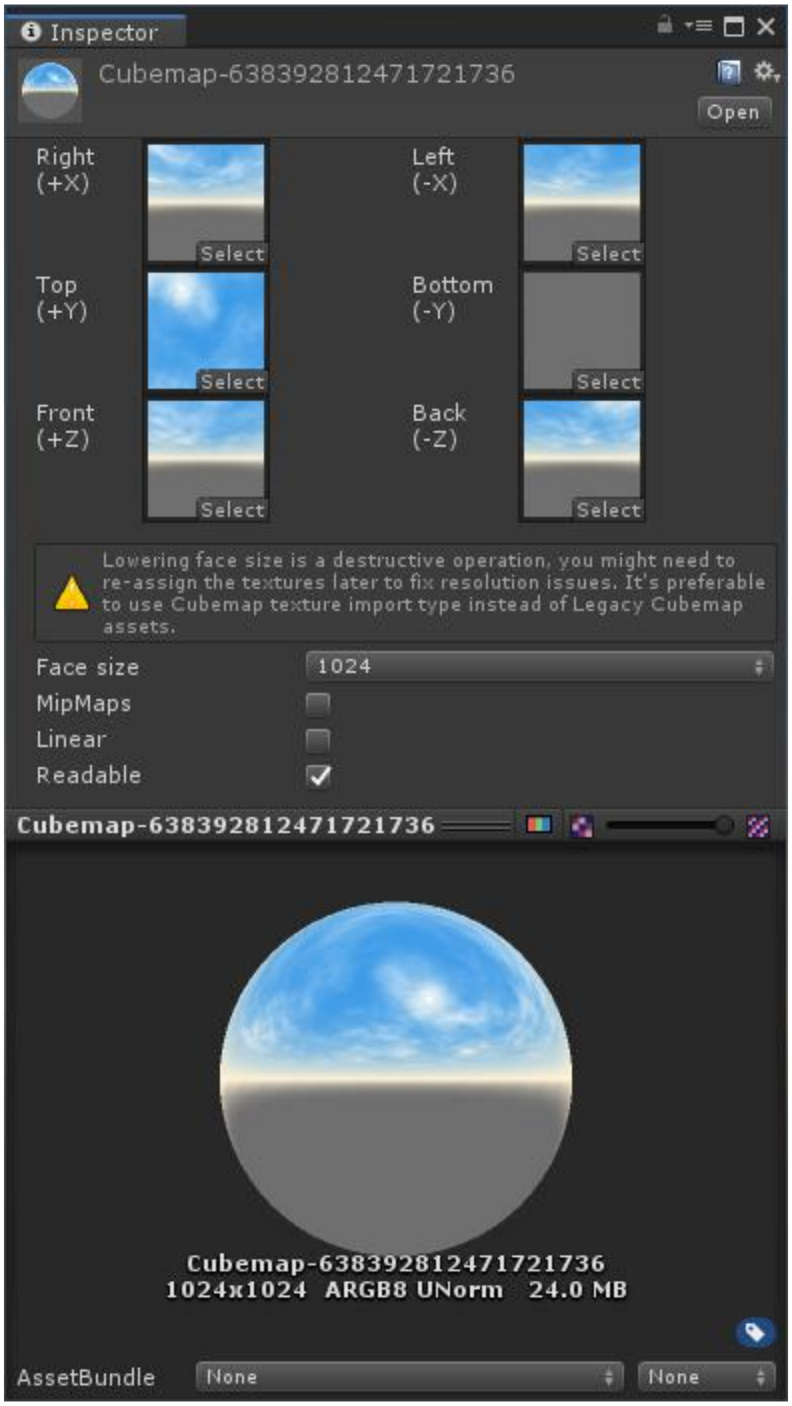
Resolution 1024

Export Face Textures ☐

Directory *Assets/*

- Position/Near Plane/Far Plane/Clear Flags: The camera properties to render the cubemap.
- Resolution: Size of each face on the cubemap.
- Export Face Textures: Check this on to export individual face textures along with the cubemap, for further polishing in other software.
- Directory: A folder to save these asset.

Hit Export to create the assets.



Lowering face size is a destructive operation, you might need to re-assign the textures later to fix resolution issues. It's preferable to use Cubemap texture import type instead of Legacy Cubemap assets.

Face size 1024
MipMaps ☐
Linear ☐
Readable ✓

Cubemap-638392812471721736

Cubemap-638392812471721736
1024x1024 ARGB8 UNorm 24.0 MB

AssetBundle None None

Use Cubemap as static skybox

To use the exported cubemap as static skybox, create a material with Skybox>Cubemap shader. Assign the cubemap to its slot, then assign the material to the skybox material slot in Lighting window.

Render separate sky effect into cubemaps

You can render separate sky effect such as Stars, Sun, Moon into cubemaps to use with Baked mode, which is a performance saver on Mobile.

To do this, just enable only the effect you want to render, set Sky Color, Horizon Color, Ground Color alpha to 0 and use the Cubemap Creator to export.

For Sun and Moon effect, it is important to set their **light source rotation to (0,0,0)**, or NULL before render to cubemap, so in Baked mode it can react correctly to light direction and Day Night Cycle.

Some example cubemaps are provided under **Assets/Jupiter/Demo/Cubemaps** folder.

PERFORMANCE TIPS

It is necessary to do some optimization to achieve high framerate, or 60 FPS on Mobile devices. Below are some tips for you:

- Bake it, don't simulate: Some effect such as Stars, Sun, Moon provide the Baked mode where it sample from a cubemap instead of procedurally render the element. See BAKING SKY TO CUBEMAP for more detail.
- Use as less module as you can: The more effect to render, the more time it cost.
- Disable Step/Banding effect: Step effect uses modulo (%) operation, which is heavy on Mobile, disable it by uncheck Utilities>Allow Step Effect.
- If you must use procedural stars, use as less star layer as possible.
- If you must use procedural sun/moon, it will render faster without texture.
- Use small/low resolution texture/cubemap, to reduce texture bandwidth.
- If you use Day Night Cycle, disable Real-time Environment Lighting in the Lighting window.
- If you don't need animated effect, just setup a super beautiful sky then render it into a cubemap and use it statically. See Use Cubemap as static skybox for more detail.

LIGHTWEIGHT/UNIVERSAL RENDER PIPELINE UPGRADE GUIDE

Jupiter also support for Unity Lightweight/Universal Render Pipeline. Note that render pipeline other than the built-in and LWRP/URP is not supported. It will work after installing the render pipeline, you don't need to do anything.

Minimum version for LWRP support: **Unity 2019.1**

Minimum version for URP support: **Unity 2019.3**