

Physically Based Night Sky

Introduction

Thank you for purchasing the asset “PBR Night Sky” for Unity! This asset will provide you with a simple way of adding realism to your night sky by allowing you to simulate the location of celestial bodies such as the sun, moon, planets, and stars within your Unity project’s sky.

Support for SRP

If you are using the built-in render pipeline (BIRP), you can skip over this section as the asset is set up for this pipeline by default.

Universal Render Pipeline (URP)

To get started with the URP version of the asset, navigate to the folder “*PBR Night Sky\SRP*” and import the Unity package labelled as “*URP*”. Import all the assets under this Unity package and then you can move onto the next section of this document.

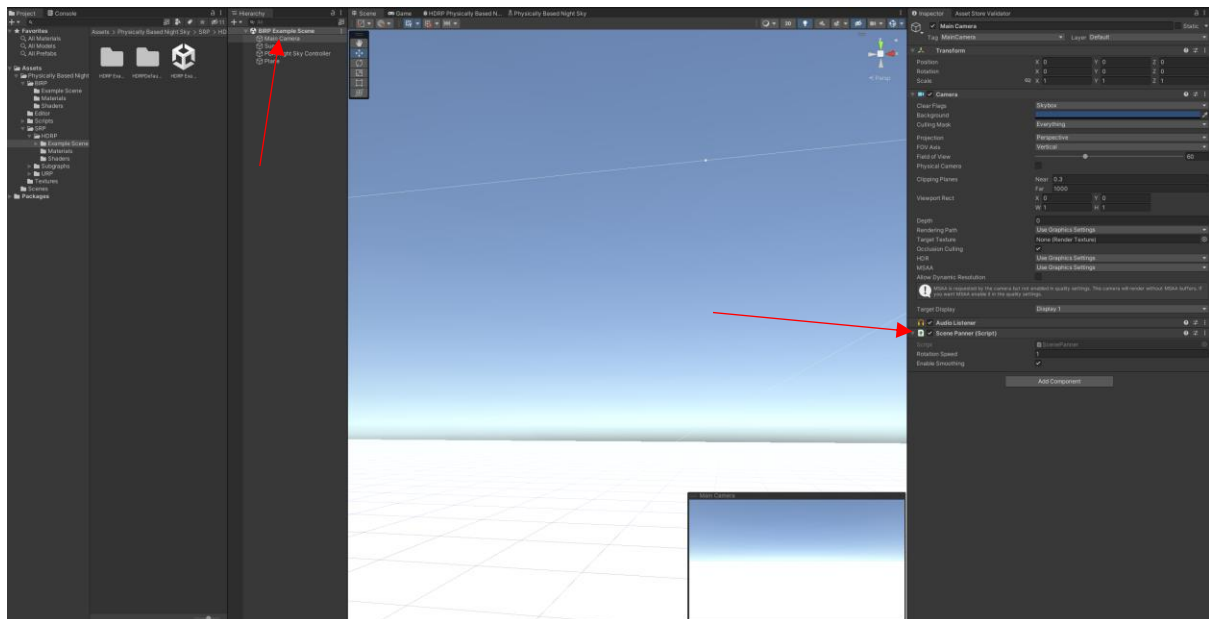
High-Definition Render Pipeline (HDRP)

To get started with the HDRP version of the asset, navigate to the folder “*PBR Night Sky\SRP*” and import the Unity package labelled as “*HDRP*”. Import all the assets under this Unity package and then you can move onto the next section of this document.

Example Scene

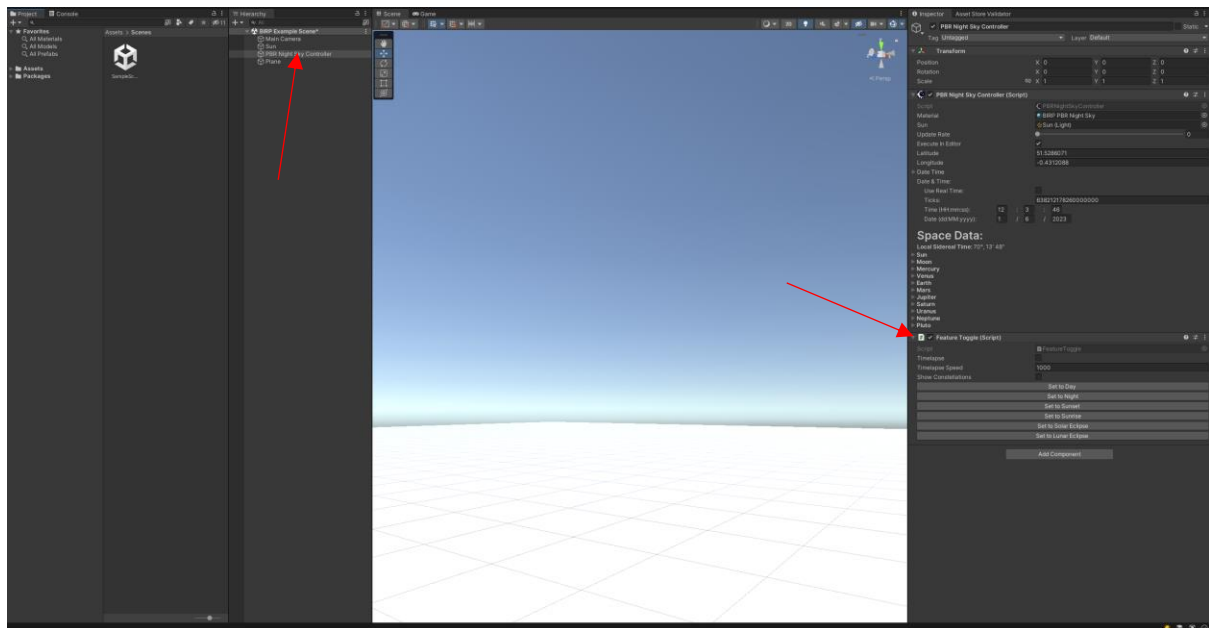
To get started, open the example scene that corresponds to the render pipeline that is in use. For the BIRP, this is located at “*PBR Night Sky\BIRP\Example Scene\BIRP Example Scene.unity*”, for URP it is located at “*PBR Night Sky\SRP\URP\Example Scene\URP Example Scene.unity*”, and for HDRP it is located at “*PBR Night Sky\SRP\HDRP\Example Scene\HDRP Example Scene.unity*”.

Once the scene is loaded, go ahead, and enter play mode. If you hold down the left mouse button and move the mouse around, you will be able to pan around and look at the sky. If the sensitivity is too high this can be changed in the inspector under the Camera Game Object, there is a script called “Scene Panner”. The sensitivity can be modified here as toggling whether the mouse movement is smoothed.



To customize the setup of the example scene, navigate to the “PBR Night Sky Controller” Game Object. There are two components that can be modified here. There is the main “PBR Night Sky Controller” component, which is the driver of the sky, and then there is the “Feature Toggle” component which is responsible for showcasing some features. To see how to customize the setup using the “PBR Night Sky Controller” component, please refer to the “Usage” section of this document.

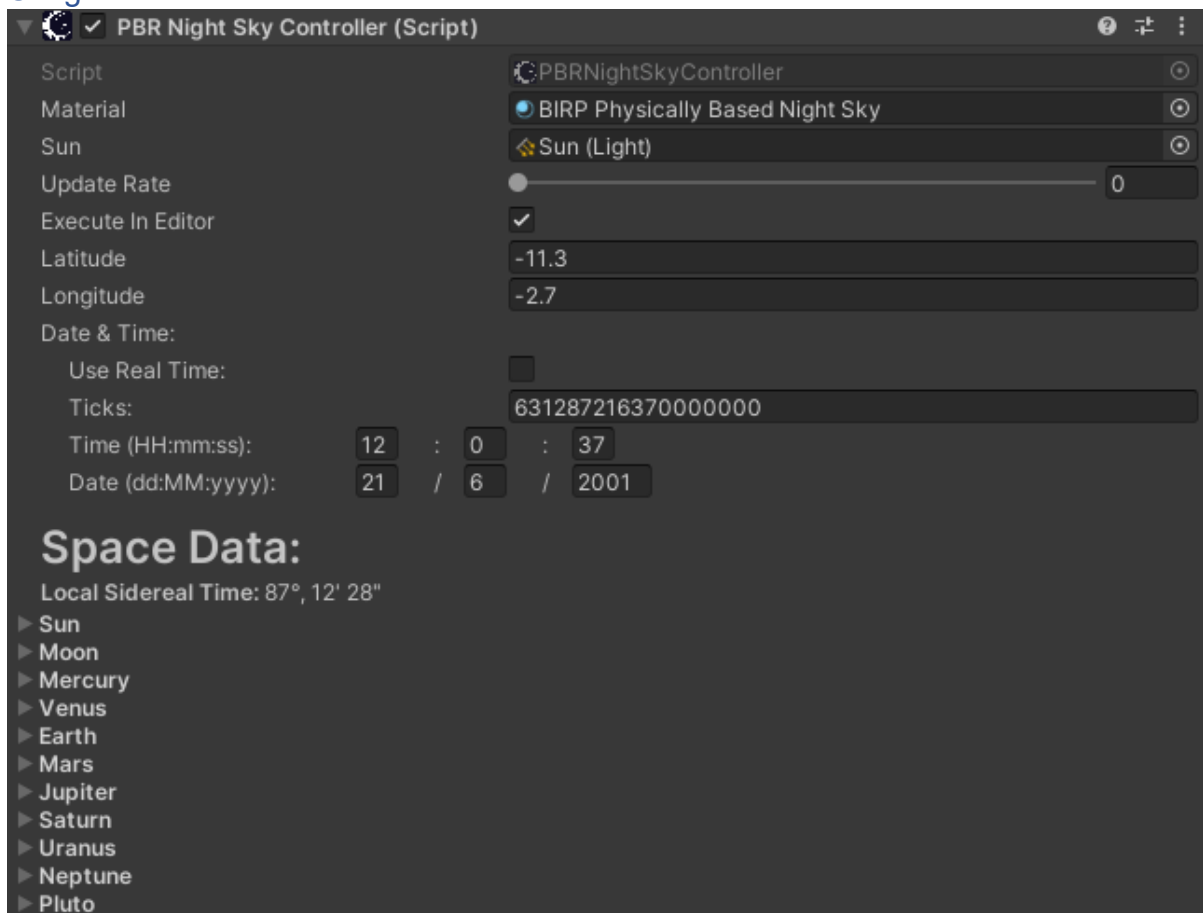
To use the feature toggle component, the timelapse toggle can be enabled to see the sky update in real-time, cycling through the day and night. If the cycle is too fast or too slow, the timelapse speed can be modified to adjust for this. There is the toggle for showing the constellations, this will show the major constellations in the night sky (so make sure that it is not day when looking at this feature!). This can also be toggled directly in the example shaders provided. Finally, there are some buttons that will set some presets of what you can expect from different times of day and phenomena such as a solar eclipse and lunar eclipse!



Setup

The setup for this asset is as simple as adding the “PBR Night Sky Controller” to a Game Object in your scene! Make sure that you assign all the fields in the inspector (please refer to usage section if any of these fields are not self-explanatory). This asset is intended to work with the default Unity skybox, but it will work with any skybox that draws the sun and does not draw the moon, planets, or stars.

Usage




1. Material – the material used to render the sky; this should be set to the material that is in the pipeline's folder that you are using.
2. Sun – The directional light that represents the Sun in the scene.
3. Update Rate – the rate at which to calculate and update the celestial data. This can be left at zero for real-time updating. However, especially when using the physically based skybox in HDRP, moving the sun can cause some large FPS drops, so changing this setting can help alleviate this.
4. Execute in Editor – toggles whether the celestial data should be calculated in the editor. If this is left unchecked, the moon, stars and planets will still be rendered, but they will not have their positions updated until play mode is entered.
5. Latitude – the geographical latitude on Earth that the player is observing the sky from.
6. Longitude – the geographical longitude on Earth that the player is observing the sky from.
7. Date & Time
 - a. Use Real Time – if toggled will use the local real time to update the sky.
 - b. Ticks – the System.DateTime ticks that the date and time currently represent. This can be used to easily progress time forwards or backwards without having to individually modify the date and time.
 - c. Time – the 24 hour time to use when calculating the positions of celestial bodies in the format Hour:Minute:Second.
 - d. Date – the date to use when calculating the positions of celestial bodies in the format Day/Month/Year.

Space Data

The space data shows some more technical data relating to what has been calculated to display the celestial bodies where they are.

1. Local sidereal time - https://en.wikipedia.org/wiki/Sidereal_time
2. For each celestial body, there are the following pieces of data:



▼ Sun
 RA: 16h, 19m 34s
 DEC: -21°, 26' 3"
 AZM: 246°, 1' 22"
 ALT: 60°, 48' 19"

RA (Right ascension) – https://en.wikipedia.org/wiki/Right_ascension

DEC (Declination) – <https://en.wikipedia.org/wiki/Declination>

AZM (Azimuth) & ALT (Altitude) –

https://en.wikipedia.org/wiki/Horizontal_coordinate_system