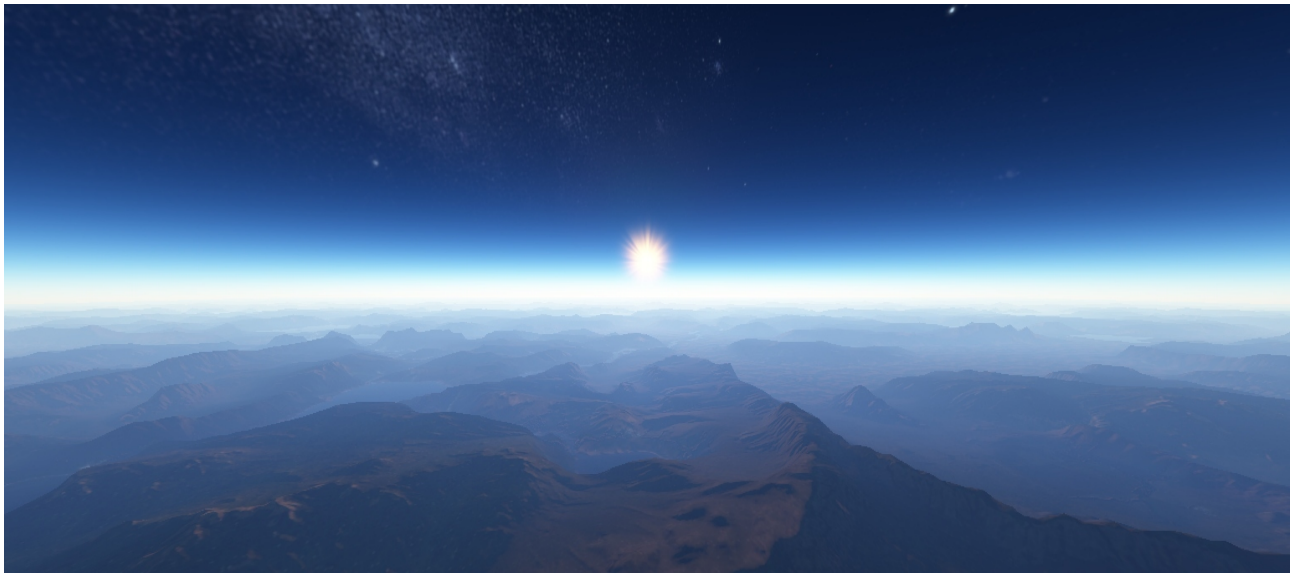


AZURE[SKY]

DYNAMIC SKYBOX

Document Version 4.1.0

Contents

INTRODUCTION

Welcome to Azure[Sky] Dynamic Skybox!

Azure[Sky] Dynamic Skybox is a powerful physically based dynamic sky system that will rise your project to another level. It was developed to be simple, fast and easy to use while keeping all the power and control in your hands.

Azure has two different types of sky models, giving you the possibility to use Azure in the most varied types of graphic styles, ranging from cartoon to the most realistic.

- **Standard Sky Model:** This sky model comes with a lot of customization options that allow you to create realistic and stylized scenes of the most varied types. This model has been used and improved in Azure since its creation and received a great focus on performance, you will hardly find in the Asset Store another option that offers you so much quality along with excellent performance. This model is indicated for those who want an excellent performance along with a great capacity of customization.

- **Precomputed Sky Model:** This sky model is the most realistic model of the present day, it uses precomputed data and gives you a result very similar to real life. Because it is more focused on reality, this sky model is based only on Earth's atmosphere and therefore has fewer styling options than the Standard model. This model is suitable for anyone who wants an extremely realistic atmospheric effect.

Azure[Sky] is easy to integrate into your project and its clean and organized interface makes learning very fast. Basically what you need to do is drag a prefab to your scene and give it your artistic touch.

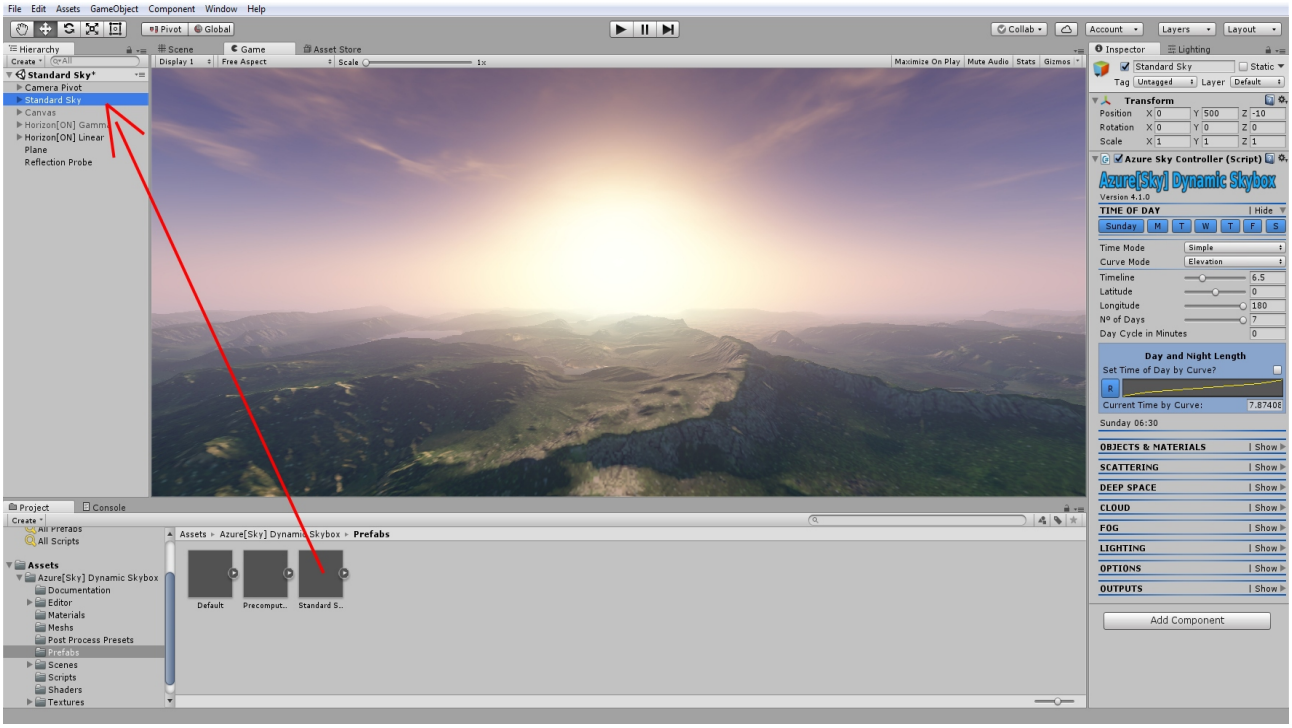
- *PS: English is not my native language, so I apologize for possible grammar errors.*

GETTING STARTED

After importing Azure[Sky] Dynamic Skybox to your project, all you have to do is drag one of the Azure prefabs to your scene.

- *Make sure there is no other directional light in the scene, because the prefabs come with directional lights of the sun and moon attached to it.*

- *Azure prefabs are located in the folder: Assets/Azure[Sky] Dynamic Skybox/Prefabs.*



(The terrain used in some images belongs to Horizon[ON] and is not included with Azure[Sky].)

Automatically Unity's default-skybox material will be removed from the scene to make way for the (skydome/skybox) of Azure[Sky].

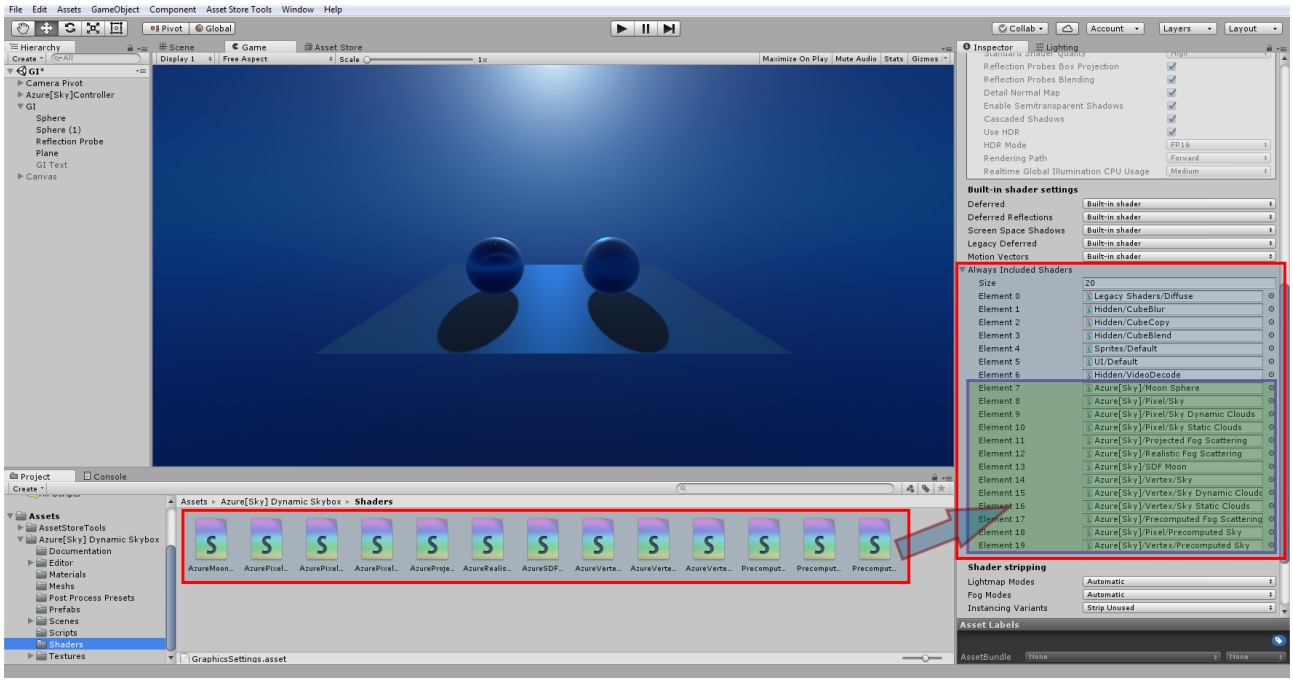
You can also choose to add the fog scattering effect on your camera by attaching the **AzureFogScattering.cs** script to it or by the menu: *Component/Azure[Sky]/Fog Scattering*.

The fog distance has been set to the values used in the showcase scenes, so do not worry if the fog does not appear in your scene or if it seems too far away, you may need to adjust the values on the Fog tab to suit your scene.

Azure uses different shader variations that will be applied to the materials depending on the settings you set in the Inspector. Unity includes in the build only the shaders referenced in the materials or in some script. For this reason, if you have two scenes that share the same sky and fog materials, but with different settings, an error in one of these scenes will undoubtedly occur after you build the project, because Unity included only the referenced shader in the material , this means that the shaders of one of these scenes will be missing in the build, and consequently the sky system will not work.

A very important step that can avoid this problem in the future is to add the Azure shaders in the "Always Included Shaders" array located in the menu: Edit> Ptoject Settings> Graphics> Always Included Shaders.

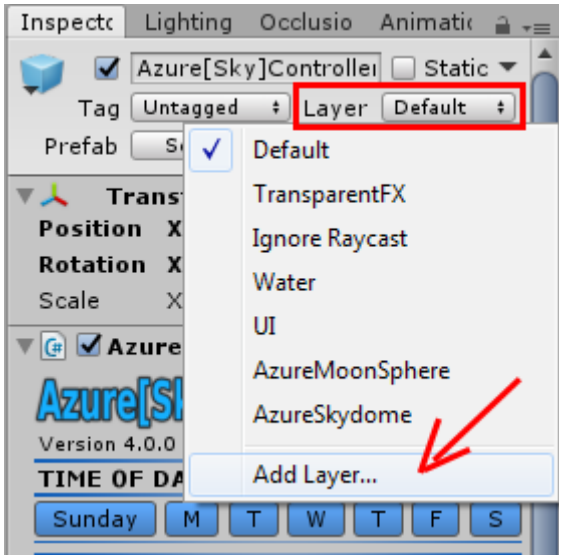
In this way, all shaders will be present in the project build and there will be no errors when switching from one scene to another that has a sky configuration that uses different shaders.



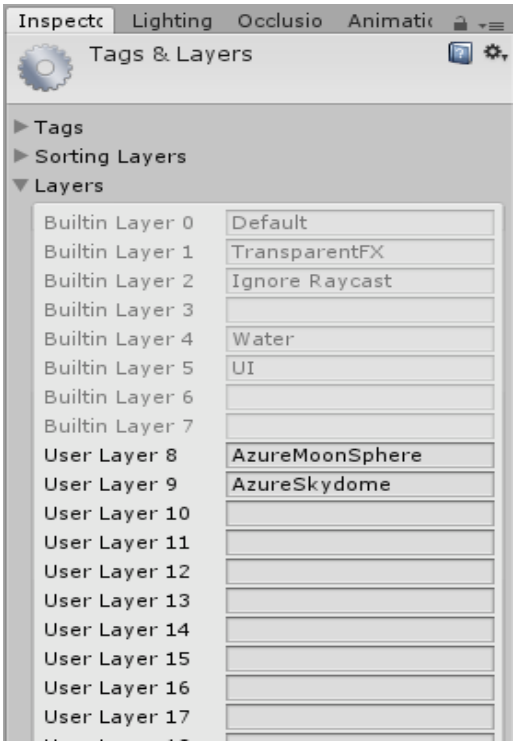
THIS IS THE POINT YOU NEED TO MAKE SOME SETTINGS AND CHOICES TO BETTER ADJUST AZURE TO YOUR PROJECT.

- First of all, you need to create two layers, one to set the sphere of the moon and another to set the skydome.

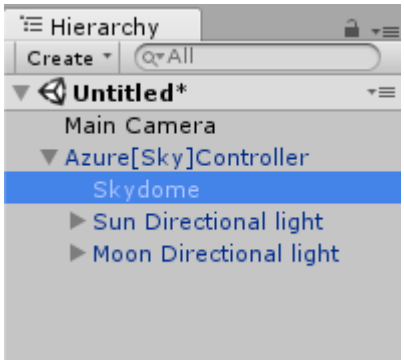
  So, select some GameObject in scene and in the Inspector select the **Layer** menu button and then click on **Add Layer...**

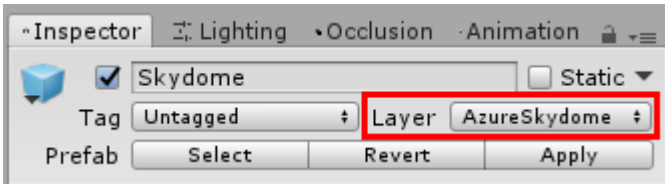


Then create the two layers as shown in the image below.
AzureMoonSphere
AzureSkydome



- You need to select the skydome inside the prefab and set it to the layer you just created.

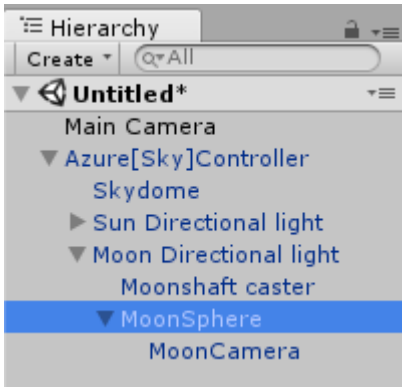


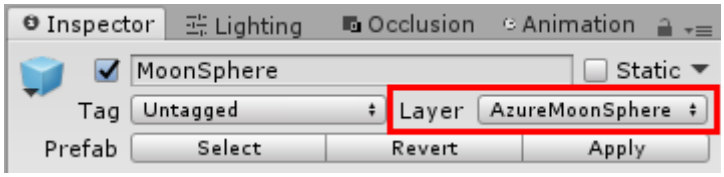


- *Thus, if you want the sky to be reflected in the scene, simply add a Reflection Probe involving the whole scene and in **Culling Mask** that reflection probe set to get only the **AzureSkydome** layer.*

Now let's define the layer for the moon sphere.

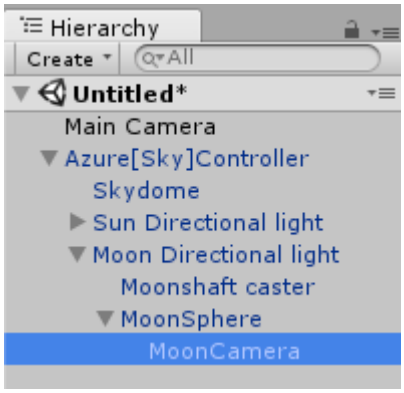
Select the moon sphere in the prefab and set the **AzureMoonSphere** layer for it.

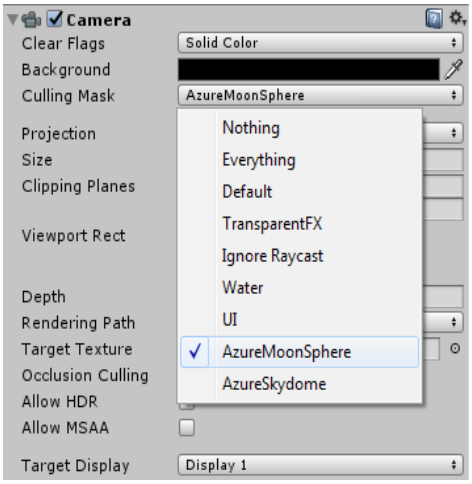




- Now let's say to the camera that renders the texture of the moon, to render only the moon sphere and no other object on the scene.

Select the **MoonCamera** inside the prefab and in **Culling Mask** define only the **AzureMoonSphere** layer.





- *MoonCamera is responsible for rendering the texture of the moon with the correct phase relative to the position of the sun, this Render Texture is stored in the AzureMoonRenderTexture located inside the textures folder. That's why it's very important that MoonCamera render only the Moon's sphere and nothing else.*

- *Since the Moon's sphere is a 3D object in the scene, it's very important that you deactivate the **AzureMoonSphere** layer from your other cameras so that the 3D moon sphere does not appear in your scene during the gameplay.*

Note that by default, Azure is using layers number 8 and 9 to store the keywords of AzureMoonSphere and AzureSkydome. If these layers are already being used in your project, you must setting again the demonstration scenes and prefabs by repeating the previous steps using another layers, because perhaps in your project these layers are used for other purposes.

- You also need to decide if you want the sky system to work as a Unity's standard skybox or as a skydome. To do this, simply set the chosen option in the **Sky Mode** property on the Options tab.

  Skydome: This option activates a mesh around the scene and removes any material defined as skybox in the Lighting window. The biggest thing about using a skydome is that most calculations in the shader can be done by vertex and this gives us a considerable performance gain. We can also control the resolution of the skydome mesh, which is not possible to do with the mesh of the Unity skybox. The downside is that you will not be able to benefit from Global Illumination based on the skybox as this is only possible if you use a skybox instead of the skydome. If you use skydome mode, set **Shader Mode** to **Vertex** to gain performance.

  Skybox: This option disables the skydome and automatically sends the sky material to the skybox property in the Lighting window. The good thing about using this option is that you will be able to enjoy Skybox-based Global Illumination. The downside is that the mesh of the Unity skybox is very low resolution and this causes many artifacts in the sky if you use Shader Mode as Vertex. It is advisable to set the **Shader Mode** to **Pixel** if you are using the sky as a skybox, so the performance may not be as good as the Vertex option.

- You may also want to choose between the two time modes available in the Time of Day tab, the **Simple** mode and the **Realistic** mode.

  Simple Mode: In the simple mode the sun and moon will rise and sets always in the same schedules. And the moon will always be on the opposite side of the sun.

  Realistic Mode: In realistic mode the sun and moon will be realistically positioned according to the time, date and geographical location. The phases of the moon will automatically adjust depending on the position of the moon and the sun in the sky.

- In addition to the time mode, two modes of curves and gradients are also available in the Time of Day tab. The curve and gradient mode based on the **Timeline** and the curve and gradient mode based on the **Elevation** of the sun and moon in the sky.

  Timeline mode: All values of curves and gradients will be based on the time of day between 00h and 24h. This mode works best with the time mode set to Simple.

  Elevation mode: All values of curves and gradients will be based on the elevation of the sun/moon in the sky between -1 an +1. This mode works well with the two time modes.

Another great feature that has been available since version 4.1.0 is the option to set a sky model in the Scattering tab.

NOW YOU ARE READY TO START CUSTOMIZING YOUR SKY.

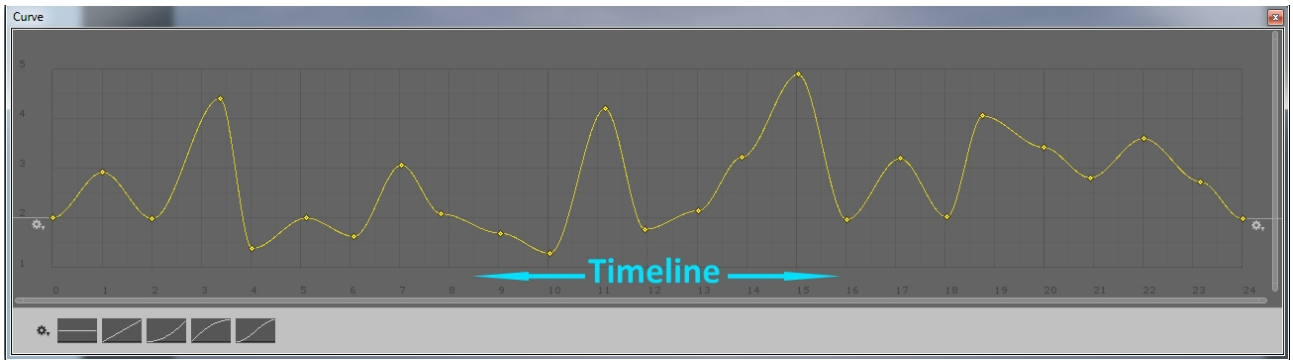
ABOUT THE CURVE SYSTEM

Unlike other sky systems, instead of sliders, the properties are configured by curves and gradient colors. With this innovative system you will be able to determine the best possible setting at any time of the day.

Two curve modes are available for you to choose in the Time of Day tab.

Timeline

All values of curves and gradients will be based on the time of day between 00h and 24h. This mode works best with the time mode set to Simple.



- *You need only click on the property field in the Inspector to open the curve editor.*
- *On the horizontal axis are the hours of the day between 00-24. On the vertical axis is determined the property value at each time of day. When the time of day changes, the property value is updated to the value that is in the curve at that time. You can create as many keys you want, no need to create a key for each hour if this is not necessary.*
- *To create smooth transitions, simply start the day with the same values as the previous day ended.*

Elevation

All values of curves and gradients will be based on the elevation of the sun/moon in the sky between -1 an +1. This mode works well with the two time modes.

In Simple time mode if you set the curve mode to Timeline, the sun/moon

rises and sets always at the same time of day. For example: The sun always rises at 6h and always sets at 18h, so the setup you set in the curves and gradients for that particular time will always match the exact position of the sun and the moon in the sky.

But when you set the time mode to Realistic. Depending on the geographical location and date, the sun and moon will be in different positions and the setup you set in the curves and gradients will no longer match with the position of the sun and moon in the sky.

For example: If you set a value in curves and gradients to the sunset(at 18h) and then change the date or location, the sunset will no longer be at 18h, and may be at 20h, 21h, etc... It will depend on the location that you set. Then the value that you previously set for sunset at 18h, will be affecting a different time of day and no longer the sunset.

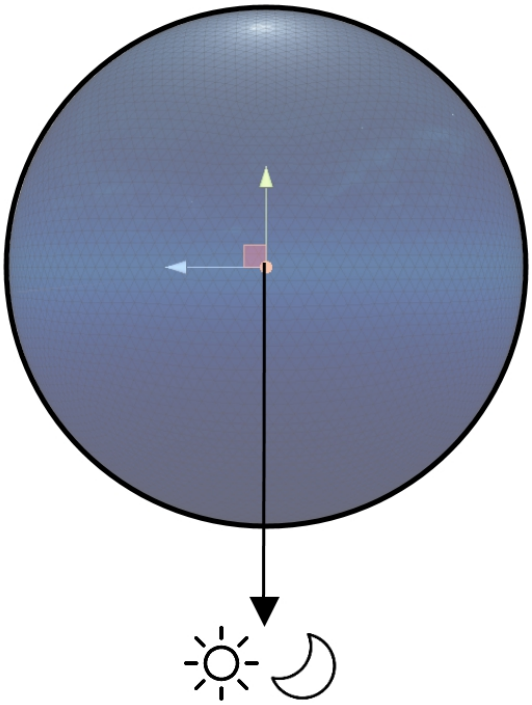
For this reason, when the time mode is set to Realistic, the curves and gradients will not be adjusted in line with the time of day, instead you need to set the curve mode to Elevation for the values of the curves and gradients to match with the elevation of the sun and moon in the sky. So you will for example, set the intensity of the directional light of the sun based on the altitude where the sun is in the sky. You set a value for when the sun is rising on the horizon and another value for when the sun is at the top of the sky or below of horizon line. As the day progresses, the value changes smoothly depending of the altitude.

In this curve mode, instead of hours, we will use the elevation of the sun or the moon*(will depend on the property you want edit)* in the sky to control the value of each property. So the new timeline will be between -1 and +1 instead of 00h and 24h.

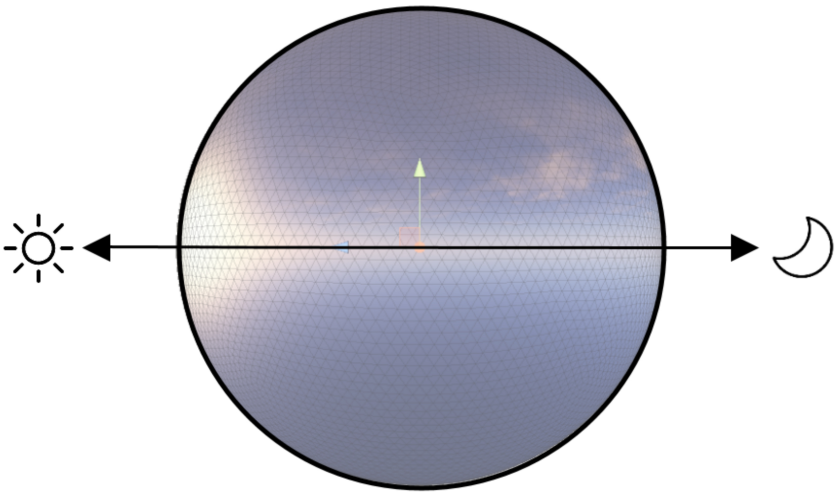
See the image below

- *On the horizontal axis are the elevation of the sun/moon between -1 and +1. On the vertical axis is determined the property value at each elevation. When the sun/moon altitude changes, the property value is updated to the value that is in the curve at that elevation.*

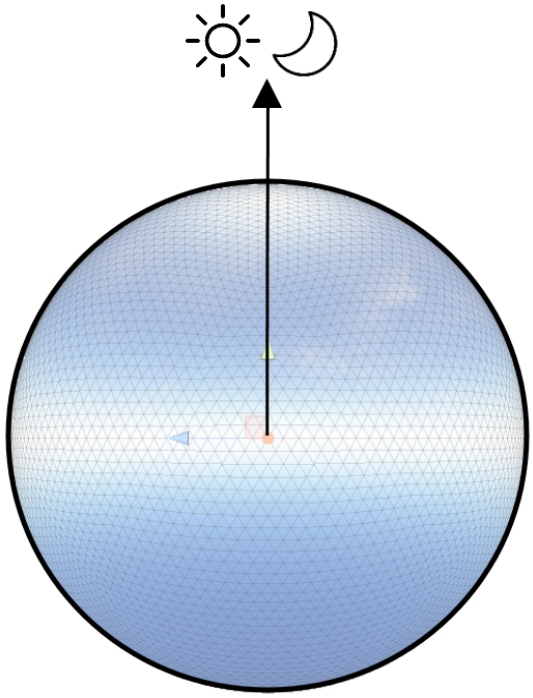
Elevation -1: When the sun or moon is positioned at the lowest height of the sky. At 270º.



Elevation 0: When the sun or moon is positioned exactly at the height of the horizon line. At 0º or 180º.

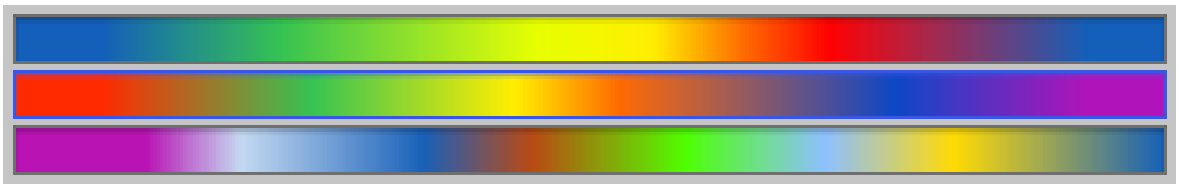


Elevation +1: When the sun or moon is positioned at the top of the sky. At 90º.



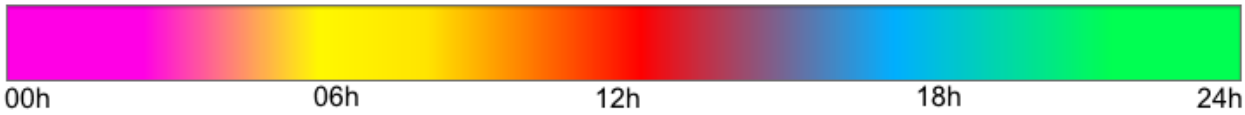
ABOUT THE GRADIENT COLORS

Instead of using the same color for the entire cycle of the day, Azure uses gradient colors to increase the customization power of your game. This way you can set different color values for different times of the day. Even ambient and lighting colors can now be set to a new level that Unity's default mode does not allow.

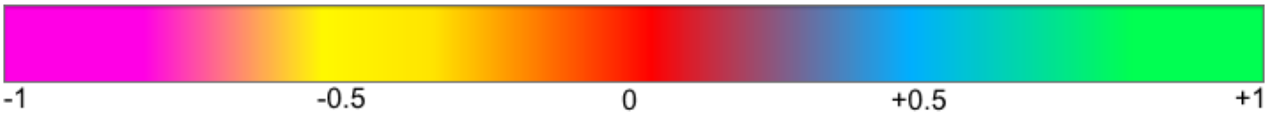


The way to customize the gradients also varies according to what was defined in **Curve Mode**.

In the Timeline curve mode, each location in the gradient represents one hour of the day.

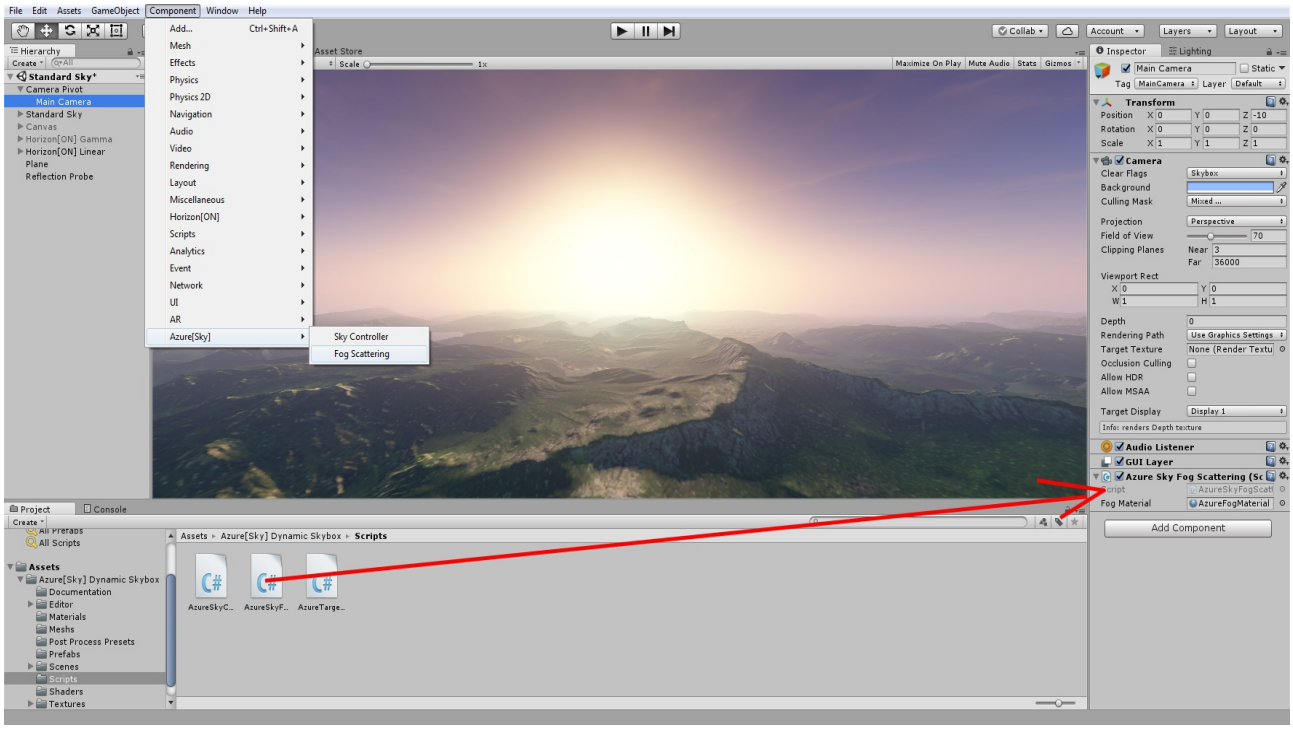


In the Elevation curve mode, each location in the gradient represents one altitude of the sun/moon in the sky.



ABOUT THE FOG SCATTERING

Fog Scattering is an image effect that you need to attach to your camera. To activate the effect you can simply drag the "AzureSkyFogScattering.cs" script into the Inspector of the camera or via the menu "*Component>Azure[Sky]>Fog Scattering*".



- *It is advised to disable "Anti Aliasing" in quality settings, as this interferes in a variety of image effects and can cause some artifacts in the Fog Scattering effect. Ideally, use the "Anti Aliasing" through image effect.*

- *If the fog behaves strangely with transparent objects, you may want to uncomment the line with the **[ImageEffectOpaque]** command in the AzureSkyFogScattering.cs script.*

- *The order of the image effects in the camera Inspector can change the result of the image. It is advised to attach the fog scattering before the other image effects.*

- *The fog will affect only objects that draw to the depth buffer. Usually the objects that are in the "Transparent" queue do not draw to the depth buffer.*

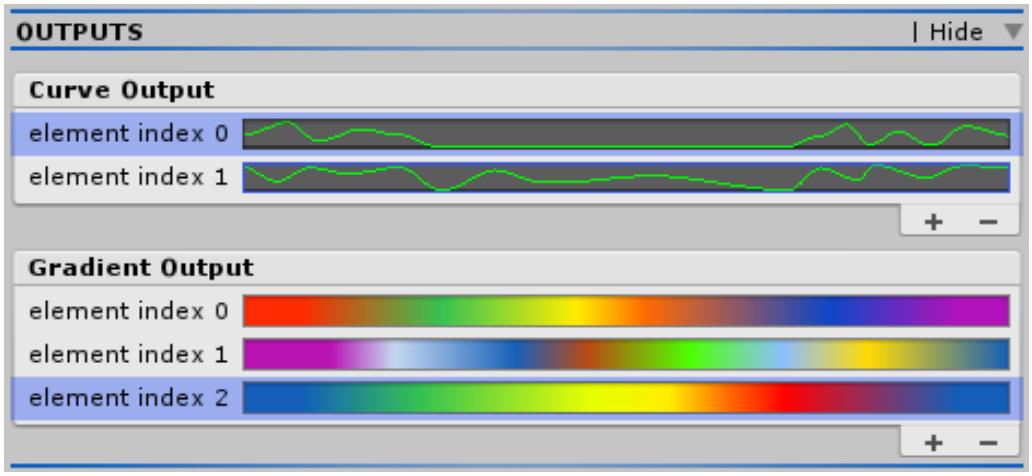
In the Standard sky model you can switch between two fog modes. The **Projected** mode and the **Realistic** mode.

Projected Mode: The sky background is projected over the scene.

Realistic Mode: This method works more correctly than the projected mode, and the fog colors change from blue to white based on the distance of objects from the camera.

ABOUT THE OUTPUT SYSTEM

With the Output System you can control virtually all elements of your game that need to be changed according to the time of day. This is an innovative and very useful tool that lets you control from the fx sounds to the properties of materials and lights in the scene.



- *As you can see in the figure above. As an example I created two "Outputs" of curve type and three "Outputs" of gradient type. You can create and delete as many "Outputs" you need.*

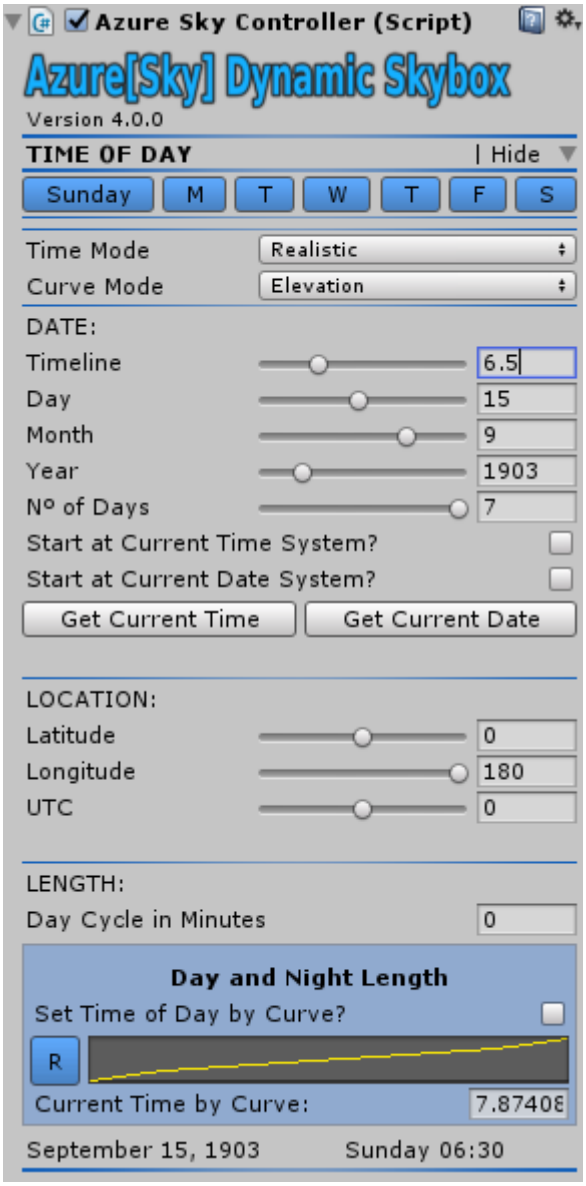
- *To use the "Output System" although it is very simple, it is recommended that you understand the basics of scripts programming.*

- *To access the value based on the timeline/elevation of the "Curve Output" just use the method "**AzureGetCurveOutput(int index, int curveMode)**".*
  **In the "index" you must put the number of the element of "Curve Output" that you are accessing.*
  ** In the "curveMode" you must put if you want to get the curve based on timeline=0, sun elevation=1 or moon elevation=2.*
  *This will return a value of type "**float**" that you can convert in "**integer**" if necessary.*

- *To access the value based on the timeline/elevation of the "Gradient Output" just use the method "**AzureGetGradientOutput(int index, int curveMode)**".*
  **In the "index" you must put the number of the element of "Gradient Output" that you are accessing.*
  ** In the "curveMode" you must put if you want to get the gratient based on timeline=0, sun elevation=1 or moon elevation=2.*
  This will return a Color.

TIME OF DAY TAB

This tab has the properties related to the day/night cycle. Here you will set the date, time and location, also select the day of week you want to customize.



At the top are the buttons of the days of the week, each day will store a different customization and you can switch between them just by clicking each button.

Below are the time and curve modes already mentioned previously. If you set the time mode to Realistic, more options will be available in the tab for you to customize.

Timeline: *This slider is used to control the timeline, do not confuse the timeline with the hour of day. If "**Set Time of Day by Curve**" is disabled, the "hour" of day will reflect the value set here, otherwise it will only represent the cycle of day and the actual hour of day will be displayed on the property "**Current Time by Curve**". The scene will start on time that is set here in this slider.*

Day: *The day of the month. Each month will have the correct number of days for you to select, including February which will have 29 days in leap years.*

Month: *

Year: *The year you want to use. You can set a value between 0000 and 9999.*

Nº of Days: *Here you define the number of days used in the weekly cycle. If you set 3 days for example, after finishing the third day, will return to the first. If you always want to repeat the first day, leave it with a value of 1.*

Start at Current Time System: *If enabled, it will start the scene with the current system time.*

Start at Current Date System: *If enabled, it will start the scene with the current system date.*

Get Current Time Button: *When pressed, it applies the current system time to the Inspector properties.*

Get Current Date Button: *When pressed, it applies the current system date to the Inspector properties.*

Latitude: *This property is used to control the latitudinal coordinate to indicate different locations on Earth.*

Longitude: *This property is used to control the longitudinal coordinate to indicate different locations on Earth.*

UTC: *Coordinated Universal Time*

Day Cycle in Minutes: *Here is where you will set the duration of the cycle of day in minutes, do not confuse with the speed that daytime and night will pass. This sets the time it will take for the sun to complete a full circle around the scene.. For the time stay static just set this value to zero.*

Set Time of Day by Curve: *Enabling this option you set the speed that the daytime and the night will pass based on the curve that can be edited in the field below. For example, if you set the duration of the "day cycle" to 15 minutes, you*

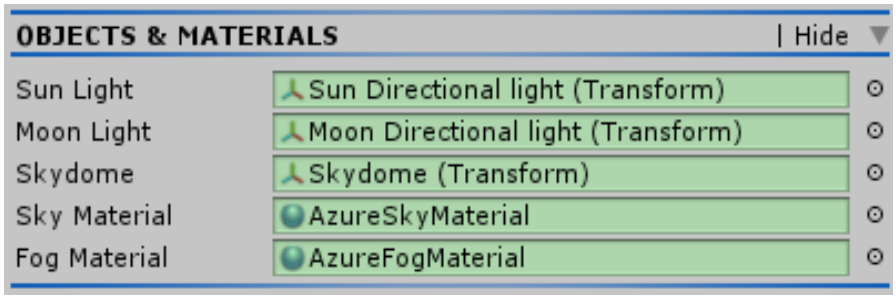
*can edit the curve to make the daytime last 10 minutes and the night last 5 minutes. Note that the total duration of the “day cycle” will still be 15 minutes. By changing the length of daytime and night, the hour of day will not match with the value of the "**Timeline**" slider. The correct hour of day will be displayed on the property "**Current Time by Curve**".*

Current Time by Curve: *Just to show the correct hour of day if the "**Set Time of Day by Curve**" option is enabled.*

At the bottom shows in text format the date and time converted correctly.

OBJECTS & MATERIALS TAB

Here in this tab are the fields of objects and materials that need to be referenced to the sky system to work properly. It should already be set by default, so there is no need to change anything here.



SCATTERING TAB

Here in this tab are the properties that make up the mathematical equations used to generate the sky. And it is also where you can choose the sky model you want to use.

Standard Sky Model:

Sky Model: *The sky model you want to use.*

Wavelength: *Defines the wavelength or informally speaking "sets the color of the sky". In Precomputed mode the wavelength is already pre-set to a default value. In Real-Time mode, the wavelength is calculated based on the color you set in lambda(color gradient that will appear just below). Note that in real-time mode, it requires more processing, use precomputed mode for better performance.*

Rayleigh: *https://en.wikipedia.org/wiki/Rayleigh_scattering*

Mie: *https://en.wikipedia.org/wiki/Mie_scattering*

Kr: *To calculate the Rayleigh optical depth.*

Km: *To calculate the Mie optical depth.*

Sun Intensity: *It controls the intensity of sun brightness and the sun disck intensity in the skybox. Do not confuse with the intensity of the sun directional light used to illuminate the scene.*

Rayleigh Color: *
Mie Color: *

Light Speed: *This property is very interesting because it controls how fast the light will propagate in the scene after the sun or moon crosses above the horizon line. This property controls both the initial intensity of the directional light of the sun and the moon, as well as the intensity of the mie and moon's brightness in the sky.*

For example, if you set the value of this variable to zero, the mie intensity, the moon brightness, and the directional light intensity of the sun and moon will illuminate the sky and the scene instantly when the sun or moon crosses above the horizon line, using the values that you set in the curves.

But if you set the value of this property to the maximum, the mie intensity, the moon brightness, and the intensity of the directional light of the sun and moon will slowly illuminate the sky and the scene with a smooth transition to the values you set in curves when the sun or moon crosses above the horizon line.

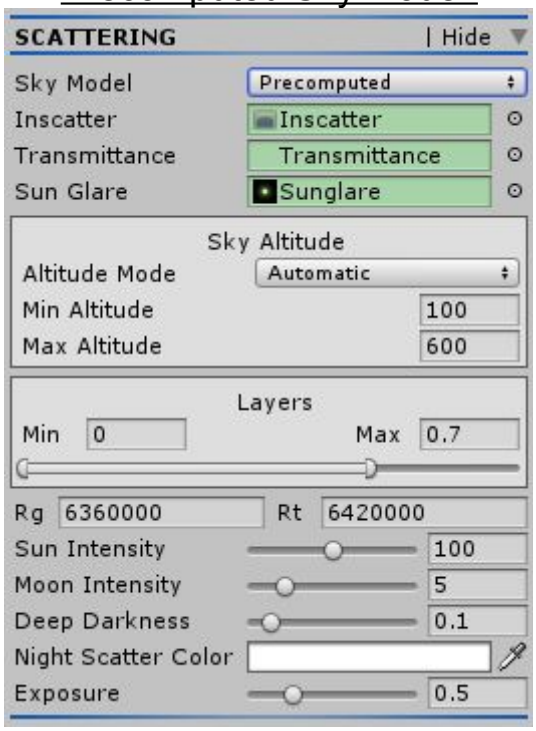
Day Darkness: *Controls the darkness of the blue color of the day sky.*

Night Intensity: *The intensity of the night brightness.*

Exposure: *Controls the exposure of the tonemapping that is performed internally in the shader.*

Sun Disk Size: *
Sun Disk Propagation: *

Precomputed Sky Model:



Inscatter: *Precomputed 3D table.*
Transmittance: *Precomputed 2D table.*
Sun Glare: *Sun Glare 2D texture.*

Sky Altitude:

Altitude Mode: *Always leave this option automatic when building the project.*

In automatic mode the altitude of the sky will change automatically according to the altitude of the active MainCamera in the scene, so make sure you have an active camera in the scene with the Tag MainCamera. Sky altitude variations will occur when the camera is between the minimum and maximum altitude set in "Min Altitude and Max Altitude".

In the manual mode, a slider will appear for you to take a quick test and see if the altitude changes are working well. So when you move the slider, the camera's altitude will change based on the minimum and maximum altitude you set in "Min Altitude and Max Altitude".

Min Altitude: *Set here the minimum value that the camera altitude will affect the sky, when the camera is below the minimum value, the sky will look at sea level regardless of the minimum altitude you set.*

Max Altitude: *Set here the maximum value that the camera altitude will affect the sky, when the camera is above the maximum altitude value, then you will be out of the atmosphere, practically in outer space.*

Layers: *In Layers you can clip a minimum and maximum value of the altitude that the sky will be able to reproduce. For example, if you set the **max** layer to 0.5, when the camera exceeds the maximum altitude you set in Sky Altitude, the atmosphere will not represent that it is in outer space, but rather that at medium altitude, and no matter how high the camera stays in the sky, the atmosphere will never go beyond that apparent altitude.*
*If you set the **min** layer to 0.5, then the camera may be at sea level, but the atmosphere will represent that you are still at medium altitude.*

Rg and **Rt** are constant values used in the mathematical formulas of the sky, it is recommended that you do not change these values.

Sun Intensity: *
Moon Intensity: *

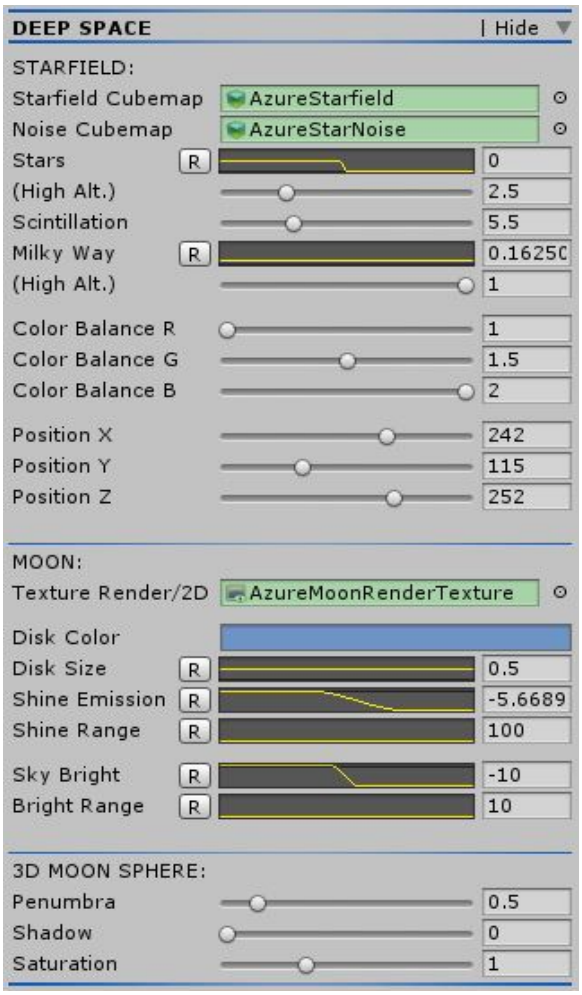
Deep Darkness: *It is useful for controlling the darkness of the night when there is no moon in the sky. Keep a low value if there is the moon in the sky, or increase the value so that the night does not go completely dark when there is no moon in the sky.*

Night Scatter Color: *Night color multiplier.*

Exposure: *Controls the exposure of the tonemapping that is performed internally in the shader.*

DEEP SPACE TAB

On this tab are the controls of the outer sky elements.



Starfield Cubemap: *The star field cubemap.*
Noise Cubemap: *The noise cubemap to apply the scintillation of stars.*

Stars: *Controls the intensity of the entire starfield, including the Milky Way.*
(High Alt.): *Stars intensity when in maximum altitude.*

Scintillation: *The speed of the stars scintillation.*

Milky Way: *Controls the intensity only of the Milky Way, with this property you can remove the Milky Way and keep only the main stars in the sky.*
(High Alt.): *Milky Way intensity when in maximum altitude.*

Color Balance: *Gently apply a color tone to the entire star field.*

Position: *Relative position of the star field in the sky.*

Moon Texture2D: *Here you put the texture of the moon, you can set a RenderTexture or a simple texture. For best performance, it is advisable to define a simple texture that is available inside the Textures folder and to delete the moon sphere along with the MoonCamera inside the prefab.*
PS: *Keep in mind that if you use a simple texture, the phases of the moon will not change according to the relative position of the sun, as this is currently only possible using a RenderTexture.*

Disk Color: *The color of the moon disk and also the color of the Shine Emission.*

Disk Size: *The size of the moon disk.*

Shine Emission: *The intensity of the characteristic brightness in front of the moon disk.*

Shine Range: *The range of the Shine Emission.*

Sky Bright Color: *The color of the moon bright in the sky.*

Sky Bright: *The intensity of the moon bright in the sky.*

Bright Range: *The range of the moon bright in the sky.*

Penumbra: *The hardness of the shadow of the 3d moon sphere.*

Shadow: *The shadow intensity in the 3D moon sphere.*

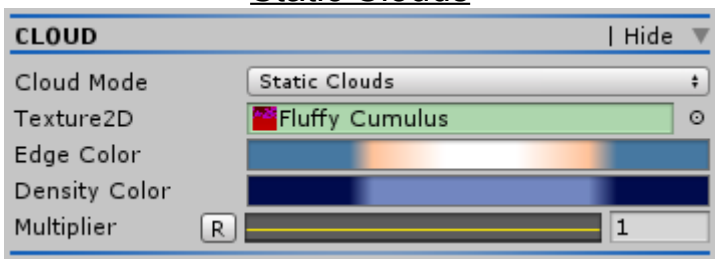
Saturate: *Color saturation of the 3D moon sphere.*

CLOUD SETTINGS TAB

In this tab are all the customization properties of clouds.
You just need to choose the cloud mode that you wish to use and customize.

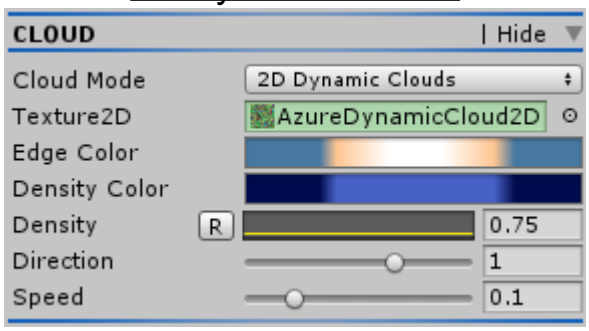
Cloud Mode: *In this option you will choose the cloud mode that you want to use. For now, only 2D dynamic clouds, static clouds and empty sky (without clouds) are available.*

Static Clouds



Texture2D: *The texture 2d map of the clouds.*
Edge Color: *The color of the edge of the clouds.*
Density Color: *The color of the dense regions of the cloud.*
Multiplier: *Cloud color multiplier factor.*

2D Dynamic Clouds



Texture2D: *The 2D noise texture to create the clouds.*
Edge Color: *The edge color of the clouds.*
Density Color: *The color of the dense regions of the cloud.*
Density: *Density of the cloud.*
Direction: *Direction of the moving clouds.*
Speed: *Speed of the moving clouds.*

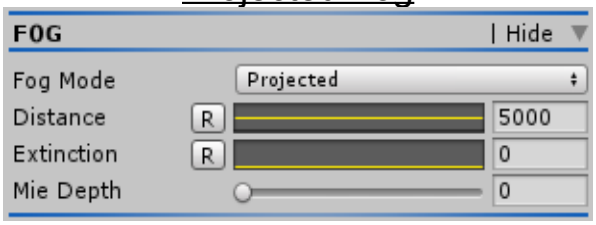
Note that in the Precomputed sky model, clouds are not yet available.

FOG TAB

Here in this tab are all the Fog Scattering control properties. If you do not want use the Azure fog scattering and only use the Unity's fog, just do not attach the script "AzureSkyFogScattering.cs" for your camera.

Fog Mode: *In this option you will choose the fog mode that you want to use. There are Projected and Realistic fog mode available.*

Projected Fog



Distance: *The distance in meters the fog will start.*
Extinction: *How much fog will cover the objects in the scene.*
Mie Depth: *Great for Closer Scenes, you can set the distance at which the mie brightness will appear, you can avoid the undesirable effect of the mie that seems to be crossing the objects very close to the camera when the fog is set to near.*

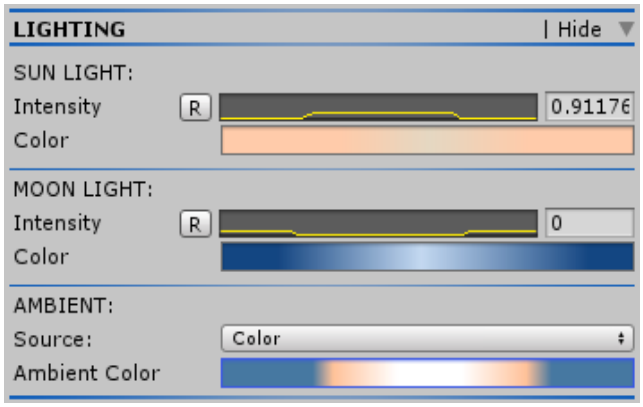
Realistic Fog

Distance: *The distance in meters the fog will start.*
Scale: *The fog scale factor. When the far clipping plane of your camera is too large, it is recommended to set a smaller value on this property for better results. And when your camera's far clipping plane is smaller, a higher value might be better at this property.*
Extinction: *How much fog will cover the objects in the scene.*
Mie Depth: *Great for Closer Scenes, you can set the distance at which the mie brightness will appear, you can avoid the undesirable effect of the mie that seems to be crossing the objects very close to the camera when the fog is set to near.*

Note that in the Precomputed sky model, the fog does not work by distance but by density.

LIGHTING TAB

In this tab you will be able to set the lighting and ambient colors of your scene.



Sun Light Intensity: *The intensity of the sun directional light in the scene.*
Sun Light Color: *The color of sun directional light.*

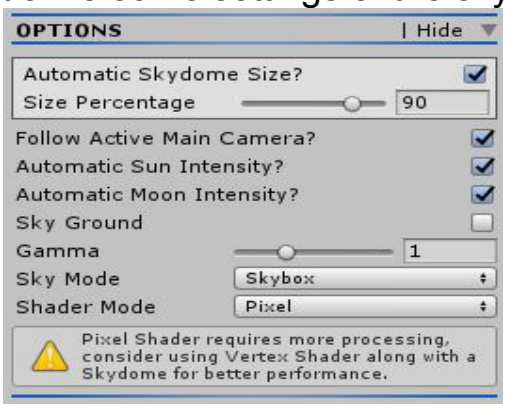
Moon Light Intensity: *The intensity of the moon directional light.*
Moon Light Color: *The color of moon directional light.*

Ambient

In the ambient session, you will set the environment lighting mode and colors the same as you would in the standard Unity way. The difference here is that you can use gradient colors, which is not possible in traditional mode.

OPTIONS TAB

In this tab you will define some settings of the sky system behavior.



Automatic Skydome Size: *If this option is enabled, the skydome will change its size automatically according to the far clipping plane of the active MainCamera with the relative percentage that you set in the Size Percentage property. Set a smaller percentage to avoid the possibility that some vertex of the skydome mesh will be beyond the view of the camera and can not be rendered. If you do not want the skydome to change its size automatically, then uncheck this option that will show a field for you to set a fixed-scale value for the skydome.*

Follow Active Main Camera: *If this option is enabled, the sky controller (GameObject) will always be in the same position as the active MainCamera. This is very important, especially if you are using the sky as a skydome. The sky must always be centered with the position of the MainCamera so that the effects in the sky coincide with the direction of the directional lights of the sun and moon in the scene and also with effects of sun shafts and lens flare.* ***So it is very important that you have an active camera in the scene with the MainCamera Tag, otherwise the sky controller will not update to the camera position.***

Automatic Sun Intensity: *When enabled, this option automatically changes the intensity of the directional light when the sun is crossing the horizon line. When the sun is setting, the intensity of the sun's directional light will be softly set to zero. When the sun is rising, the intensity of the sun's directional light will be set smoothly to the value set in the curvefield.*

Automatic Moon Intensity: *

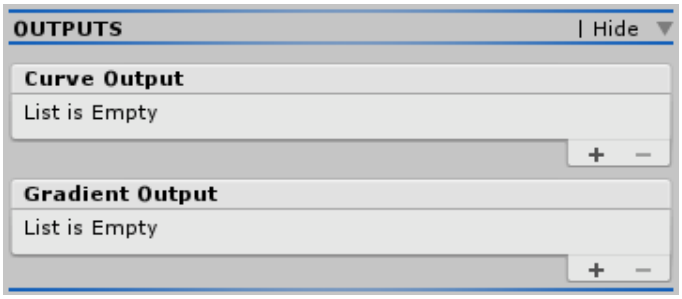
Gamma: *Gamma correction.*

Sky Mode: *Select the sky mode between Skybox and Skydome.*

Shader Mode: *Choose whether you want the calculations in the shader to be done by pixels or by vertices. Vertex calculations are faster, but can produce artifacts if the sky mesh is with low resolution like the Unity's standard skybox mesh. If you are using* ***Sky Mode*** *as* ***Skybox****, then it is advisable to set this option to* ***Pixel****.*

OUTPUT TAB

In this tab is where you will create the outputs of curves and gradients to control the elements of your game that need to be modified according to the time of day.



Now you can select whether the Output will be based on the timeline, or on the elevation of the sun and moon in the sky.

Note that if you want Output to be based on the timeline, you need to customize the curve and gradient by taking into account the hours between 0 and 24.

If you want Output to be based on the elevation of the sun or the moon, you need to customize the curve and gradient by taking into account the elevation between -1 and +1.

USING THE OUTPUT SYSTEM

Sometimes you will need to change the properties of an element of the scene based on time of day, such as turn on the lights of a city at night and turn off at daytime or change some value or color of any material. For this was created a Output System that gives the option to add extra color gradients and curves that can be accessed by other scripts.

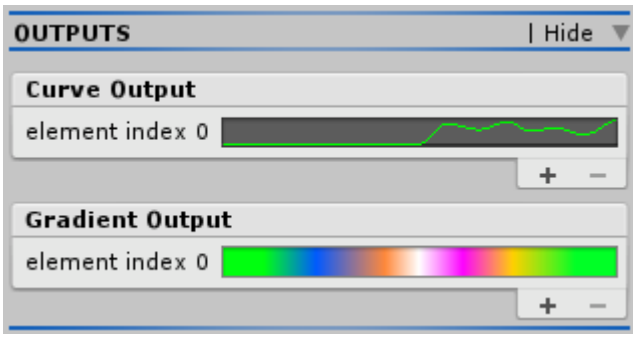
In this example I will show how to change the color and intensity of a "Point Light" using the Output System.

First you need to create a new scene.

- *Delete the default directional light;*
- *Add the prefab "Azure[Sky]Controller" to the scene;*
- *Add a "Point Light";*

Now let's create the Outputs.

- *Select in the scene the Azure[Sky]Controller prefab.*
- *Open in the Inspector the "Output" tab.*
- *Create a curve Output and set the values to be based on the moon elevation between -1 and +1.*
- *Create a gradient Output and set the values to be based on the timeline between 00h and 24h.*



The curve output will be to set the light intensity and the gradient output will be to set the color of the point light.

Now we need to get the values of the curve and gradient and send it to the "Point Light" properties.

- *Create a C# script.*
- *Attach this script to the "Point Light".*

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class OutputExemple : MonoBehaviour
{
    //Drag in the Inspector the AzureSkyController.
    public AzureSkyController azureOutput;

    //To store the Light component
    private Light myLight;

    // Use this for initialization
    void Start ()
    {
        //Store the Light component to use later.
        myLight = GetComponent<Light> ();
    }

    // Update is called once per frame
    void Update () {
        //curveMode = 0; Based on the Timeline.
        //curveMode = 1; Based on the Sun Elevation.
        //curveMode = 2; Based on the Moon Elevation.

        //Getting element 0 of "Curve Output" in "Azure Inspector".
        //This Curve Output will be based on the Moon Elevation.
        myLight.intensity = azureOutput.AzureGetCurveOutput(0,2);

        //Getting element 0 of "Color Output" in "Azure Inspector".
        //This Gradient Output will be based on the Timeline.
        myLight.color     = azureOutput.AzureGetGradientOutput(0,0);
    }
}
```

As you can see in the above script, "AzureSkyController" contain two public methods that you can access by other scripts to get the values of the outputs. If you have created multiple outputs you need pass to the method the element number of output that you want to access.

Methods:

AzureGetCurveOutput*(int index, int curveMode)*
AzureGetGradientOutput*(int index, int curveMode)*

Index: Is the element number of the Output that you want to access.
CurveMode: Is the curve mode you want the Output to be based on.

0 = Timeline.
1 = Sun Elevation.
2 = Moon Elevation.

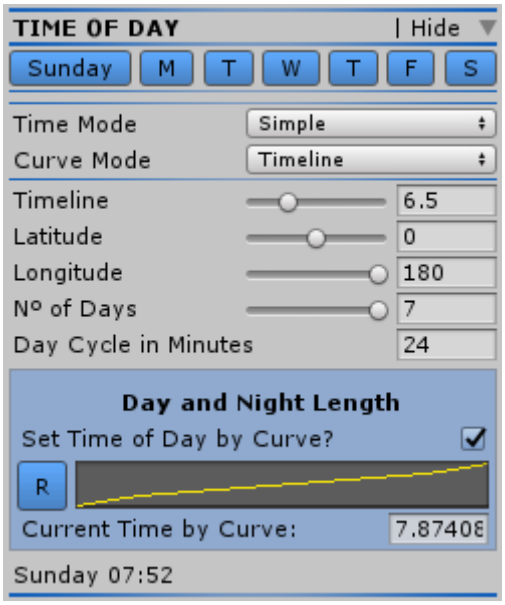
Access the link to watch a video that shows the entire process of using the Output System. https://www.youtube.com/watch?v=5t4lnAo_LxI

PS: The video was created using the Azure[Sky]Lite version, so the curve mode does not exist in this version and is not shown in the video.

CUSTOMIZING THE LENGTH OF DAY AND NIGHT

Azure allows you to customize the length of day and night. With this feature you can create a day and night cycle with length of 15 minutes and make the daytime with 10 minutes and the night with 5 minutes.

In this example, to be easier to understand we set the property "Day Cycle in Minutes" to 24, thus each hour of the game will last 1 minute in real life time. Also check the "Set Time of Day by Curve" for the length of daytime and night are based on the curve time.



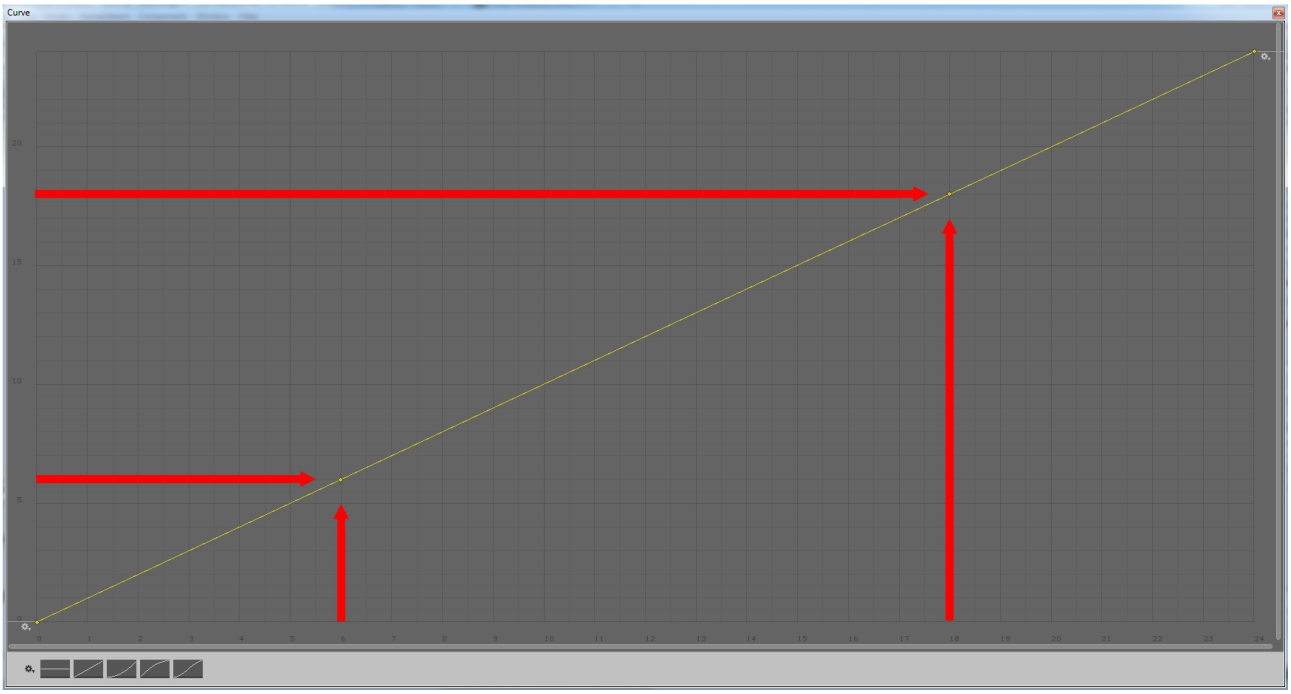
Before you edit the time curve, you need to know that the daytime in the Simple time mode begins exactly at 6am and ends at 18pm. The night begins exactly at 18pm and ends at 6am. This information is very important and helps to understand how to edit the time curve without problems.

The time curve has values between 0-24. In the horizontal line representing the value of timeline slider and also has values between 0-24. In the vertical line representing the hours you want set at that time. It seems to be complicated but is very simple and with the tips that I will give will become easier to understand.

In the first step you need to:

- *Create a key at the intersection point between 6am of the horizontal axis with the vertical axis.*
- *Create a key at the intersection point between 18pm of the horizontal axis with the vertical axis.*

Exemple Image:



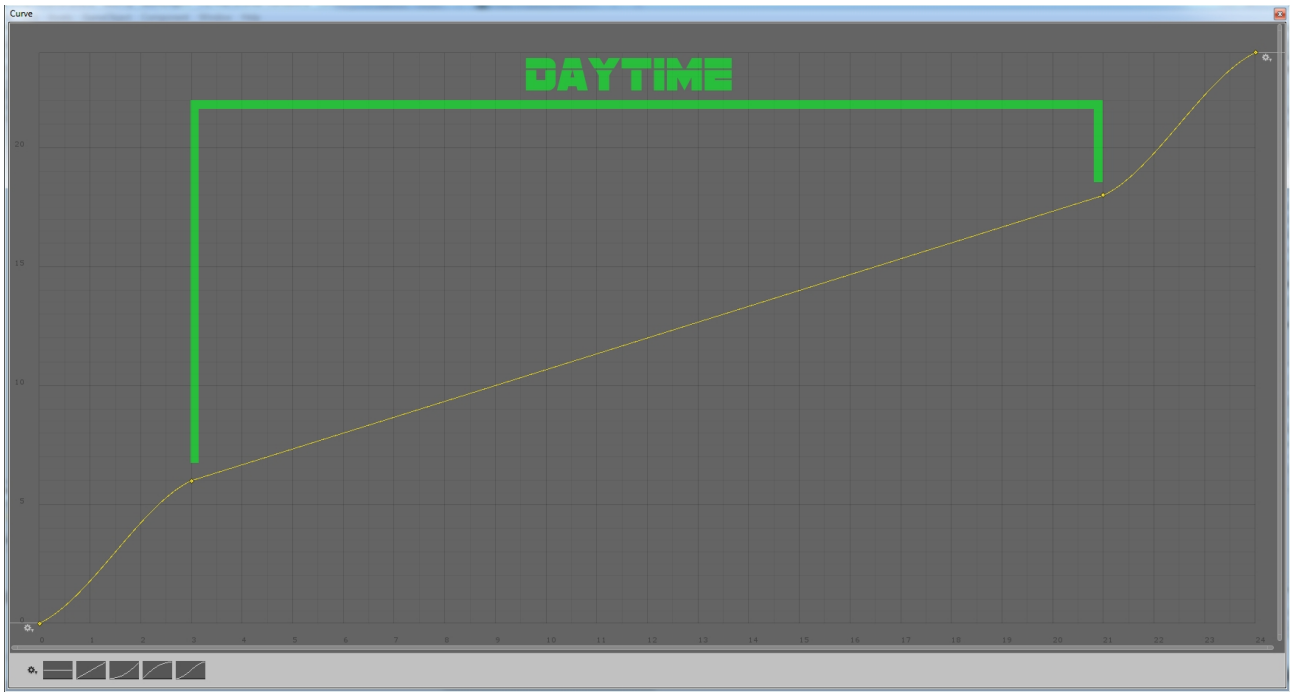
- *As you can see in the example image above, the hours are between the keys (6am - 18pm) represent the daytime and the hours that are outside of the keys represent the night.*

Now to change the length of day and night is very simple, just move the key points on the horizontal axis by increasing or decreasing the hours between them.

In this example we will make the length of the day to be greater than the night. Then we will move the 6am key to value of 3 hours to the left and move the 18pm key to value of 3 hours to the right.
Just make sure to move each of the keys the same number of hours.

- *Click with the right mouse button on the 6am key and choose "**Right Tanjent >> Linear**".*
- *Click with the right mouse button on the 18pm key and choose "**LeftTanjent >> Linear**".*

Exemple Image:



- *As you can see in the picture above the daytime will start when the timeline value is 3h and will end when the timeline value is 21h.*
- *The night will begin when the timeline value is 21h and will end when the timeline value is 3h.*
- *In this example the daytime will last 18 minutes and the night will last only 6 minutes.*
- *The properties of the sky will adapt to whatever is the length of daytime and night, because they are based on hour of day instead of the cycle of day.*

AZURE METHODS

Below are some methods that may be of interest to you and may help you.

`public void AzureSetTime(float hour, float dayDuration)` Set the "Time of Day" and duration of Day/Night cycle. If you need to change only the duration of the day you can try something like this: *AzureSetTime(Azure_Timeline, 10.0f);*
This will change only the day duration and keep the current time.

`public Vector2 AzureGetHourAndMinutes()` Convert timeline to hour and minutes. Will return a Vector2(hour, minutes). This method is useful if you want to show the correct time of day in your game.
Exemple:

```
void Update()
{
    Vector2 hours = AzureGetHourAndMinutes();
    Debug.Log( hours.x.ToString("00") +":" + hours.y.ToString("00"));
}
```