Documentation
v 1.8.1

Thank you for buying
Enviro – Sky and Weather

Getting started

It's really easy to setup enviro in your project! Import the package and follow these easy steps to get started!

1. **Drag and drop the "EnviroSky 1.8" prefab ("Enviro - Dynamic Environments" -> "Prefabs") into your scene.**

2. **Assign your "Player Object" and your "Player Camera" OR check "AssignInRuntime" and fill in your Tag's.**

3. **You are ready to go!**

Further settings to check:

✓ Disable other directional lights in your scene!

✓ For best visuals use "Linear Color Space"!

✓ Enviro uses one layer for cloud and one layer for satellite rendering. Defaults: 30, 31

✓ Do not use Desktop and Mobile version in same project, it won't work!

Attention!

Sometimes the import of Enviro can fail and not all settings are applied. This will most likely break the system.

The fix is easy, just reimport the package and **do not delete** the enviro folder.

All Settings should be applied corretly now!

Tweaking the Settings

When you done with the initial setup you may want to tweak the settings for your project. Take a look on the EnviroSky component. All settings are sorted in categories to get better overview of all the different features.

Gradients and Curves

Some settings like Light Color or Light Intensity using Curves or Gradients to be able to control these over the day right in the editor. To be able to get correct values for all possible locations on earth, Enviro uses the position of the sun in sky.
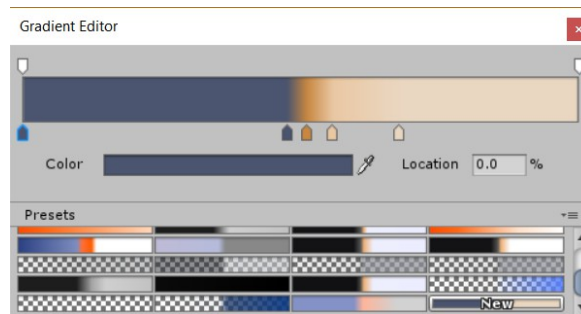
The left side of these gradients and curves representing nighttime, the right side representing daytime. So at the middle the sun will stay exactly at the horizon.

Rendering Setup
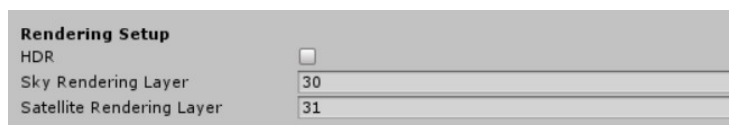
Check the the HDR checkbox to use High Dynamic Range Color buffers. You of course want to use an 3rd party tonemapping effect on your camera to get best results. HDR Rendering can be very usefull for Image Effects! If unchecked Enviro will do the exposure tonemapping for you.

Enviro need two free layers to work correctly. You may want to name these layers in Unity for a clear project structure.

Time

Here we can control the date, time of day, time progress mode and location.
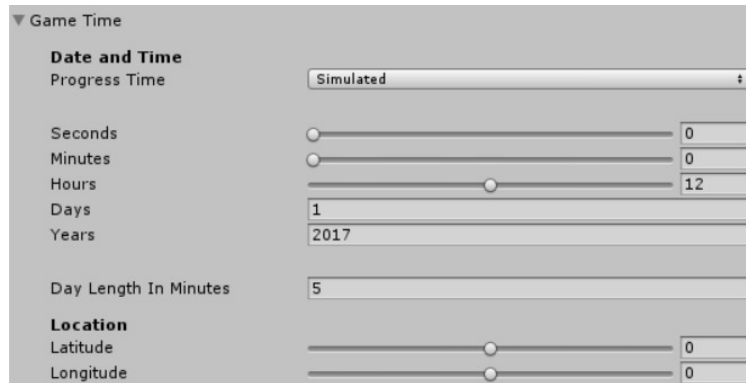

You also have the options to use your system time or stop the time progress completely.
In "Simulated" Progress Mode a complete day cycle will take the "Day Lenght In Minutes"

Audio

You can control the global volume of ambient and weather sound effect in EnviroSky component → "Audio" category.
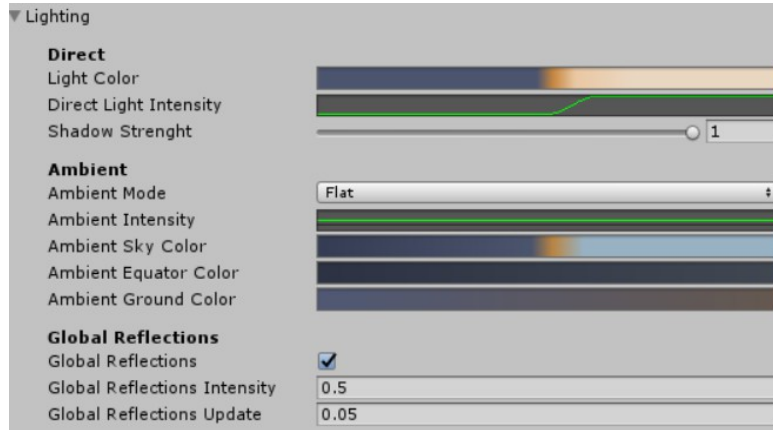

What to play will be defined in current playing weather. You of course can leave it empty to play nothing in this particular weather.
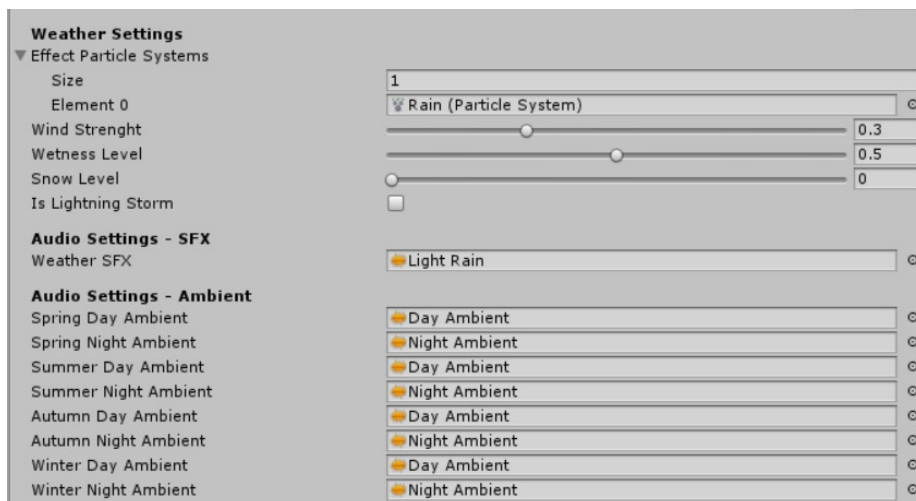
Lighting

*The lighting will be calculated based on the global gradients in "Lighting" category in EnviroSky component **AND** the "Weather Light Mod" gradient of current playing weather.*

This gradient represents the whole day and is based on current sun position like all the other gradients. The alpha value will define the "intensity" of overriding the global light color.
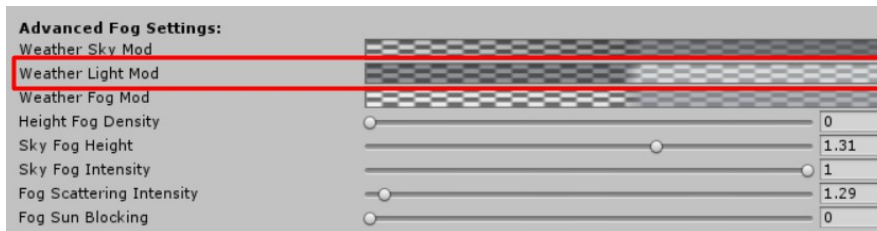
Fog

At the "Fog" category in EnviroSky component you have global control over the Enviro Fog Image Effect. **Please note** *that you define the Fog Intensity in each weathertype, you also can modify the fog color over the day with the "Weather Fog Mod" gradient there.*
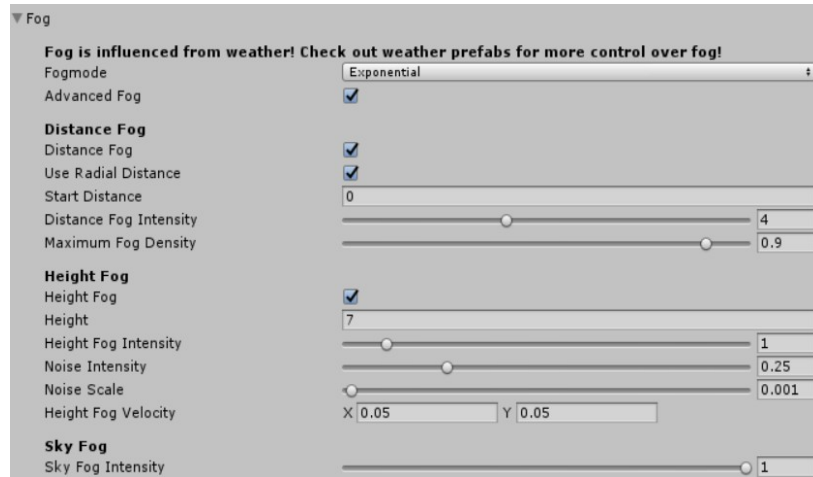

The "Weather Fog Mod" gradient represents the whole day and is based on current sun position like all the other gradients. The alpha value will define the "Intensity" of overriding the global light color.
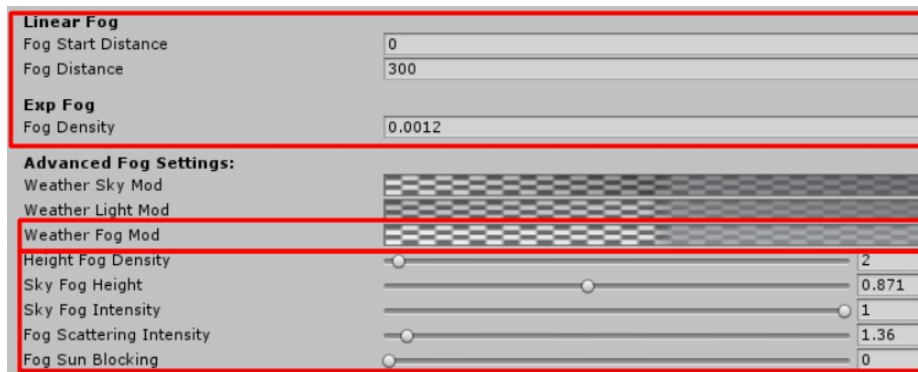

The Fog Intensity will be controlled with the "Fog Start Distance" and "Fog Distance" when choosing "Linear Fog" in global fog control in EnviroSky component. If choosing "Exponential" Fog, the "Fog Density" value be used.
*You also can control the height fog density and sky fogging here. The "Fog Scattering Intensity will control the fog light scattering strength. The "Sun Blocking" value will control if and how much of the sundisk will be visible through the sky fog.*

Clouds

In the "Clouds" category of EnviroSky component you are able to setup the different layers, clouds altitude, wind animations and cloud lighting. The actual clouds colors and coverage will be controlled in each weather type.
For more details about each setting hover over the elements please.
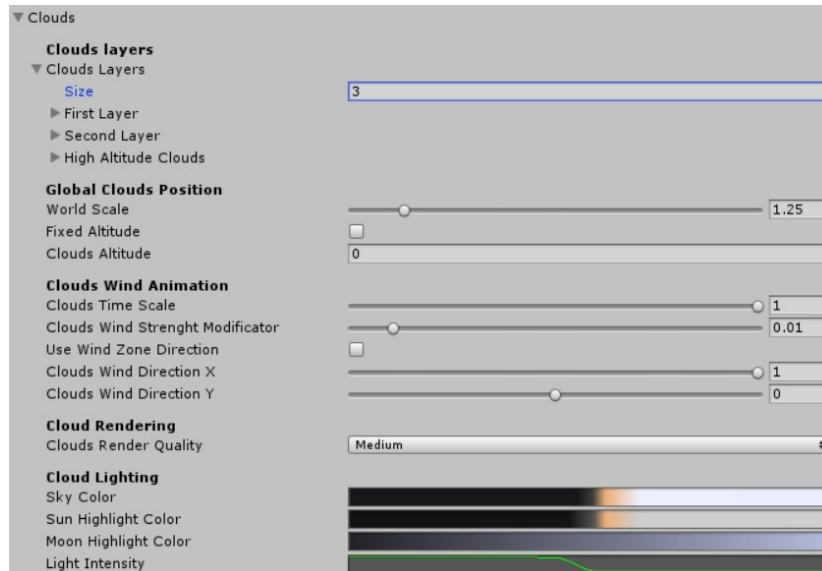
The Clouds Render Quality will influence the performance hit of clouds dramaticly. Try to use "Low" settings for best performance. The visual difference should be not visible.

Also check each clouds layer "Quality" setting for better performance. With right setup you should be able to run enviro even on older graphic cards. All these settings can be changed on runtime. To apply changed values call:

```
EnviroSky.instance.InitClouds ();
```

If you want to apply settings in editor you just need to toggle (disable/enable) the EnviroSky component.

The clouds color, lighting influence and alpha is controlled in each weather type.
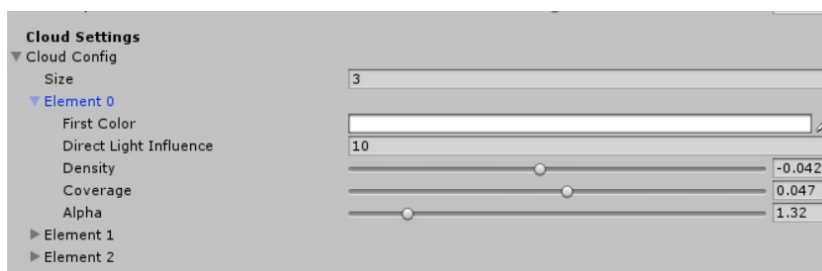
If you setup less layers in weather type than you got globaly the missing layer will be excluded and not rendered for this weather.

Weather (Enviro Zone)

The Enviro Zone component will control the currently played weather. You can add as many zones as you need. A default zone is already added to EnviroSky Prefab. The default zone will only be active if your player isn't in any other zone.
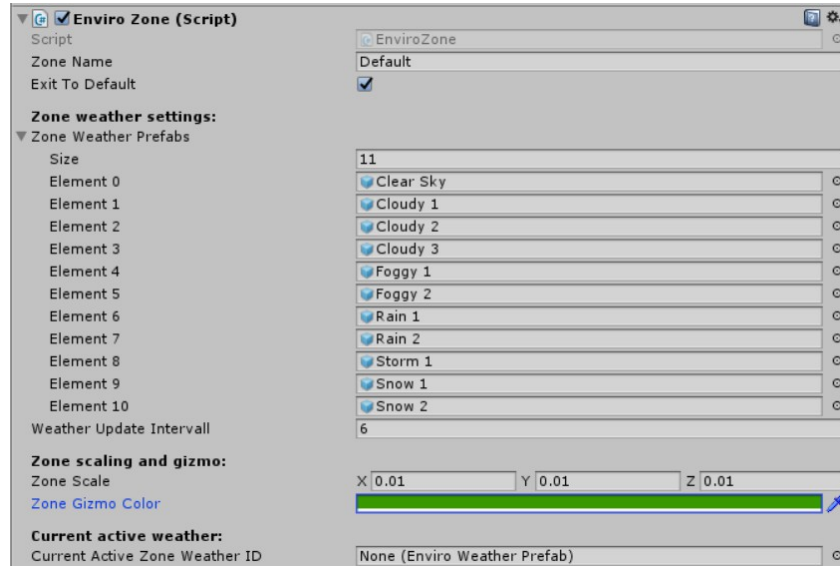
Assign all possible weather types for this zone to the list. The "Weather Update Interval" is the interval (GameTime Hours) Enviro tries to choose a new weather based on current season and possibilty rate.

Create your content

In this section you will learn how to create your own content with ENVIRO like   growing vegetation, creating your own weather-types or setting up the season   components and dynamic surface shaders.

Growing vegetation setup

*Let's start how to setup your own tree/bush/plant to grow! This look like a lot   of work, but when created you first trees/plants you will be able to create them in less than 5 minutes when you got all the models you want to use!*

1. *Create a new empty GameObject and change the name. ("GameObject" -> "Create Empty")*

2. *Add the "EnviroVegetationInstance" component . ("Component" -> "Enviro" -> "Vegetation Growth Instance")*

3. *Let's add our models! For this example I used the SpeedTree White Oak package. But you can use any models you like.*
   Just drag and drop the models from your "Project File List" into "Scene View" onto the GameObject you created in first step. Make sure that the models are child objects! When you got all models in scene, deactivate them! You should have something like this:
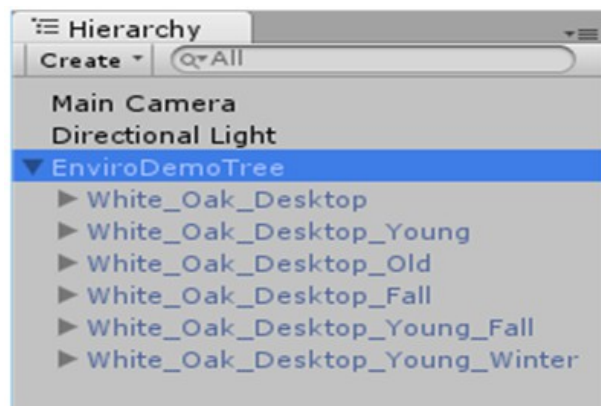
4. *Let's create our Grow Stages! Select your GameObject created in first step and click on "Grow Stages". In this example I want to have three stages! One for a young tree, one for an adult tree and one for a very old nearly dead tree. So I need to set the "Size" to "3"!*

5. *Now let's link the models you added in third step in the correct "Grow GameObject slots" for each season and stage. Also we setup the "Min Age Percent" for the second and third stage to let the system know after what age in percent the models should be switched to next stage. We choose 35 in second stage and 80 in third here. Also we define that in third stage the Tree shouldn't grow anymore. So we set "Grow Action" to "Stay"!*
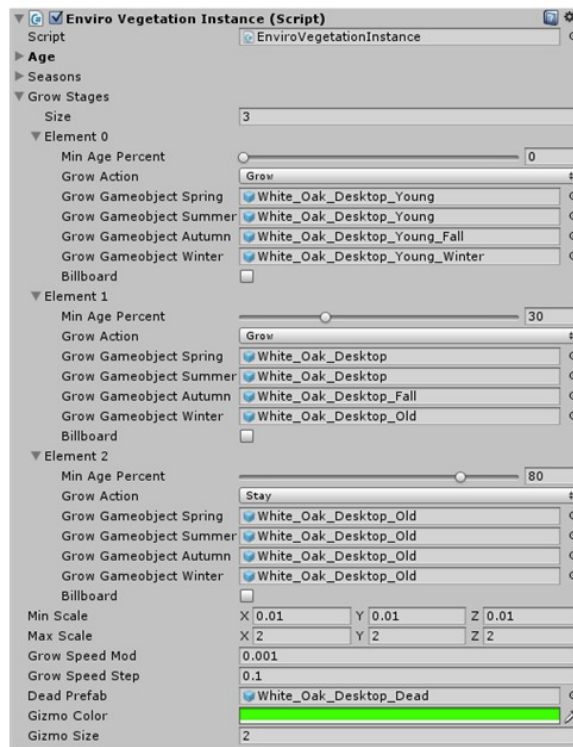
6. *We are done with the "Grow Stages" now. Now we setup "Age" settings. Click on "Age" and define the Max Age in hours, days and years. I choose to set it to 120 days. After this time the tree will die. I also check the "Loop" checkbox. This way a new tree will grow after the old tree died.*

7. *Next is "Seasons" Click on "Seasons" to open the season setting. Here you can define in which seasons the tree should grow and what action should executed if it can't grow in a season. In this example I want the tree to grow in all four seasons. So I doesn't change anything here. But if you may create a plant or anything like that you can activate the checkboxes in what seasons this plant can grow. Take a look into the example scene.*

8. *The last settings we want to setup are the Min and Max Scale and the Grow Speed Mod and Step! The "Min Scale" defines the start scale on birth. I choose 0.1 in all three axes. The "Max Scale" should be clear. The Tree can't get bigger than this value. I choose a value of 2 in all three axes. The "Grow Speed Mod" and "Step" values are the most important and hardest to setup. It all depends in what type of tree/plant you are creating and how old the tree/plant can get. The best way to find good values is to trial and error. Start the scene and try different values. In our case a "Grow Speed Mod" of 0,001 and a "Grow Speed Step" of 0,1 gives good results. In general the older the tree/plant can get, the lower the "Grow Speed Step" value should be.*

The plant/tree is ready for use now! But there is something you may want to add. When the tree dies it just vanish at the moment. So we may want to create a "Dead Prefab" that is spawned like we have in the sample scene.

Dead Tree Setup:

1. *Drag the model you want to use for this into scene view.(I use the winter version of the oak)*

2. *Add a "Rigidbody" to these and setup correct mass, also make sure that "Use gravity" is checked and freeze the "y" rotation in "Constraints".*

3. *Add a collider (for trees use a "Capsule Collider") as child and setup scale to match your tree also give it slight rotation in the direction it should fall.*

4. *Add an "AudioSource", assign a matching sound effect, make sure to check "Play on Awake".*

5. *Add a "Timed Object Destructor Script". You can find this in the Standard Assets Utility scripts from Unity. Setup the "Time Out" value.*

6. *Drag this into your "Project List" to save it as a prefab. And assign the prefab to your "Enviro tree/plant" you created in the "Dead Prefab" slot.*

Scripting

A few example for frequently used lines of code for your scripts to "communicate" with enviro.

General

Assign your player and/or camera and start enviro:
```
EnviroSky.instance.AssignAndStart (GameObject player, Camera Camera);
```

Start enviro in headless server mode:
```
EnviroSky.instance.StartAsServer();
```

Change focus on another player and/or camera:
```
EnviroSky.instance.ChangeFocus (GameObject player, Camera Camera);
```

Recalculate cloud's settings in runtime:
```
EnviroSky.instance.InitClouds ();
```

Time

Get current time:
```
int currentSecond = EnviroSky.instance.GameTime.Seconds;
int currentMinute = EnviroSky.instance.GameTime.Minutes;
int currentHour = EnviroSky.instance.GameTime.Hours;
int currentDay = EnviroSky.instance.GameTime.Days;
int currentYear = EnviroSky.instance.GameTime.Years;
float timeOfDay = EnviroSky.instance.internalHour;
```

Getcurrent sun and moon "time" based on position in sky (0-1):
```
float solarTime = EnviroSky.instance.GameTime.solarTime;
float lunarTime = EnviroSky.instance.GameTime.lunarTime;
```

Set time:
```
EnviroSky.instance.SetInternalTimeOfDay(float internalHour);
EnviroSky.instance.SetTime(int year,int day,int hour,int minute,int second);
EnviroSky.instance.SetTime(System.DateTime dateTime);
```

Seasons

Season enum:
```
SeasonVariables.Seasons.Spring
SeasonVariables.Seasons.Summer
SeasonVariables.Seasons.Autumn
SeasonVariables.Seasons.Winter
```

Get current season:
```
SeasonVariables.Seasons season = EnviroSky.instance.Seasons.currentSeasons;
```

Set current season:
```
EnviroSky.instance.Seasons.currentSeasons = SeasonVariables.Seasons.Spring;
```

Weather

Get current active weather:
```
EnviroWeatherPrefab  currentWeather = EnviroSky.instance.Weather.currentActiveWeatherID;
```

Get a List of all weather types:
```
List<EnviroWeatherPrefab> weatherTypes = EnviroSky.instance.Weather.WeatherTemplates;
```

Set current active weather with smooth transition:
```
EnviroSky.instance.ChangeWeather(int weatherID);
EnviroSky.instance.ChangeWeather(string weatherName);
```

Set current active weather without transition:
```
EnviroSky.instance.SetWeatherOverwrite(int id);
```

Get current active zone:
```
EnviroZone zone = EnviroSky.instance.Weather.currentActiveZone;
```

Events

You can listen to enviro's events do actions in your own scripts.

```csharp
EnviroSky.instance.OnWeatherChanged += (EnviroWeatherPrefab type) =>
{
    Debug.Log("Weather changed to: " + type.Name);
};

EnviroSky.instance.OnSeasonChanged += (SeasonVariables.Seasons season) =>
{
    Debug.Log("Season changed");
};

EnviroSky.instance.OnHourPassed += () =>
{
    Debug.Log("Hour Passed!");
};

EnviroSky.instance.OnDayPassed += () =>
{
    Debug.Log("New Day!");
};

EnviroSky.instance.OnYearPassed += () =>
{
    Debug.Log("New Year!");
};
```