Azure[Sky]

Dynamic Skybox

Document Version 4.2.0

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

Introduction

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[Sky] contains two different sky systems that work independently and each of them has a different purpose and features. The sky material can be used as a default skybox or as a skydome and it is possible to customize the sky with several different styles and effects.

Standard Sky System: This sky system has been completely redone in version 4.2.0 and comes with a new interface and many new features. The Standard Sky System exists and is improved since the beginning of Azure development, so it has this name.

The main new features of the Standard Sky System are:

- new Profiles System
- new Climate Control
- new Sound effects
- new Height Fog
- new Calendar System
- new Moon System redone from scratch
- new Sky Model redone from scratch
- new Fog System redone from scratch
- new Curve System redone from scratch
- new Output System redone from scratch

Precomputed Sky System: This sky system is based on the most realistic sky model of the present days, it uses precomputed data and gives you a high fidelity sky like the real life. This sky model supports altitude variations and is suitable for anyone who wants an extremely realistic atmospheric effect. *The new Precomputed Sky System is still in development, so the package version 4.1.2 is included in the package allowing the use of the old version of the precomputed sky system along with the old version of the standard sky system. Check the documentation of version 4.1.2 for how to use the old sky system.*

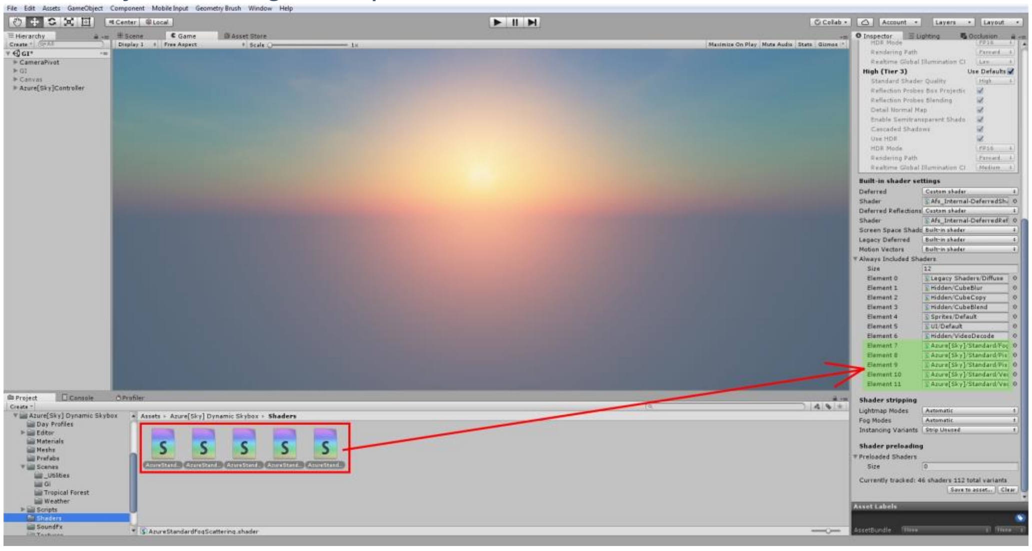
Getting Started

- It is always a good practice to save a copy of your project before importing any package or creating a new project for testing.

- If the AA is enabled, the fog scattering effect can cause artifacts at the edges of the scene objects. Disable Anti Aliasing in the project quality settings by the menu: *Edit>Project Settings>Quality.*

- Azure works in both color spaces, but I advise switching the project to linear color space, since all sample scenes have been set up in linear color space.

- To avoid future errors after building the project, add all variations of Azure[Sky] shaders in the "**Always Included Shaders**" list located in: *Edit>Project Settings>Graphics*.



Built-in shader settings

Deferred	Custom shader
Shader	Afs_Internal-DeferredSh
Deferred Reflections	Custom shader
Shader	Afs_Internal-DeferredRef
Screen Space Shado	Built-in shader
Legacy Deferred	Built-in shader
Motion Vectors	Built-in shader

Always Included Shaders

Size	12
Element 0	Legacy Shaders/Diffuse
Element 1	Hidden/CubeBlur
Element 2	Hidden/CubeCopy
Element 3	Hidden/CubeBlend
Element 4	Sprites/Default
Element 5	UI/Default
Element 6	Hidden/VideoDecode
Element 7	Azure[Sky]/Standard/Fo
Element 8	Azure[Sky]/Standard/Pix
Element 9	Azure[Sky]/Standard/Pix
Element 10	Azure[Sky]/Standard/Ve
Element 11	Azure[Sky]/Standard/Ve

Shader stripping

Lightmap Modes	Automatic
Fog Modes	Automatic
Instancing Variants	Strip Unused

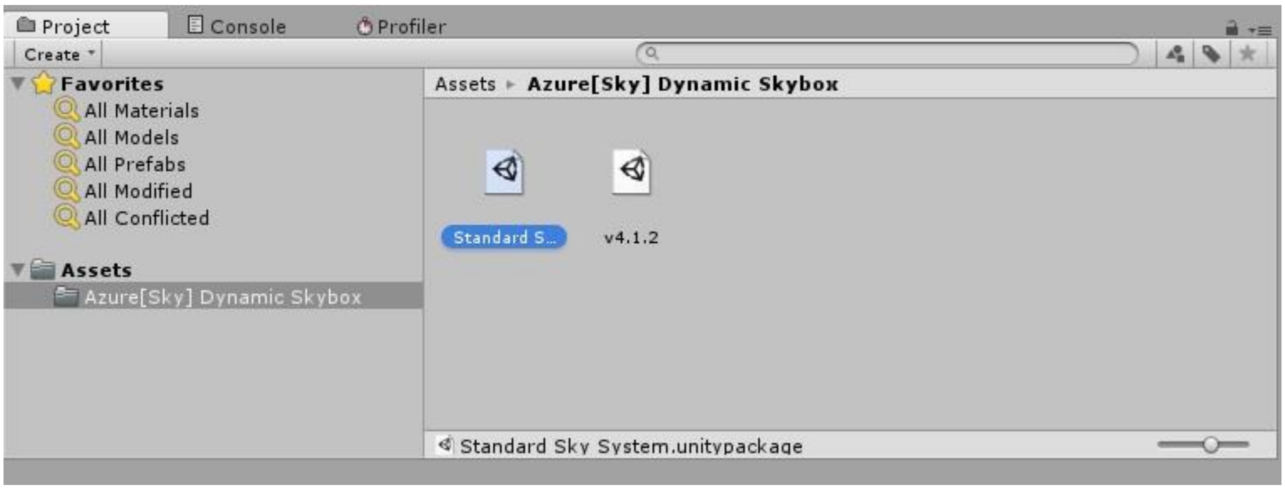
Shader preloading

Preloaded Shaders Size 0

Currently tracked: 46 shaders 112 total variants

AzureStandardFogScattering.shader

When you import **Azure[Sky] Dynamic Skybox** from the Asset Store, the package will come with two subpackages. One of the subpackages contains the new **Standard Sky System** and as the new **Precomputed Sky System** is still under development, the other subpackage contains the previous version of Azure[Sky] with the two old sky models built into the same system.



It does not make much sense to use both systems in the same project, so choose only one and double click to unpack in your project the chosen sky system.

- While the new **Precomputed Sky System** is under development, the version 4.1.2 will be included for those who need to use the old version of the precomputed sky system.

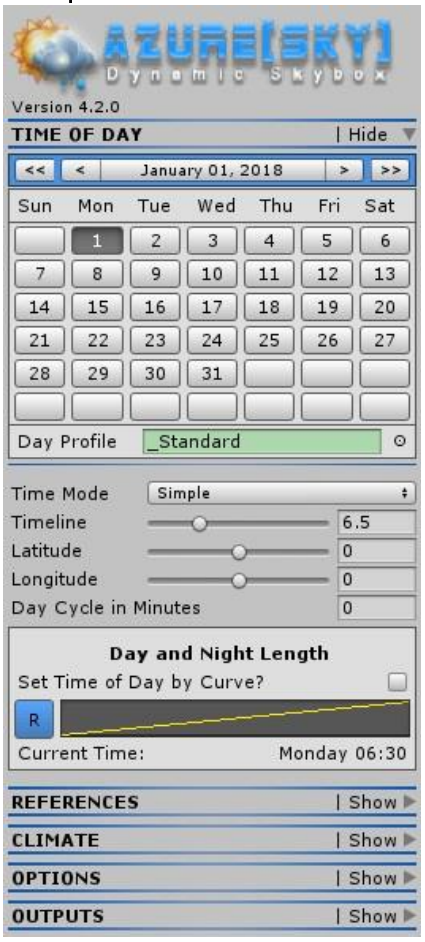
- For now, do not unpack the two systems in the same project, since many scripts and classes share the same names of the old version, and therefore this action can cause many errors.

- Version 4.1.2 comes with its own documentation, so this documentation will explain only the new **Standard Sky System**.

After unpacking the Standard Sky System in your project, all you have to do is drag the **Azure[Sky]Controller** prefab to your scene. Also, add the fog scattering effect to the camera by dragging the **AzureSkyFogScattering.cs** script to the camera Inspector or by the menu: *Component>Azure[Sky]>Fog Scattering.*

- The **Azure[Sky]Controller** prefab is located inside the "Prefabs" folder.
- The Azure's sky material will be applied to the skybox automatically.
- Make sure you have removed any other directional light source from the scene, the sky controller prefab already comes with two directional lights for the sun and moon lighting.

All right, now you can start customizing the sky... When you select the sky controller, the Inspector should look like this:



Version 4.2.0

TIME OF DAY | Hide

<< < January 01, 2018 > >>

Sun	Mon	Tue	Wed	Thu	Fri	Sat
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31			

Day Profile _Standard

Time Mode Simple
Timeline 6.5
Latitude 0
Longitude 0
Day Cycle in Minutes 0

Day and Night Length
Set Time of Day by Curve?
R
Current Time: Monday 06:30

REFERENCES | Show
CLIMATE | Show
OPTIONS | Show
OUTPUTS | Show

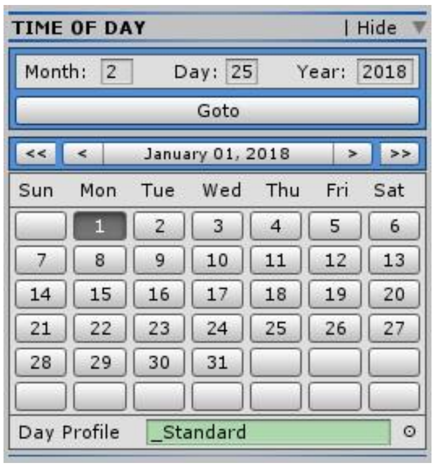
Time of Day Tab:

On the Time of Day tab are all properties for setting time, date, and location.

Date:

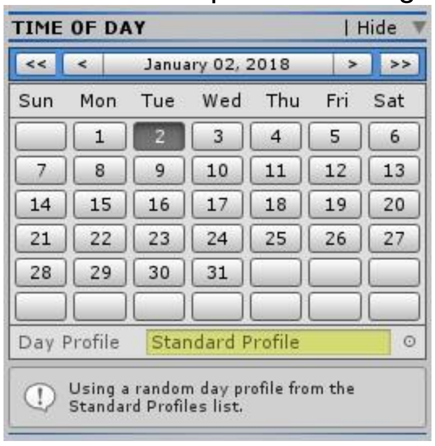
- You can change the date by just browsing the calendar.

- In the calendar header are the buttons for changing the month and year. If you press the center button that shows the date, it will open a panel to enter a custom date.



- The Day Profile property is the object field to attach a profile to the selected day. You can add a profile for each day in a one-year cycle, even if the defined year is leap year. If the selected day contains an attached profile, the object field will be green.

- You are not required to attach a profile to every day of the year, the sky controller will automatically check if the selected date has an attached profile. If the selected day does not have an attached profile, the sky controller will randomly choose a profile from the Standard Profiles list located on the Options tab, the object field will turn yellow and a message will appear saying that a random profile is being used on that day.



Using a random day profile from the Standard Profiles list.

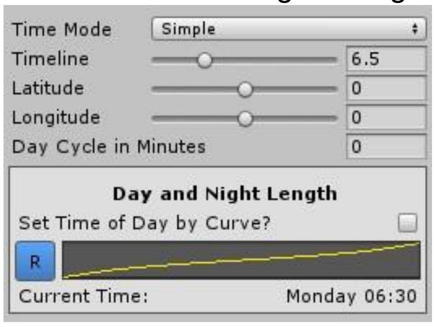
Time and Location:

- Azure has two Time Modes:
    - The **Simple** time mode is faster and in this mode, the moon is always on the opposite side of the sun.
    - The **Realistic** time mode realistically positions the sun and moon according to the time, date, and geographic location.
    - You can change the location by adjusting the **Latitude** and **Longitude** properties. When Realistic time mode is selected, the **UTC** property will appear in the Inspector to set the time zone.

- To change the time of day, simply drag the Timeline slider.

- In the **Day Cycle in Minutes** property you define the time in minutes that a full cycle of the day will last. If it is zero, the day will remain static at the time set in the timeline.

- The days and nights of Azure have the same duration, but it is possible to make the daytime longer than the night or the night longer than the daytime. To do this, simply activate the Set Time of Day by Curve option and adjust it with the setting that is right for you. By default the curve is set for the daytime to last twice the length of night time.



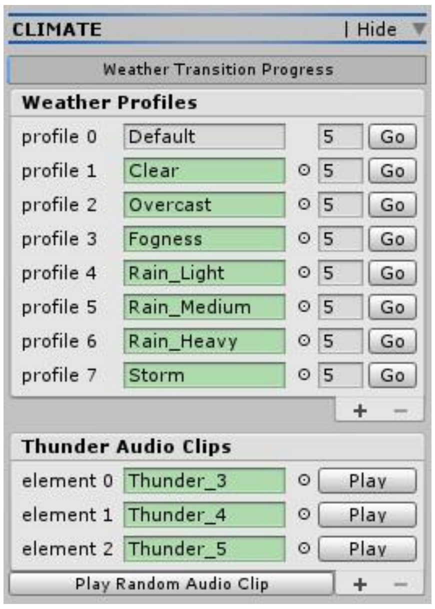
Time Mode: Simple
Timeline: 6.5
Latitude: 0
Longitude: 0
Day Cycle in Minutes: 0

Day and Night Length
Set Time of Day by Curve?
Current Time: Monday 06:30

References Tab:

In the References tab are attached all Materials, GameObjects, Textures, Sounds and other resources used by Azure. There is not much to do on this tab, but if you want to change for example the textures of the sun or moon, this is the right place.

Climate Tab:



CLIMATE | Hide

Weather Transition Progress

Weather Profiles

profile 0	Default	5	Go
profile 1	Clear	5	Go
profile 2	Overcast	5	Go
profile 3	Fogness	5	Go
profile 4	Rain_Light	5	Go
profile 5	Rain_Medium	5	Go
profile 6	Rain_Heavy	5	Go
profile 7	Storm	5	Go

Thunder Audio Clips

element 0	Thunder_3	Play
element 1	Thunder_4	Play
element 2	Thunder_5	Play

Play Random Audio Clip

In the Climate tab is made the whole configuration of climatic changes of your game. **Weather Profiles:** Weather Profiles is a list of profiles that contains all the weather variations used in your game. The new Standard Sky System uses the Profiles System to customize the sky and each profile can store the customization of a complete day cycle. So to change the weather, just change the current day profile by a profile that has the day set for rain, for example. *Later will be explained how to create your own profiles.*

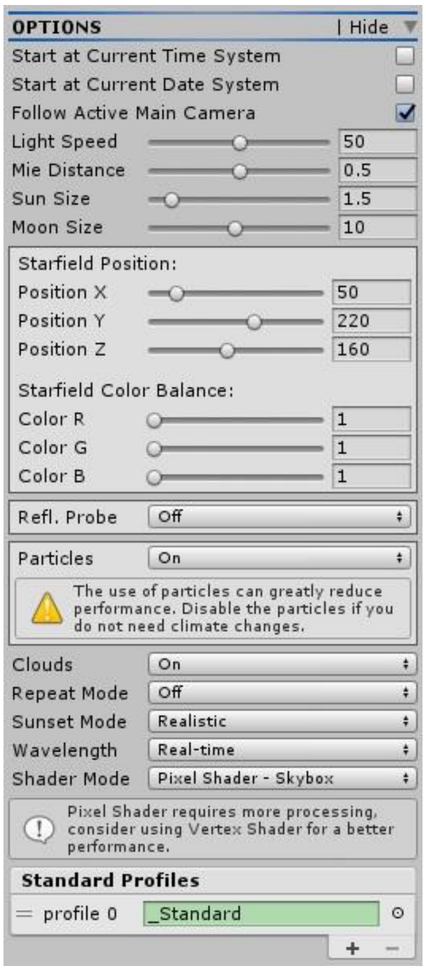
- You can add or remove from the **Weather Profiles** list as many profiles as you want. In the Profiles folder there is a range of pre-defined profiles with the most common types of weathers ready for use.
- Next to each profile in the list, there is a field for you to enter the transition time for that profile.
- Any profile can be used as a weather and you can create as many variations of the same type of weather you want. You can even use a different profile for morning, afternoon and night, everything will depend on your needs.
- When the application is playing in the **Editor**, the easiest way to change the weather profile to do some quick testing is to hit the "**Go**" button next to each profile in the list.
- To dynamically change the weather during gameplay, you just need to call the public method: **SetNewWeatherProfile (int target)** from the *AzureSkyController* script and pass as parameter, the profile number in the **Weather Profiles** list.
- To return to the original profile defined in the calendar, simply call the Default profile from the list. If the day changes during the transition, to avoid breaking the transition, the sky controller will continue the transition to the previous day's profile, so in that case you need to chall the Default profile again after the transition is over.

Thunder Audio Clips: In this list you can add the sound effects of thunderstorms. Depending on the occasion, you can use this feature to play other types of sound effects.

- When the application is playing in the **Editor**, the easiest way to play the thunder audio clip to do some quick testing is to hit the "**Play**" button next to each sound element in the list. In addition, you can also press the "**Play Random Audio Clip**" button to play a random sound from the list.

- To dynamically play a thunder audio clip during gameplay, you just need to call the public method: **PlayThunderAudioClip (int element)** from the *AzureSkyController* script and pass as parameter, the element number in the **Thunder Audio Clips** list. In addition, you can also call the public method: **PlayThunderAudioClipRandom ()** from the *AzureSkyController* script to play a random sound from the list.

Options Tab:



OPTIONS | Hide

Start at Current Time System	☐
Start at Current Date System	☐
Follow Active Main Camera	✓
Light Speed	50
Mie Distance	0.5
Sun Size	1.5
Moon Size	10

Starfield Position:

Position X	50
Position Y	220
Position Z	160

Starfield Color Balance:

Color R	1
Color G	1
Color B	1

Refl. Probe	Off
Particles	On

The use of particles can greatly reduce performance. Disable the particles if you do not need climate changes.

Clouds	On
Repeat Mode	Off
Sunset Mode	Realistic
Wavelength	Real-time
Shader Mode	Pixel Shader - Skybox

Pixel Shader requires more processing, consider using Vertex Shader for a better performance.

Standard Profiles

profile 0	_Standard

In the Options tab there are many options that will define how the sky controller will work, here you will also find some customization options that are constant regardless of the configuration of each profile and therefore do not have to configure several times in all profiles.

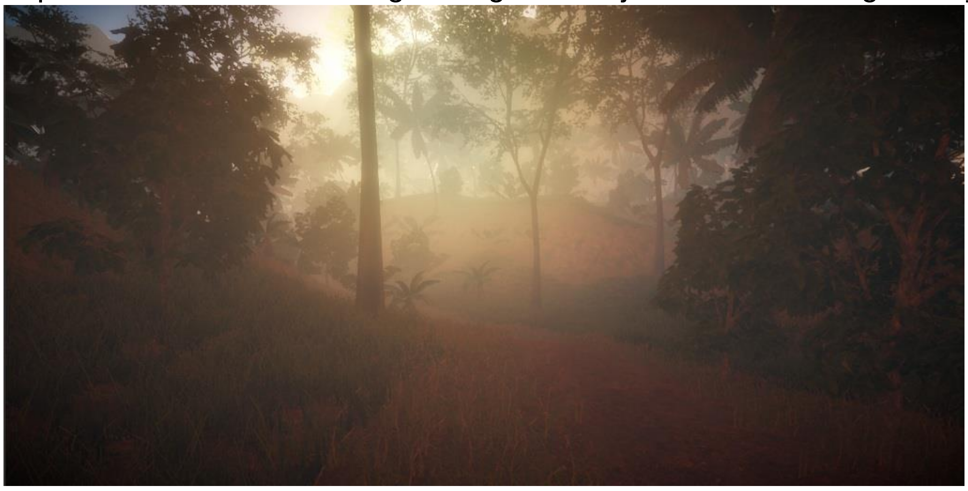
The name of most of the options are self explanatory, so I'll just cite the options that need to be explained in more detail.

Follow Active Main Camera: This option is very important, especially if you want to use the effects of rain and snow. If enabled, this option makes the sky controller always stay in the same position as the active camera tagged with the MainCamera Tag, so that the particles remain always visible on the screen.

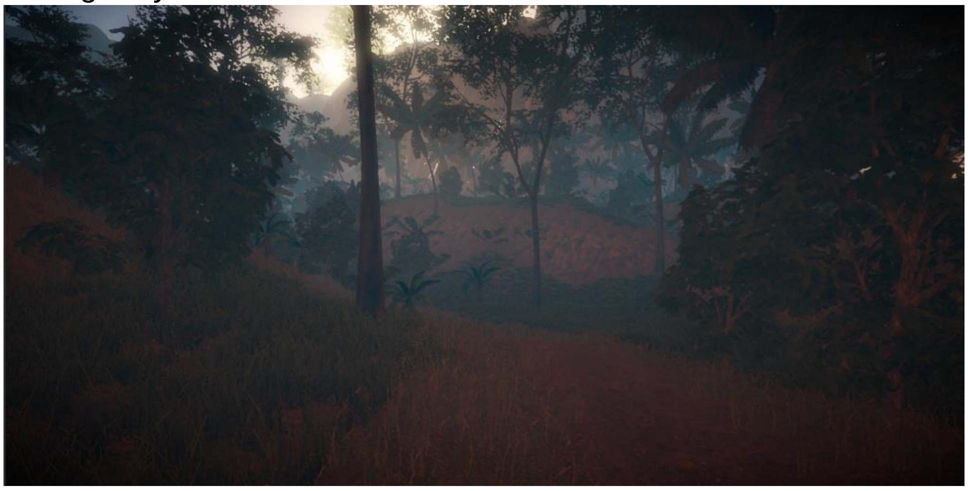
Light Speed: This option sets the speed at which the sun's mie brightness in the sky will rise and disappear on the horizon during sunrise and sunset.

Mie Distance: This option controls the distance of Mie brightness when the fog scattering effect is attached to the camera.

- At some point, you will come across a scene with very close objects such as a very dense forest and the mie glow of the fog can get very close, giving the impression that it is crossing through the objects when the fog is very dense.



- In that case, you can increase the mie brightness distance to get the best setting for your scene.



Refl. Probe: In this option you can activate the reflection probe. If enabled, some options will show to adjust the probe refresh mode. Any other settings, you should do directly on the reflection probe located inside the sky controller game object. Note that using the reflection probe can dramatically reduce performance.

Particles: Enables and disables the use of standard sky controller particle effects. If disabled, "Keep Weather Properties Updated" option will appear, this option causes the Weather tab properties to continue to be updated, so you can access each of these properties to control your preferred third party particle system. Particles can also dramatically decrease performance.

Repeat Mode: This option sets the repeat mode of the days, you can change the repeat mode to always repeat the same day, the same month, or the same year.

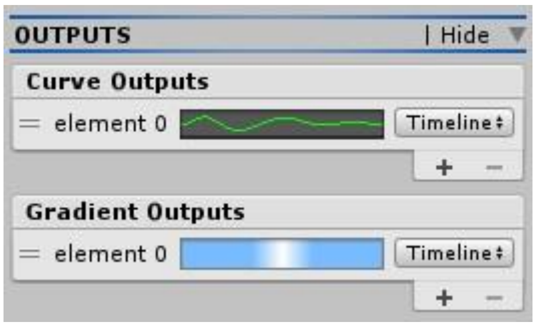
Shader Mode: Sets the calculation mode in the shader and applies the sky material appropriately as skybox or skydome.

- **Pixel Shader:** The calculations are done per pixel, it is more expensive for performance, but it is the ideal mode to use as Unity's skybox because it has a very low polygon count.
- **Vertex Shader:** The calculations are done by vertex, it is better for performance and is the ideal mode to use as a skydome if the mesh does not have a very low polygon count.

Standard Profiles: In this list you can add the profiles that will be used on days that do not have a selected profile in the calendar. If the list has more than one profile, a random profile will be chosen.

Outputs Tab:

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

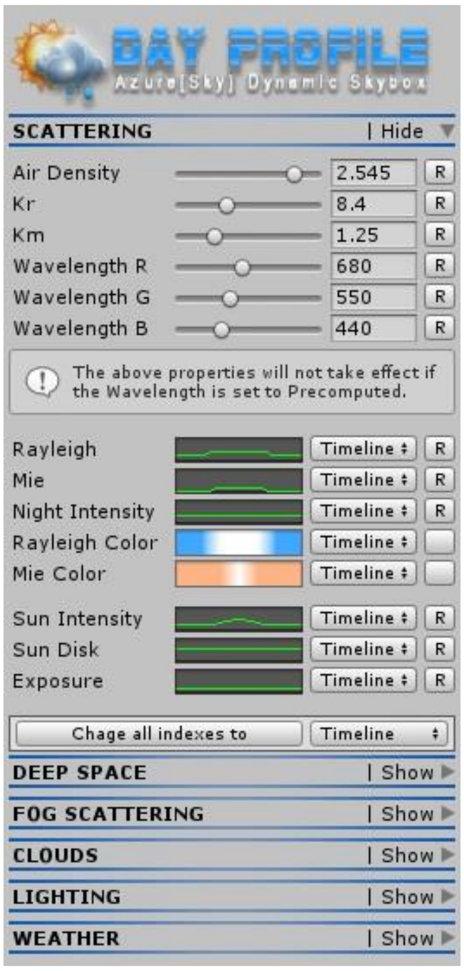


- You can access each Curve Output by calling the public method: **GetCurveOutput (int element)** of the *AzureSkyController* script and pass as parameter, the element number in the list. The output will return a float value between 0 and 1 evaluated by the timeline.

- You can access each Gradient Output by calling the public method: **GetGradientOutput (int element)** of the *AzureSkyController* script and pass as parameter, the element number in the list. The output will return a color evaluated by the timeline.

- Next to each Output in the lists, you have the option to set the mode the curve or gradient will be evaluated.
    - **Timeline:** The output will be evaluated by the timeline.
    - **Sun:** The output will be evaluated by the sun elevation in the sky.
    - **Moon:** The output will be evaluated by the moon elevation in the sky.

Profiles:



SCATTERING | Hide

Air Density	2.545	R
Kr	8.4	R
Km	1.25	R
Wavelength R	680	R
Wavelength G	550	R
Wavelength B	440	R

The above properties will not take effect if the Wavelength is set to Precomputed.

Rayleigh	Timeline	R
Mie	Timeline	R
Night Intensity	Timeline	R
Rayleigh Color	Timeline	
Mie Color	Timeline	
Sun Intensity	Timeline	R
Sun Disk	Timeline	R
Exposure	Timeline	R

Chage all indexes to | Timeline

DEEP SPACE | Show

FOG SCATTERING | Show

CLOUDS | Show

LIGHTING | Show

WEATHER | Show

Profiles store the customization information of a complete day cycle. The advantages of using profiles are many.

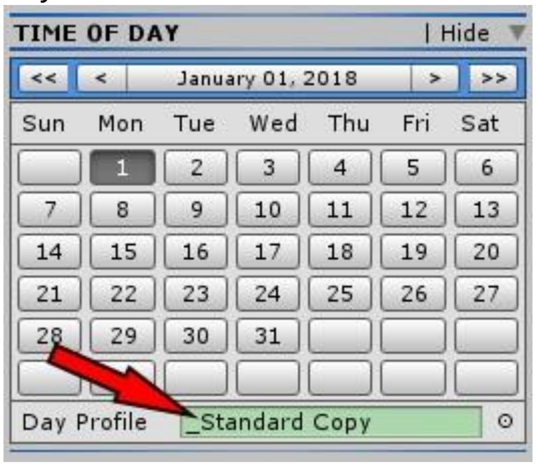
- You can create as many profiles as you need.
- The profiles can be shared among team members.
- And one of the best reasons to use profiles is that it is possible to blend the profiles based on weights, an exceptional feature to make perfect weather transitions.

Creating a profile...

To create a profile is very simple, just click on the menu: *Create>Azure[Sky] Dynamic Skybox>New Day Profile*. This way will create a profile without the proper configuration with all colors and gradients set to white. Then the best way is to select the **_Standard** profile from the Profile folder and click on the keyboard shortcut "Ctrl + D" to duplicate the existing profile.

- Now you can start to customize your new profile.

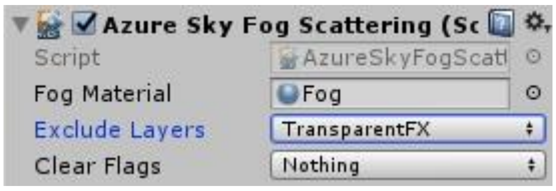
- To view your profile changes in real time while you customize it, the best way is to temporarily add the new profile to the profile field of the current calendar day.



Day Profile _Standard Copy

Fog Scattering:

The fog scattering effect of the new Standard Sky System adds a lot of realism to the scene. To attach the fog effect to your camera, simply drag the **AzureSkyFogScattering.cs** script to the Camera Inspector or with the camera selected go to the menu: *Component>Azure[Sky]>Fog Scattering*.



Azure Sky Fog Scattering (Sc

Script	AzureSkyFogScatt
Fog Material	Fog
Exclude Layers	TransparentFX
Clear Flags	Nothing

- Now the fog effect has the option to ignore any selected layer in the **Exclude Layers** property. Note that this feature demands a bit more performance, but so far it's the best solution I've come up with.

- It is recommended to position the fog effect before any post-processing effects that may be attached to your camera such as Unity Post Processing Stack or similar.

- The fog is only applied on the scene objects that draw to the depth buffer. Particle effects are not usually drawn in the depth buffer, so the best way is to add your particle effects to a layer excluded by the fog, this will prevent strange behaviors between the particles and the fog scattering effect.

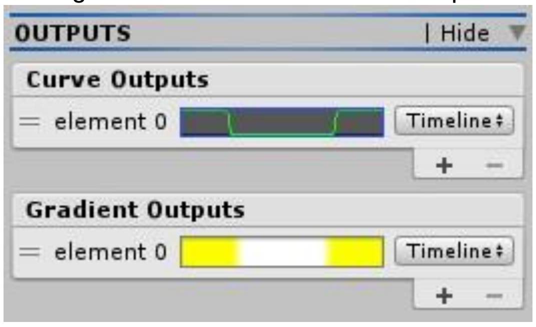
- The fog has been set by default from the aerial view perspective of the weather demo scene, so you may need to adjust the distance and density of the fog so that it fits perfectly in your scene.

Useful Methods:

The AzureSkyController class contains a range of methods that can be useful when programming the mechanics of your game. To access all Azure[Sky] classes and components, you must use the "UnityEngine.AzureSky" namespace.

- Below is an example of a script that controls the intensity and color of a light, taking the value of the Curve and Gradient Outputs previously created on the Outputs tab of the Sky Controller's Inspector.
  The Curve Output has been set to turn on the light at night*(increasing the light intensity to 1)* and turning off the light during the day*(decreasing the light intensity to 0)*.
  The Gradient Output has been set to apply a yellow color to the light at night.

  - Setting of the Curve and Gradient Outputs:



  - Exemple Script:

```
using UnityEngine;
using UnityEngine.AzureSky;

public class SpotLightController : MonoBehaviour
{
    public AzureSkyController skyController;
    private Light m_spotLight;

   //Update is called once per frame
    void Update ()
    {
        m_spotLight.intensity = skyController.GetCurveOutput (0);
        m_spotLight.color = skyController.GetGradientOutput (0);
    }
}
```

- With the same example, you have already learned how easy it is to use the Output System. You can use the outputs to control various elements of your game, your imagination is the limit.

GetCurveOutput

```
public float GetCurveOutput (int element);
```

Parameters:

element: The element number of the Curve Output list.

Description:

Get curve output and return as a float. You can easily convert to int or bool if necessary.

GetGradientOutput

```
public Color GetGradientOutput (int element);
```

Parameters:

element: The element number of the Gradient Output list.

Description:

Get gradient output and return as a Color.

GetCalendarDayProfile

```
public AzureSkyProfile GetCalendarDayProfile ();
```

Description:

Get the current day profile from the calendar. It will return a random standard profile if there is no day profile attached to the current calendar day.

SetNewWeatherProfile

```
public void SetNewWeatherProfile (int target);
```

Parameters:

target: The profile number in the "Weather Profiles" list located on the Climate tab of the Sky Controller Inspector.

Description:

Changes the weather with a smooth transition.

PlayThunderAudioClip

```
public void PlayThunderAudioClip (int element);
```

Parameters:

element: The element number in the "Thunder Audio Clips" list located on the Climate tab of the Sky Controller Inspector.

Description:

It plays the thunder sound fx from the "Thunder Audio Clips" list.

PlayThunderAudioClipRandom

```
public void PlayThunderAudioClipRandom ();
```

Description:

It plays a random thunder sound fx from the "Thunder Audio Clips" list located on the Climate tab of the Sky Controller Inspector.

SetParticlesActive

```
public void SetParticlesActive (bool value);
public void SetParticlesActive (int value);
```

Parameters:

value: Activate or deactivate the particles effects.

Description:

Activates/Deactivates the Particles GameObjects.

UpdateProfiles

```
public void UpdateProfiles ();
```

Description:

Updates the profiles and calendar days. It is very important to always call this method right after you change the date through other scripts.
Exemple:

```
using UnityEngine;
using UnityEngine.AzureSky;

public class ChangeDate : MonoBehaviour
{
    public AzureSkyController skyController;

    void Start ()
    {
        skyController.timeOfDay.GotoDate (9, 15, 1903);
        skyController.UpdateProfiles ();//Update profiles after change the date.
    }
}
```

Time of Day Component:

The **AzureSkyTimeOfDayComponent** class contains many methods related to the date and time that may be useful during the development of your game. To access these methods, you must do this through the "timeOfDay" property of the **AzureSkyController** class.

Exemple:

```
using UnityEngine;
using UnityEngine.AzureSky;

public class AccessTimeOfDayComponent : MonoBehaviour
{
    public AzureSkyController skyController;
    private Vector3 m_myDate;

    void Start ()
    {
        m_myDate = skyController.timeOfDay.GetDate ();
        Debug.Log ("Month:" + m_myDate.x);
        Debug.Log ("Day:"   + m_myDate.y);
        Debug.Log ("Year:"  + m_myDate.z);
    }
}
```

GetDayOfWeek

```
public int GetDayOfWeek ();
```

Description:

Get the current day of the week and return an integer between 0 and 6.

GetDayOfYear

```
public int GetDayOfYear ();
```

Description:

Returns the current day number of a 366-day year. The return value is fix when it is not leap year, this method is ideal for accessing, adding, or removing day profiles from the calendar. If you want to get the correct number for the day in the year, then use System.DateTime.DayOfYear instead.

IsLeapYear

```
public bool IsLeapYear ();
```

Description:
Return true if the current year is a leap year.

GetTime

```
public Vector2 GetTime ();
```

Description:
Converts the timeline in hours and minutes and returns as a Vector2 (hours, minutes).

GetDate

```
public Vector2 GetDate ();
```

Description:
Get the current date and returns as a Vector3 (month, day, year).

JumpToNextDay

```
public void JumpToNextDay ( bool keepTime = true);
```

Parameters:
keepTime: Keep the current time? If true, will keep the current time without resetting the day to 00:00 hours.

Description:
Skips to the next day and ignores the Repeat Mode set in the Inspector.

GotoDate

```
public void GotoDate (int month, int day, int year);
```

Parameters:
month: The number of the month you want to go.
day: The number of the day you want to go.
year: The number of the year you want to go.

Description:
Go to a custom date.

GotoTime

```
public void GotoTime (int hours, int minutes);
```

Parameters:

hours: The hour you want to go.
minutes: The minutes you want to go.

Description:

Go to a custom time.