

Time of Day

Generated by Doxygen 1.8.8

Mon Nov 10 2014 17:41:53



# Contents

<b>1</b>	<b>Main Page</b>	<b>1</b>
1.1	About . . . . .	1
1.2	Getting Started . . . . .	3
1.3	Day & Night Cycle . . . . .	3
1.4	Weather Manager . . . . .	4
1.5	Time Zone & Location Coordinates . . . . .	4
1.6	Ambient Light & Reflections . . . . .	4
1.7	Rendering Quality . . . . .	4
1.8	Performance Remarks . . . . .	5
1.9	Rendering Order . . . . .	5
1.10	Custom Shaders . . . . .	6
1.11	Networking . . . . .	6
1.12	Parameter Import & Export . . . . .	6
1.13	Example Scripts . . . . .	6
1.14	Frequently Asked Questions . . . . .	7
1.15	Contact Information . . . . .	8
1.16	Literature . . . . .	8
1.17	Changelog . . . . .	8
<b>2</b>	<b>Hierarchical Index</b>	<b>15</b>
2.1	Class Hierarchy . . . . .	15
<b>3</b>	<b>Class Index</b>	<b>17</b>
3.1	Class List . . . . .	17
<b>4</b>	<b>Class Documentation</b>	<b>19</b>
4.1	TOD_AmbientParameters Class Reference . . . . .	19
4.1.1	Detailed Description . . . . .	19
4.2	TOD_Animation Class Reference . . . . .	19
4.2.1	Detailed Description . . . . .	20
4.3	TOD_AtmosphereParameters Class Reference . . . . .	20
4.3.1	Detailed Description . . . . .	21

4.4	TOD_Camera Class Reference	21
4.4.1	Detailed Description	21
4.5	TOD_CloudParameters Class Reference	21
4.5.1	Detailed Description	22
4.6	TOD_Components Class Reference	22
4.6.1	Detailed Description	24
4.7	TOD_CycleParameters Class Reference	24
4.7.1	Detailed Description	25
4.8	TOD_DayParameters Class Reference	25
4.8.1	Detailed Description	26
4.9	TOD_FogParameters Class Reference	26
4.9.1	Detailed Description	26
4.10	TOD_LightParameters Class Reference	26
4.10.1	Detailed Description	27
4.11	TOD_MoonParameters Class Reference	27
4.11.1	Detailed Description	28
4.12	TOD_NightParameters Class Reference	28
4.12.1	Detailed Description	29
4.13	TOD_Parameters Class Reference	29
4.13.1	Detailed Description	29
4.14	TOD_PostEffectsBase Class Reference	29
4.14.1	Detailed Description	30
4.15	TOD_Rays Class Reference	30
4.15.1	Detailed Description	31
4.16	TOD_ReflectionParameters Class Reference	31
4.16.1	Detailed Description	31
4.17	TOD_Resources Class Reference	31
4.17.1	Detailed Description	32
4.18	TOD_Sky Class Reference	32
4.18.1	Detailed Description	35
4.18.2	Member Function Documentation	35
4.18.2.1	FakeHDR2LDR	35
4.18.2.2	FakeHDR2LDR	35
4.18.2.3	FakeHDR2LDR	36
4.18.2.4	FakeHDR2LDR	36
4.18.2.5	OrbitalToLocal	36
4.18.2.6	OrbitalToUnity	36
4.18.2.7	RenderToCubemap	37
4.18.2.8	RenderToCubemap	37
4.18.2.9	RenderToSH3	37

4.18.2.10 SampleAtmosphere . . . . .	37
4.18.2.11 SampleEquatorColor . . . . .	38
4.18.2.12 SampleFogColor . . . . .	38
4.18.2.13 SampleSkyColor . . . . .	38
4.19 TOD_StarParameters Class Reference . . . . .	38
4.19.1 Detailed Description . . . . .	39
4.20 TOD_SunParameters Class Reference . . . . .	39
4.20.1 Detailed Description . . . . .	39
4.21 TOD_Time Class Reference . . . . .	40
4.21.1 Detailed Description . . . . .	40
4.22 TOD_Weather Class Reference . . . . .	40
4.22.1 Detailed Description . . . . .	41
4.23 TOD_WorldParameters Class Reference . . . . .	41
4.23.1 Detailed Description . . . . .	42



# Chapter 1

## Main Page

### 1.1 About

Time of Day is a package to render realistic dynamic sky domes with day and night cycle, clouds, cloud shadows, weather types and physically based atmospheric scattering.

#### **Sky:**

- Physically based sky shading
- Rayleigh & Mie scattering
- Highly customizable
- Dynamic weather manager
- Sun and moon god rays (Unity Pro)

#### **Lighting:**

- Full PBR & HDR support
- Realtime Unity 5 ambient light
- Realtime Unity 5 reflections

#### **Clouds:**

- Dynamic wind speed & direction
- Configurable shape, color & scale
- Correctly projected cloud shadows

**Time & Location:**

- Dynamic day & night cycle
- Adjustable time progression curve
- Full longitude, latitude & time zone support
- Full Gregorian calendar support
- Rotating night sky
- Realistic moon position
- Continuous moon phases

**Performance & Requirements:**

- Extremely optimized shaders & scripts
- Zero dynamic memory allocations
- Supports shader model 2.0
- Supports all platforms
- Supports linear & gamma color space
- Supports forward & deferred rendering
- Supports HDR & LDR rendering
- Supports virtual reality hardware



[ [Forum Thread](#) | [Web Player](#) | [Documentation](#) ]

**You can expect a thoroughly documented, well-written and highly optimized code base.**

**All equations used in the shaders and scripts include references to the scientific papers they are based on.**

## 1.2 Getting Started

1. Add the sky dome to your scene:

- Drag the prefab "Time of Day/Prefabs/Sky Dome" into your scene
- Tweak the parameters until you are satisfied with the result

2. Move the sky dome to the camera position in every frame:

- Select your camera and add the Time of Day camera script (Component -> Time of Day -> Camera Main Script)
- Tag this camera game object as "MainCamera"

3. Render god rays on the main camera:

- Select your camera and add the Time of Day god ray script (Component -> Time of Day -> Camera God Rays)
- Tweak the parameters until you are satisfied with the result

**REMARK:** The camera script moves the sky dome directly before clipping the scene, guaranteeing that all other position updates have been processed. You should not move the sky dome in "LateUpdate" because this can cause minor differences in the sky dome position between frames when moving the camera.

## 1.3 Day & Night Cycle

The sky dome prefab has a script [TOD\\_Time](#) attached to it that manages the dynamic day & night cycle. Enabling and disabling this script enables and disables the automatic cycle.

The following parameters are being set by the script:

- TOD\_Sky.Cycle.Hour
- TOD\_Sky.Cycle.Day
- TOD\_Sky.Cycle.Month
- TOD\_Sky.Cycle.Year
- TOD\_Sky.Moon.Phase

The script [TOD\\_Time](#) offers a time curve property that can be modified via the Unity inspector to speed up or slow down certain parts of the day-night cycle. The X axis of the graph denotes the current internal time, which always progresses linearly. The Y axis of the graph denotes the time that is being set in the sky dome and is therefore visible to the player. That means the higher the inclination of the curve the faster this certain part of the day passes by.

## 1.4 Weather Manager

The script [TOD\\_Weather](#) can be used to automatically set various parameters of [TOD\\_Sky](#) according to weather presets.

The following parameters are being set by the script:

- `TOD_Sky.Atmosphere.Fogginess`
- `TOD_Sky.Clouds.Density`
- `TOD_Sky.Clouds.Sharpness`
- `TOD_Sky.Clouds.Brightness`

## 1.5 Time Zone & Location Coordinates

The [TOD\\_Sky.World](#) and [TOD\\_Sky.Cycle](#) parameter sections allow for configuration of the sky dome to represent the exact sun movement and day length of any location on the planet depending on Gregorian date, UTC/GMT time zone and geographic coordinates. It is important to set the correct time zone with the longitude and latitude of the location to guarantee consistent results with the real world. All of those parameters are completely optional though - if the sky dome should be used in a generic fantasy world they can simply be ignored and left at their default values.

## 1.6 Ambient Light & Reflections

Unity 5 introduced new ways to approximate ambient light and reflections. For a primer on the new features, watch the [Unite 2014 talk](#).

Time of Day offers full support for Unity 5 image-based ambient light and reflections. It can update both the per-scene ambient light and the per-scene reflection probe at runtime. Ambient light can be flat, trilight and spherical harmonics. The reflection probe is always a cubemap. Both contain approximations of the atmosphere in the top half and lerp to the configured ambient light color towards the bottom half. This can be looked at as the ground color since Time of Day cannot know what color to use for the ground by itself.

In addition to the per-scene ambient light and the per-scene reflection probe, Unity 5 offers statically baked reflection probes that can be scattered around a scene. Since those are baked in-editor, they do not change over the course of the day / night cycle. It is therefore recommended to only use them inside buildings or in outdoor scenes with somewhat static lighting conditions. Unity automatically falls back to the realtime per-scene ambient light and reflections that are calculated by Time of Day for all objects that do not have baked light probes nearby.

## 1.7 Rendering Quality

There are various different quality levels for both the sky dome and the cloud shader. Those quality settings can be configured both dynamically at runtime and directly in the Unity editor window using two inspector enums.

`TOD_CloudQuality`:

- Bumped offers complex cloud shading with dynamic density and cloud normal mapping

- Density offers simplified cloud shading with dynamic density but without normal mapping
- Fastest offers extremely simplified cloud shading with simplified cloud shape calculations

TOD\_MeshQuality:

- High tessellation sky dome (2562 verts) and moon (574 verts)
- Medium tessellation sky dome (642 verts) and moon (294 verts)
- Low tessellation sky dome (162 verts) and moon (148 verts)

For the best visual quality it is recommended to use Time of Day with the following Unity Pro setup:

- Linear color space
- HDR enabled on the main camera
- The following image effects on the main camera (in that order)
  1. "Image Effects -> Bloom and Glow -> Bloom" or "SE Natural Bloom & Dirty Lens" from the Asset Store
  2. "Image Effects -> Color Adjustments -> Tonemapping"
  3. "Image Effects -> Color Adjustments -> Color Correction" or "Amplify Color" from the Asset Store

## 1.8 Performance Remarks

- The size of a web player with just the sky dome is only around 200KB as most equations are evaluated dynamically
- All scripts and shaders are highly optimized and will not have a significant FPS impact on desktop computers
- Older mobile devices should switch to the cloud and dome quality settings that offer suitable performance
- Cloud shadows utilize a Unity projector and require another draw call for all objects they are projected on
- Reducing the texture resolution in the cloud texture import settings can increase performance on mobile

## 1.9 Rendering Order

All components of the sky dome are being rendered after the opaque but before the transparent meshes of your scene. That means that only areas of the sky dome that are not being occluded by any other geometry have to be rendered.

The rendering order of the sky dome components is the following:

- Transparent-500 Unity Skybox (if your scene uses one)
- Transparent-490 Space Dome (only at night)
- Transparent-480 Moon Mesh (only at night and if visible)
- Transparent-470 Atmosphere (if not manually disabled)

- Transparent-460 Sun Plane (only at day and if visible)
- Transparent-455 Clear Alpha (if god rays are enabled)
- Transparent-450 Cloud Layer (if not manually disabled)

This leads to 2-5 draw calls to render the complete sky dome depending on the scene setup.

## 1.10 Custom Shaders

The [TOD\\_Sky](#) script sets some global shader parameters that can be used in your custom shaders. For a complete list see the `TOD_Base.cginc` file. Any of those variables can be used in any shader by simply defining uniform variables with the same name, which will then automatically be set to the most recent values every frame. It is also possible to simply include `TOD_Base.cginc` to get access to all variables.

In addition to those base variables there is also `TOD_Scattering.cginc`, which offers functions to easily evaluate the scattering equations in custom shaders. The method `float4 ScatteringColor(float3 dir)` is the easiest to use since it automatically takes care of all the required calculations. It is recommended to call this function in a vertex shader and interpolate the color to the fragment or surface shader, but it can also be used in image effects to implement things like global fog.

## 1.11 Networking

- To network the cycle state, synchronize the property `TOD_Sky.Cycle.Ticks` of type `long`
- To network the moon phase, synchronize the variable `TOD_Sky.Moon.Phase` of type `float`
- To network the cloud movement, synchronize the property `TOD_Sky.Components.Animation.CloudUV` of type `Vector4`

## 1.12 Parameter Import & Export

It is possible to export custom presets via the "Export" button in the [TOD\\_Sky](#) inspector panel and import them on a different prefab or even in a different project via the "Import" button.

## 1.13 Example Scripts

The package comes with various example scripts to demonstrate sky dome interaction.

- `AudioAtDay` / `AudioAtNight` / `AudioAtWeather`: Fade audio sources in and out according to a time of day or a specific weather type

- ParticleAtDay / ParticleAtNight / ParticleAtWeather: Fade particle systems in and out according to a time of day or a specific weather type
- RenderAtDay / RenderAtNight / RenderAtWeather: Enable or disable renderer components according to a time of day or a specific weather type
- LightAtDay / LightAtNight / LightAtWeather: Fade light intensities in and out according to a time of day or a specific weather type
- DeviceTime: Automatically set the time of day to the device time on scene start

## 1.14 Frequently Asked Questions

Q: How can I get a [TOD\\_Sky](#) reference in my custom scripts?

- [TOD\\_Sky.Instance](#) keeps a static reference to the most recent sky dome that has been instantiated
- [TOD\\_Sky.Instances](#) keeps a static list of references to all sky domes that have been instantiated

Q: How can I use the sky dome with virtual reality devices like the Oculus Rift?

- Add the [TOD\\_Camera](#) script to one of the cameras (preferably the one that's being rendered first)
- The sky will render correctly without duplicate images or artifacts

Q: How can I render a cubemap or custom skybox at night?

- Select the shader "Time of Day/Space (Cube)" on the space material
- Assign your cubemap to the material

Q: How can I align the sky dome geographic directions with those of my scene?

- Rotate the sky dome around the y-axis such that the sun rises in the east of your scene
- Do not set the scale to negative values as this will lead to rendering artifacts

Q: How can I fix Z-fighting and sorting issues with the cloud shadows?

- Adjust the values for "Offset" directly in the shader code of the cloud shadow shaders

**REMARK:** Offset values have to be constants and can therefore only be adjusted directly in the shader code. Suitable values depend on the depth buffer resolution of the targeted platform and hardware. While the default values work in most scenarios, some scenes might require some further tweaking. If you are having issues with setting those values up correctly, please feel free to contact me.

Q: How can I disable some part of the sky dome?

- Disable any child game object to keep that specific part of the sky dome from rendering
- You can also disable any script on the parent game object to disable that specific functionality

**REMARK:** Always disable the child objects instead of their individual components like mesh renderers. The enabled states of components are being modified by the sky dome scripts, which will probably override your changes.

## 1.15 Contact Information

If you have any questions that cannot be answered using the FAQ or documentation feel free to contact me:

- In the official [forum thread](#) of the package
- Via [personal message](#) on the Unity community forums
- Via [Twitter](#)
- Via [my website](#)

**REMARK:** I should always be able to reply within 48 hours. If I did not reply after several days, please try using a different method to contact me as there might be an issue with the one you chose. If I am not available for multiple days I will always try to announce this beforehand in the official forum thread.

## 1.16 Literature

The following literature has been used to implement physically correct atmospheric scattering:

1. [Preetham, Shirley, Smits](#)
2. [Schafhitzel, Falk, Ertl](#)
3. [Bruneton, Neyret](#)
4. [Riley, Ebert, Kraus, Tessendorf, Hansen](#)
5. [Hoffman, Preetham](#)
6. [Nishita, Sirai, Tadamura, Nakamae](#)
7. [Nielsen, Christensen](#)

These papers are being referenced in the code in the following way:

See [N] page P equation (E)

Where the letters are being replaced according to this:

- N: Paper #
- P: Page #
- E: Equation # (if available)

## 1.17 Changelog

VERSION 2.3.1

-----

- Fixed errors if sky dome renderers or mesh filters were deleted (i.e. when running on a server)
- Fixed that `ScatteringColor(...)` in `TOD_Scattering.cginc` would add some stuff to its alpha value
- Fixed issues if the main camera of a scene changes after scene load

- Added TOD\_Sky.World.Horizon to specify whether or not to adjust the horizon to zero level
- Added TOD\_Sky.UpdateFog(), TOD\_Sky.UpdateAmbient() and TOD\_Sky.UpdateReflection() to API
- Added headless mode detection to skip some rendering calculations when running on a server
- Made TOD\_Sky.SampleAtmosphere(...) only include the moon halo if directLight is true
- Made the moon halo always fade out when the moon is below the horizon
- Made TOD\_Sky.Cycle.DateTime have DateTimeKind.Utc instead of DateTimeKind.Unspecified
- Made the fog color values clamp between 0 and 1 to avoid super bright glowing directional fog
- Changed TOD\_AdditiveColor and TOD\_MoonHaloColor in TOD\_Base.cginc to float3 (alpha is unused)
- Removed TOD\_Components.CameraTransform as it is no longer required

## VERSION 2.3.0

-----

- Fixed atmosphere banding towards nighttime by adding dithering from a lookup texture
- Fixed that SetupQualitySettings() would allocate 0.6kb of memory every frame
- Added TOD\_Animation.RandomInitialCloudUV to randomize the clouds at startup
- Added optional shader "Moon (Flat)" for a flatter moon shading
- Added TOD\_Sky.World.ZeroLevel to set the zero / water level of a scene
- Added TOD\_Camera.DomePosOffset to specify a sky dome position offset relative to the camera
- Added TOD\_Sky.Initialized to check whether or not the sky dome has been initialized
- Made RenderSettings.ambientLight get set in every ambient mode (for legacy shaders)
- Made fog, ambient and reflection really get updated every single frame if their update interval is 0
- Made sun and moon meshes fade out exactly at the horizon line
- Made the color of the sky dome beneath the horizon line fade to a darker tone towards the bottom
- Made the atmosphere shader additive (greatly improves moon / atmosphere blend)
- Made the night texture fade to black at daytime (due to the new additive atmosphere)
- Made the moon phase always be rotated towards the direction of the orbital path of the moon
- Made the sun texture converge towards a circle for very high sun mesh brightnesses
- Moved more enums to the global namespace and added the TOD\_ prefix
- Moved Cycle.Longitude, Cycle.Latitude and Cycle.UTC to the World parameter category
- Changed the returned alpha value of TOD\_Sky.SampleAtmosphere(...) to one
- Changed the returned alpha value of ScatteringColor(...) in TOD\_Scattering.cginc to one
- Renamed TOD\_Sky+Variables to TOD\_Sky+API (now contains all API methods and properties)
- Renamed TOD\_Sky+Quality to TOD\_Sky+Settings (now sets all project and scene settings)
- Renamed TOD\_SunShafts to TOD\_Rays (now handles god rays of both sun and moon)
- Renamed TOD\_Sky.SunShaftColor to TOD\_Sky.RayColor
- Renamed TOD\_Sky.Light.ShaftColoring to TOD\_Sky.Light.RayColoring
- Renamed TOD\_Sky.Sun.ShaftColor to TOD\_Sky.Sun.RayColor and added TOD\_Sky.Moon.RayColor
- Removed TOD\_Sky.World.HorizonOffset and TOD\_Sky.World.ViewerHeight (now covered by ZeroLevel)
- Removed TOD\_AmbientType.Hemisphere since it was removed from Unity 5 (use trilight instead)
- Removed clampAlpha parameter from TOD\_Sky.SampleAtmosphere(...)
- Replaced TOD\_Sky.Ambient.Exposure with Day.AmbientMultiplier and Night.AmbientMultiplier
- Replaced TOD\_Sky.Reflection.Exposure with Day.ReflectionMultiplier and Night.ReflectionMultiplier

## VERSION 2.2.0

-----

- Fixed a moon shader compilation error in Unity 5 on Windows
- Added support for Unity 5 ambient light modes (tricolor, hemisphere, spherical harmonics)
- Added support for Unity 5 realtime reflections (sky cubemap)
- Added TOD\_Sky.Stars.Position to specify whether or not to move the stars with the earth rotation
- Added TOD\_Sky.SampleAtmosphere(...) overload that ignores direct light
- Added TOD\_Sky.RenderToCubemap(...) with various overloads
- Added TOD\_Sky.RenderToSH3(...) with various overloads
- Added TOD\_Sky.SampleFogColor(), TOD\_Sky.SampleSkyColor() and TOD\_Sky.SampleEquatorColor()
- Added optional shader to project cubemaps onto the space object
- Removed TOD\_Sky.FogColor (access RenderSettings.fogColor instead)
- Removed TOD\_Sky.Stars.Density (directly adjust the texture instead)
- Moved all fog parameters to TOD\_Sky.Fog
- Moved all ambient light parameters to TOD\_Sky.Ambient
- Moved all reflection parameters to TOD\_Sky.Reflection
- Made audio example scripts set the volume in OnEnable()

## VERSION 2.1.1

-----

- Fixed various issues in gamma color space
- Fixed time not properly incrementing in some cases if TOD\_Time.ProgressDate was checked
- Fixed some inconsistencies with the light and cloud color calculations, leading to better results overall
- Fixed cloud shadow shape calculation being off for the lowest quality setting
- Fixed cloud UV world space adjustments being off for rotated sky domes
- Rescaled TOD\_Sky.Light.CloudColoring (custom prefabs have to be readjusted accordingly)
- Rescaled TOD\_Sky.Night.CloudMultiplier (custom prefabs have to be readjusted accordingly)

- Added TOD\_Sky.Day.CloudColor and TOD\_Sky.Night.CloudColor
- Added TOD\_Sky.Instance and TOD\_Sky.Instances to easily get the most recent sky or all skies in the scene
- Added TOD\_Animation.WorldSpaceCloudUV
- Added overloads of T() and ScatteringColor() that take distance into account to TOD\_Scattering.cginc
- Removed TOD\_Base.cginc include from TOD\_Scattering.cginc (now has to be included in the shader file)
- Brought the sun shaft image effect up to date
- Changed the code indentation policy (indent with tabs, align with spaces)
- Prepared more parts of the codebase for Unity 5

#### VERSION 2.1.0

-----

- Added XML export and import of the prefab parameters
- Added TOD\_Scattering.cginc that contains functions to sample the scattering color
- Added TOD\_Base.cginc that contains shader parameters and common transformations
- Added TOD\_World2Sky and TOD\_Sky2World shader matrices
- Added TOD\_Sky.Stars.Brightness parameter to make stars get affected by bloom image effects
- Added TOD\_Sky.LocalMoonDirection, TOD\_Sky.LocalSunDirection and TOD\_Sky.LocalLightDirection
- Added TOD\_Sky.Sun.MeshBrightness and TOD\_Sky.Moon.MeshBrightness
- Added TOD\_Sky.Sun.MeshContrast and TOD\_Sky.Moon.MeshContrast
- Added TOD\_Sky.Clouds.Glow to adjust the light source glow applied to the clouds
- Added TOD\_Sky.Atmosphere.FakeHDR to adjust the fake HDR mapping that is applied at dusk and dawn
- Added TOD\_Time.TimeCurve to specify a time progression curve for the day night cycle
- Added two new cloud textures (the old ones can be deleted if unused)
- Removed two unnecessary calls to InverseTransformDirection from TOD\_Sky.SampleAtmosphere
- Improved space texture to better work with the new brightness parameter
- Improved visual quality of the atmosphere when using HDR
- Improved cloud layer rendering
- Made TOD\_Sky.Cycle.DateTime accurate to one millisecond rather than one second
- Made camera scripts automatically search for the sky dome if no reference is set in the inspector
- Moved all moon parameters to TOD\_Sky.Moon.X (was TOD\_Sky.Night.MoonX and TOD\_Sky.Cycle.MoonX)
- Moved all sun parameters to TOD\_Sky.Sun.X (was TOD\_Sky.Day.SunX)

#### VERSION 2.0.9

-----

- Fixed time not getting incremented properly
- Fixed inaccuracies when progressing time and moon phase with extremely high frame rates
- Fixed inaccuracies when progressing time and moon phase with extremely fast time scales

#### VERSION 2.0.8

-----

- Fixed that sun and moon could visibly pop in and out if scaled extremely huge
- Fixed that the date would not get fully incremented for extremely fast time scales
- Fixed that the sun shafts could go through clouds
- Tweaked the TOD\_Sky.IsDay and TOD\_Sky.IsNight thresholds
- Replaced TOD\_Time.UpdateInterval with TOD\_Sky.Light.UpdateInterval (now only affects the light source)
- Prepared parts of the codebase for Unity 5 (specifically the new transform behaviour)

#### VERSION 2.0.7

-----

- Fixed an issue where the ambient light color would never fully lerp to the night value

#### VERSION 2.0.6

-----

- Replaced Day/Night.AmbientIntensity with Day/Night.AmbientColor to offer more customization options
- Added Light.AmbientColoring to adjust ambient light coloring at dusk and dawn
- Added example scripts to enable / disable lights in the scene at day / night / weather
- Added inspector variable to adjust the time update interval in TOD\_Time
- Added option to use the real-life moon position rather than the fake "opposite to sun" moon position
- Made all components of TOD\_Sky initialize before Start() so that they are accessible from other scripts
- Disabled the automatic light source shadow type adjustment so that the user can manually set it

#### VERSION 2.0.5

-----

- Changed cloud scale parameters from float to 2D vectors to define different scales in x and y direction
- Fixed TOD\_Camera always causing the scene to be edited if enabled
- Fixed cloud inconsistencies between linear and gamma color space
- Fixed moon halo disappearing in gamma color space and made the color alpha affect its visibility



- Fixed an issue where the demo mouse look script could overwrite previously imported Standard Assets
- Fixed possible sun and moon gimbal lock that could cause them to spin towards zenith
- Fixed sun shafts being too faint in some setups
- Improved overall lighting calculations
- Improved moon visuals
- Made the sky dome play nice with "depth only" clear flags
- Made the cloud coloring still darken the clouds even for very low values
- Made Components.Animation.CloudUV modulo with the cloud scale to avoid unnecessarily large values
- Added inspector variables to adjust sun shaft base color and sun shaft coloring
- Added the property Cycle.Ticks to get the time information as a long for easy network synchronization
- Added the property Cycle.DateTime to get the time information as a System.DateTime
- Added an inspector variable to set a minimum value for the light source height

## VERSION 2.0.4

-----

- Added a property for the atmosphere renderer component to TOD\_Components
- Added properties for all child mesh filter components to TOD\_Components
- Changed the quality settings to be adjustable at runtime via public enum inspector variables
- Merged the three prefabs into a single prefab as separate quality prefabs are no longer required
- Fixed the materials always showing up in version control
- Fixed the sky dome always causing the scene to be modified and the editor always asking to save on close
- Fixed the customized sky dome inspector not always looking like the default inspector
- Improved the performance of all cloud shaders by reducing interpolations from frag to vert
- Improved the visuals of all cloud shaders and streamlined their style
- Increased the default cloud texture import resolution to 1024x1024
- Added a white noise texture for future use

## VERSION 2.0.3

-----

- Fixed all issues with DX11 rendering in order to fully support DX11 from this point on

## VERSION 2.0.2

-----

- Fixed an issue where the image effect shaders could overwrite previously imported Standard Assets

## VERSION 2.0.1

-----

- Changed date and time organization to represent the valid Gregorian calendar
- Addressed issues with the Unity sun shaft image effect by providing a modified image effect
- Fixed clouds not correctly handling the planetary atmosphere curvature
- Fixed clouds not offsetting according to the world position of the sky dome
- Fixed cloud glow passing through even the thickest of clouds
- Fixed cloud shadow projection
- Fixed Light.Falloff not affecting the toggle point of the light position between sun and moon
- Automatically disable the corresponding shadows if Day/Night/Clouds.ShadowStrength is set to 0
- Removed Clouds.ShadowProjector toggle as it is no longer required
- Tweaked the old moon halo to not require an additional draw call and added it back in
- Made the sky dome position in world space add an offset to the cloud UV coordinates
- Added Light.Coloring to adjust the light coloring separate from the sky coloring
- Rescaled some parameters for easier use and tweaked their default values

## VERSION 2.0.0

-----

- Moved all documentation to Doxygen
- Renamed the folder "Sky Assets" to "Assets"
- Made the color space be detected automatically by default
- Reworked the sun texture and shader
- Allow light source intensities greater than one
- Reworked the way ambient light is being calculated
- Reworked the way light affects the atmosphere and clouds
- Improved all scattering calculations, especially the integral approximation
- Automatically disable space the game object at night
- Added a public method to sample the sky dome color in any viewing direction
- Added a fog bias parameter to lerp between zenith and horizon color
- Adjusted the atmosphere alpha calculation
- Added a parameter to easily adjust the scattering color
- Added shader parameters for the moon texture color and contrast
- Adjusted the render queue positions

- Removed the moon halo material as it is no longer required
- Added the physical scattering model to the night sky
- Greatly improved the weather system
- Added fog and contrast parameters to the atmosphere
- Restructured the parameter classes to be more intuitive to use
- Moved all component references into a separate class
- Made the sky presets be applied via editor script rather than separate prefabs
- Improved cloud shading and performance across the board
- Removed the cloud shading parameter
- Added cloud glow from the sun and moon
- Added sky and cloud tone multipliers to sun and moon
- Added viewer height and horizon offset parameters
- Slightly improved overall performance
- Replaced ambient intensity with two parameters for sun and moon
- Replaced the two directional lights with a single one that automatically follows either sun or moon

#### VERSION 1.7.3

-----

- Added two parameters "StarTiling" and "StarDensity" to the "Night" section
- Added "Offset -1, -1" to the cloud shadow shaders to avoid Z-fighting on some platforms
- Tweaked the cloud shader for more consistent results in linear and gamma color space
- Tweaked the moon texture to be a lot brighter by default, especially on mobile
- Tweaked the automatically calculated fog color to be similar to the horizon color
- Removed the property "Brightness" from the moon shader as it is no longer needed

#### VERSION 1.7.2

-----

- Fixed the ambient light calculation being too dark, even with high ambient light parameter values
- Added the properties "SunZenith" and "MoonZenith" to access sun and moon zenith angles in degrees
- Added a parameter "Halo" to adjust the moon halo intensity and made its color be derived from the light
- Changed several parameters to be clamped between 0 and 1
- Changed the name of the property "OrbitRadius" to "Radius"
- Tweaked the moon phase calculation of both moon mesh and moon halo
- Tweaked several default parameter values of the prefabs

#### VERSION 1.7.1

-----

- Changed the default cardinal direction axes of the sky dome (x axis is now west/east, z axis south/north)
- Removed the property "ZenithFactor" as it is no longer being used
- Moved all child object references into a separate toggleable section called "Children"
- Tweaked the default parameters of the prefabs (brightness, haziness, cloud color, moon light intensity)
- Tweaked the calculations of the moon light color, ambient light at night and cloud tone at night
- Tweaked the default sun and moon base color based on good real life approximations
- Tweaked the moon halo
- Renamed the parameter "ShadowAlpha" in "Clouds" to "ShadowStrength"
- Added the parameter "ShadowStrength" for the sun and moon lights

#### VERSION 1.7.0

-----

- Fixed an issue where the sun could incorrectly travel around the north, even though the location is in the northern hemisphere (Thanks Gregg!)
- Fixed an issue that led to the brightest parts of the sky dome being slightly too dark
- Fixed the automatically calculated fog color not being exactly the same as the horizon
- Added a name prefix to all components to prevent name collisions with other packages
- Added cloud shadows (can be disabled)
- Added UTC time zone support
- Added a parameter to configure the color of the light reflected by the moon
- Added parameters for wind direction in degrees and wind speed in knots
- Added an option to automatically adjust the ambient light color (disabled by default)
- Added a parameter to adjust the sun's light color
- Added a plane with an additive shader at the sun's position to always render a circular sun
- Added dynamic cloud shape adjustments to the "Low" prefab (cloud weather types will now also work)
- Added shading calculations to the "Low" and "Medium" prefabs
- Improved the performance of "Low" prefab by reducing the vertex count
- Improved the performance of "Low" prefab by removing the moon halo for that prefab by default
- Improved the cloud shading of the "High" prefab
- Improved the visual quality of the weather presets
- Improved the calculation of the sun's position
- Changed the automatic fog color adjustment to be disabled by default
- Changed the moon halo to adjust according to the moon phase

- Changed the name of the parameter from "Color" to "AdditiveColor" for both day and night
- Changed the cloud animation to support network synchronization
- Changed the default tiling of the stars texture to 1 (was 3)
- Changed the moon vertex count in all presets to scale with the device performance
- Removed the parameter "CloudColor" from "NightParameters" as it is now derived from the moon light color

## VERSION 1.6.1

-----

- Fixed an issue related to HDR rendering

## VERSION 1.6.0

-----

- Improved the visuals and functionality of the weather system (most METAR codes should now be possible to achieve visually)
- Improved performance of the moon halo shader
- Added official support for HDR rendering
- Replaced the sun mesh with implicit sun scattering in the atmosphere layer to reduce dome vertex count, draw calls and pixel overdraw
- Added an additional quality level (now Low/Medium/High instead of Desktop/Mobile)
- Added sky dome presets from various locations around the globe for easier use
- Tweaked the wavelength constants a little to allow for a wider range of sun coloring adjustments

## VERSION 1.5.1

-----

- Fixed an issue causing a missing sun material in the mobile prefab

## VERSION 1.5.0

-----

- Enabled mip mapping of the stars texture by default to avoid flickering
- Added support for using custom skyboxes at night (see readme for details)
- Greatly improved the parametrization of the sun color influence at sunrise and sunset
- Added internal pointers to commonly used components for faster access
- Split the sun and moon parameters into their own property classes
- Adjusted the cloud shading calculation to keep it from darkening some clouds too much
- Adjusted the color wavelengths to produce a more realistic blue color of the sky by default
- Made the moon phase influence the intensity of the sunlight reflected by the moon
- Replaced the lens flares with custom halo shaders that are correctly being occluded by clouds
- Enabled the new halo effects on mobile
- Moved all shaders into a "Time of Day" category
- Added a basic weather manager with three weather types

## VERSION 1.4.0

-----

- Added "Fog { Mode Off }" to the shaders to properly ignore fog
- Added the parameter "Night Cloud Color" to render clouds at night
- Added the parameter "Night Haze Color" to render some haze at night
- Added the parameter "Night Color" to add some color to the night sky
- Renamed the parameter "Haze" to "Haziness"
- Renamed the parameter "Sky Tone" to "Brightness"
- Renamed the properties "Day" and "Night" to "IsDay" and "IsNight"
- Restructured all sky parameters into groups
- Improved the sun lens flare texture
- Improved the stars texture
- Fixed a rendering artifact at the horizon for low haziness values
- Made the scattering calculation in gamma space look identical to linear space

## VERSION 1.3.0

-----

- Greatly improved performance on mobile devices
- Greatly improved sunset and sunrise visual quality
- Added a parameter to control how strongly the sun color affects the sky color
- Added realistic sun and moon lens flare effects
- Added two additional cloud noise textures
- Improved handling of latitude and longitude
- Made the sky dome render correctly independent of its rotation

## VERSION 1.2.0

-----

- Fixed some bugs regarding linear vs. gamma space rendering
- Fixed some issues with the horizon fadeout
- Adjusted sun and moon size
- Optimized sun and fog color calculation
- Greatly improved visual quality of the cloud system
- Added parameter to control cloud tone, allowing for dark clouds
- Added improved stars texture at night
- Added parameter to control the sun color falloff speed

VERSION 1.1.0

-----

- First public release on the Asset Store

VERSION 1.0.0

-----

- First private release for internal use

## Chapter 2

# Hierarchical Index

### 2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

MonoBehaviour	
TOD_Animation . . . . .	19
TOD_Camera . . . . .	21
TOD_Components . . . . .	22
TOD_PostEffectsBase . . . . .	29
TOD_Rays . . . . .	30
TOD_Resources . . . . .	31
TOD_Sky . . . . .	32
TOD_Sky . . . . .	32
TOD_Sky . . . . .	32
TOD_Sky . . . . .	32
TOD_Time . . . . .	40
TOD_Weather . . . . .	40
TOD_AmbientParameters . . . . .	19
TOD_AtmosphereParameters . . . . .	20
TOD_CloudParameters . . . . .	21
TOD_CycleParameters . . . . .	24
TOD_DayParameters . . . . .	25
TOD_FogParameters . . . . .	26
TOD_LightParameters . . . . .	26
TOD_MoonParameters . . . . .	27
TOD_NightParameters . . . . .	28
TOD_Parameters . . . . .	29
TOD_ReflectionParameters . . . . .	31
TOD_StarParameters . . . . .	38
TOD_SunParameters . . . . .	39
TOD_WorldParameters . . . . .	41



## Chapter 3

# Class Index

### 3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

<a href="#">TOD_AmbientParameters</a>	
Parameters of the ambient mode . . . . .	19
<a href="#">TOD_Animation</a>	
Cloud animation class . . . . .	19
<a href="#">TOD_AtmosphereParameters</a>	
Parameters of the atmosphere . . . . .	20
<a href="#">TOD_Camera</a>	
Camera class . . . . .	21
<a href="#">TOD_CloudParameters</a>	
Parameters of the clouds . . . . .	21
<a href="#">TOD_Components</a>	
Component manager class . . . . .	22
<a href="#">TOD_CycleParameters</a>	
Parameters of the day and night cycle . . . . .	24
<a href="#">TOD_DayParameters</a>	
Parameters that are unique to the day . . . . .	25
<a href="#">TOD_FogParameters</a>	
Parameters of the fog mode . . . . .	26
<a href="#">TOD_LightParameters</a>	
Parameters of the light source . . . . .	26
<a href="#">TOD_MoonParameters</a>	
Parameters that are unique to the moon . . . . .	27
<a href="#">TOD_NightParameters</a>	
Parameters that are unique to the night . . . . .	28
<a href="#">TOD_Parameters</a>	
All parameters of the sky dome . . . . .	29
<a href="#">TOD_PostEffectsBase</a>	
Post effects base class . . . . .	29
<a href="#">TOD_Rays</a>	
God ray class . . . . .	30
<a href="#">TOD_ReflectionParameters</a>	
Parameters of the reflection mode . . . . .	31
<a href="#">TOD_Resources</a>	
Material and mesh wrapper class . . . . .	31
<a href="#">TOD_Sky</a>	
Main sky dome management class . . . . .	32
<a href="#">TOD_StarParameters</a>	
Parameters of the stars . . . . .	38

<a href="#">TOD_SunParameters</a>	
Parameters that are unique to the sun . . . . .	39
<a href="#">TOD_Time</a>	
Time iteration class . . . . .	40
<a href="#">TOD_Weather</a>	
Weather management class . . . . .	40
<a href="#">TOD_WorldParameters</a>	
Parameters of the world . . . . .	41



## Chapter 4

# Class Documentation

### 4.1 TOD\_AmbientParameters Class Reference

Parameters of the ambient mode.

#### Public Member Functions

- void [CheckRange](#) ()  
*Assures that all parameters are within a reasonable range.*

#### Public Attributes

- TOD\_AmbientType [Mode](#) = TOD\_AmbientType.Flat  
*Ambient light mode.*
- bool [Directional](#) = true  
*Whether or not to include directional light.*
- float [UpdateInterval](#) = 1.0f  
*Refresh interval of the ambient light probe in seconds.*
- int [Resolution](#) = 8  
*Resolution of the ambient light probe.*

#### 4.1.1 Detailed Description

Parameters of the ambient mode.

The documentation for this class was generated from the following file:

- TOD\_Parameters.cs

### 4.2 TOD\_Animation Class Reference

Cloud animation class.

#### Public Attributes

- float [WindDegrees](#) = 0.0f

Wind direction in degrees. = 0 for wind blowing in northern direction. = 90 for wind blowing in eastern direction. = 180 for wind blowing in southern direction. = 270 for wind blowing in western direction.

- float `WindSpeed` = 3.0f  
Speed of the wind that is acting upon the clouds.
- bool `WorldSpaceCloudUV` = true  
Whether or not to adjust the cloud coordinates when the sky dome moves.
- bool `RandomInitialCloudUV` = true  
Whether or not the clouds should be randomized at startup.

## Properties

- Vector4 `CloudUV` [get, set]  
Current cloud UV coordinates. Can be synchronized between multiple game clients to guarantee identical cloud positions.
- Vector4 `OffsetUV` [get]  
Current offset UV coordinates. Is being calculated from the sky dome world position.

### 4.2.1 Detailed Description

Cloud animation class.

Component of the sky dome parent game object.

The documentation for this class was generated from the following file:

- TOD\_Animation.cs

## 4.3 TOD\_AtmosphereParameters Class Reference

Parameters of the atmosphere.

### Public Member Functions

- void `CheckRange` ()  
Assures that all parameters are within a reasonable range.

### Public Attributes

- Color `ScatteringColor` = Color.white  
Artistic value to shift the scattering color of the atmosphere. Can be used to easily simulate alien worlds.
- float `RayleighMultiplier` = 1.0f  
[0, ∞] Intensity of the atmospheric Rayleigh scattering. Generally speaking this resembles the static scattering.
- float `MieMultiplier` = 1.0f  
[0, ∞] Intensity of the atmospheric Mie scattering. Generally speaking this resembles the angular scattering.
- float `Brightness` = 1.0f  
[0, ∞] Brightness of the atmosphere. This is being applied as a simple multiplier to the output color.
- float `Contrast` = 1.0f  
[0, ∞] Contrast of the atmosphere. This is being applied as a power of the output color.
- float `Directionality` = 0.6f  
[0, 1] Directionality factor that determines the size and sharpness of the glow around the light source.
- float `Haziness` = 0.6f

- *[0, 1] Intensity of the haziness of the sky at the horizon.*  
float **Fogginess** = 0.0f
- *[0, 1] Density of the fog covering the sky. This does not affect the RenderSettings fog that is being applied to other objects in the scene.*  
float **FakeHDR** = 0.5f
- *[0, 1] Amount of fake HDR at sunset.*

#### 4.3.1 Detailed Description

Parameters of the atmosphere.

The documentation for this class was generated from the following file:

- TOD\_Parameters.cs

## 4.4 TOD\_Camera Class Reference

Camera class.

### Public Member Functions

- void **DoDomeScaleToFarClip** ()
- void **DoDomePosToCamera** ()

### Public Attributes

- **TOD\_Sky** sky  
*Sky dome reference inspector variable. Will automatically be searched in the scene if not set in the inspector.*
- bool **DomePosToCamera** = true  
*Automatically move the sky dome to the camera position in OnPreCull().*
- Vector3 **DomePosOffset** = Vector3.zero  
*The sky dome position offset relative to the camera.*
- bool **DomeScaleToFarClip** = true  
*Automatically scale the sky dome to the camera far clip plane in OnPreCull().*
- float **DomeScaleFactor** = 0.95f  
*The sky dome scale factor relative to the camera far clip plane.*

#### 4.4.1 Detailed Description

Camera class.

Component of the main camera of the scene to move and scale the sky dome.

The documentation for this class was generated from the following file:

- TOD\_Camera.cs

## 4.5 TOD\_CloudParameters Class Reference

Parameters of the clouds.

## Public Member Functions

- void [CheckRange](#) ()  
*Assures that all parameters are within a reasonable range.*

## Public Attributes

- float [Density](#) = 1.0f  
*[0, ∞] Density multiplier of the clouds.  
= 0 no clouds.  
> 0 thicker clouds that are less transparent.*
- float [Sharpness](#) = 3.0f  
*[0, ∞] Sharpness multiplier of the clouds.  
= 0 one giant cloud.  
> 0 several smaller clouds.*
- float [Brightness](#) = 1.0f  
*[0, ∞] Brightness multiplier of the clouds.  
= 0 black clouds.  
> 0 brighter clouds.*
- float [Glow](#) = 1.0f  
*[0, ∞] Glow multiplier of the clouds.*
- float [ShadowStrength](#) = 0.0f  
*[0, 1] Opacity of the cloud shadows.*
- Vector2 [Scale1](#) = new Vector2(3, 3)  
*[1, ∞] Scale of the first clouds.*
- Vector2 [Scale2](#) = new Vector2(7, 7)  
*[1, ∞] Scale of the second clouds.*

### 4.5.1 Detailed Description

Parameters of the clouds.

The documentation for this class was generated from the following file:

- TOD\_Parameters.cs

## 4.6 TOD\_Components Class Reference

Component manager class.

## Public Member Functions

- void [Initialize](#) ()  
*Initializes all component references.*

## Public Attributes

- GameObject [Sun](#) = null  
*Sun child game object reference.*
- GameObject [Moon](#) = null  
*Moon child game object reference.*

- GameObject [Atmosphere](#) = null  
*Atmosphere child game object reference.*
- GameObject [Clear](#) = null  
*Clear child game object reference.*
- GameObject [Clouds](#) = null  
*Clouds child game object reference.*
- GameObject [Space](#) = null  
*Space child game object reference.*
- GameObject [Light](#) = null  
*Light child game object reference.*
- GameObject [Projector](#) = null  
*Projector child game object reference.*
- Transform [DomeTransform](#)  
*Transform component of the sky dome game object.*
- Transform [SunTransform](#)  
*Transform component of the sun game object.*
- Transform [MoonTransform](#)  
*Transform component of the moon game object.*
- Transform [LightTransform](#)  
*Transform component of the light source game object.*
- Transform [SpaceTransform](#)  
*Transform component of the space game object.*
- Renderer [SpaceRenderer](#)  
*Renderer component of the space game object.*
- Renderer [AtmosphereRenderer](#)  
*Renderer component of the atmosphere game object.*
- Renderer [ClearRenderer](#)  
*Renderer component of the clear game object.*
- Renderer [CloudRenderer](#)  
*Renderer component of the cloud game object.*
- Renderer [SunRenderer](#)  
*Renderer component of the sun game object.*
- Renderer [MoonRenderer](#)  
*Renderer component of the moon game object.*
- MeshFilter [SpaceMeshFilter](#)  
*MeshFilter component of the space game object.*
- MeshFilter [AtmosphereMeshFilter](#)  
*MeshFilter component of the atmosphere game object.*
- MeshFilter [ClearMeshFilter](#)  
*MeshFilter component of the clear game object.*
- MeshFilter [CloudMeshFilter](#)  
*MeshFilter component of the cloud game object.*
- MeshFilter [SunMeshFilter](#)  
*MeshFilter component of the sun game object.*
- MeshFilter [MoonMeshFilter](#)  
*MeshFilter component of the moon game object.*
- Material [SpaceShader](#)  
*Main material of the space game object.*
- Material [AtmosphereShader](#)  
*Main material of the atmosphere game object.*
- Material [ClearShader](#)

*Main material of the clear game object.*

- Material [CloudShader](#)

*Main material of the cloud game object.*

- Material [SunShader](#)

*Main material of the sun game object.*

- Material [MoonShader](#)

*Main material of the moon game object.*

- Material [ShadowShader](#)

*Main material of the projector game object.*

- [Light LightSource](#)

*Light component of the light source game object.*

- [Projector ShadowProjector](#)

*Projector component of the shadow projector game object.*

- [TOD\\_Sky Sky](#)

*Sky component of the sky dome game object.*

- [TOD\\_Animation Animation](#)

*Animation component of the sky dome game object.*

- [TOD\\_Time Time](#)

*Time component of the sky dome game object.*

- [TOD\\_Weather Weather](#)

*Weather component of the sky dome game object.*

- [TOD\\_Resources Resources](#)

*Resource container component of the sky dome game object.*

- [TOD\\_Rays Rays](#)

*God ray component of the camera game object if available.*

#### 4.6.1 Detailed Description

Component manager class.

Component of the main camera of the scene.

The documentation for this class was generated from the following file:

- [TOD\\_Components.cs](#)

## 4.7 TOD\_CycleParameters Class Reference

Parameters of the day and night cycle.

### Public Member Functions

- void [CheckRange](#) ()

*Assures that all parameters are within a reasonable range.*

## Public Attributes

- float **Hour** = 12  
*[0, 24] Time of the day in hours.  
 = 0 at the start of the day.  
 = 12 at noon.  
 = 24 at the end of the day.*
- int **Day** = 1  
*[1, 28-31] Current day of the month.*
- int **Month** = 3  
*[1, 12] Current month of the year.*
- int **Year** = 2000  
*[1, 9999] Current year.*

## Properties

- System.DateTime **DateTime** [get, set]  
*All time information as a System.DateTime instance.*
- long **Ticks** [get, set]  
*All time information as a single long. Value corresponds to the System.DateTime.Ticks property.*

### 4.7.1 Detailed Description

Parameters of the day and night cycle.

The documentation for this class was generated from the following file:

- TOD\_Parameters.cs

## 4.8 TOD\_DayParameters Class Reference

Parameters that are unique to the day.

## Public Member Functions

- void **CheckRange** ()  
*Assures that all parameters are within a reasonable range.*

## Public Attributes

- Color **AdditiveColor** = new Color32(0, 0, 0, 255)  
*Artistic value for an additive color at day.*
- Color **AmbientColor** = new Color32(78, 97, 127, 255)  
*Color of the ambient light at day.*
- Color **CloudColor** = new Color32(255, 255, 255, 255)  
*Color of the clouds at night.*
- float **SkyMultiplier** = 1.0f  
*[0, 1] Sky opacity multiplier at day.*
- float **CloudMultiplier** = 1.0f  
*[0, 1] Cloud tone multiplier at day.*

- float [AmbientMultiplier](#) = 1.0f  
*[0, ∞] Brightness of ambient light at day.*
- float [ReflectionMultiplier](#) = 1.0f  
*[0, ∞] Brightness of reflected light at day.*

#### 4.8.1 Detailed Description

Parameters that are unique to the day.

The documentation for this class was generated from the following file:

- TOD\_Parameters.cs

### 4.9 TOD\_FogParameters Class Reference

Parameters of the fog mode.

#### Public Member Functions

- void [CheckRange](#) ()  
*Assures that all parameters are within a reasonable range.*

#### Public Attributes

- TOD\_FogType [Mode](#) = TOD\_FogType.Color  
*Fog color mode.*
- bool [Directional](#) = true  
*Whether or not to include directional light.*
- float [UpdateInterval](#) = 0.0f  
*Refresh interval of the fog color in seconds.*
- float [HeightBias](#) = 0.1f  
*[0, 1] Fog color sampling height.  
= 0 fog is atmosphere color at horizon.  
= 1 fog is atmosphere color at zenith.*

#### 4.9.1 Detailed Description

Parameters of the fog mode.

The documentation for this class was generated from the following file:

- TOD\_Parameters.cs

### 4.10 TOD\_LightParameters Class Reference

Parameters of the light source.

#### Public Member Functions

- void [CheckRange](#) ()  
*Assures that all parameters are within a reasonable range.*



## Public Attributes

- float [UpdateInterval](#) = 0.0f  
*Light source position update interval in seconds. Zero means every frame.*
- float [MinimumHeight](#) = 0.0f  
*[0, 1] Controls how low the light source is allowed to go.  
= -1 light source can go as low as it wants.  
= 0 light source will never go below the horizon.  
= +1 light source will never leave zenith.*
- float [Falloff](#) = 0.75f  
*[0, 1] Controls how fast the sun color falls off. This is especially visible during sunset and sunrise.*
- float [Coloring](#) = 0.75f  
*[0, 1] Controls how strongly the light color is being affected by sunset and sunrise.*
- float [SkyColoring](#) = 0.5f  
*[0, 1] Controls how strongly the sun color affects the atmosphere color. This is especially visible during sunset and sunrise.*
- float [CloudColoring](#) = 0.75f  
*[0, 1] Controls how strongly the sun color affects the cloud color. This is especially visible during sunset and sunrise.*
- float [RayColoring](#) = 0.75f  
*[0, 1] Controls how strongly the god ray color is being affected by sunset and sunrise.*
- float [AmbientColoring](#) = 0.5f  
*[0, 1] Controls how strongly the ambient color is being affected by sunset and sunrise.*

### 4.10.1 Detailed Description

Parameters of the light source.

The documentation for this class was generated from the following file:

- TOD\_Parameters.cs

## 4.11 TOD\_MoonParameters Class Reference

Parameters that are unique to the moon.

## Public Member Functions

- void [CheckRange](#) ()  
*Assures that all parameters are within a reasonable range.*

## Public Attributes

- Color [LightColor](#) = new Color32(181, 204, 255, 255)  
*Color of the light emitted by the moon.*
- Color [MeshColor](#) = new Color32(255, 233, 200, 255)  
*Color of the moon material.*
- Color [RayColor](#) = new Color32(81, 104, 155, 50)  
*Color of the god rays cast by the moon.*
- Color [HaloColor](#) = new Color32(81, 104, 155, 25)  
*Color of the moon halo.*
- float [MeshSize](#) = 1.0f

- [0, ∞] Size of the moon mesh in degrees.*
- float **MeshBrightness** = 1.0f
  - [0, ∞] Brightness of the moon mesh.*
- float **MeshContrast** = 1.0f
  - [0, ∞] Contrast of the moon mesh.*
- float **LightIntensity** = 0.1f
  - [0, ∞] Intensity of the moon light source.*
- float **ShadowStrength** = 1.0f
  - [0, 1] Opacity of the object shadows dropped by the moon light source.*
- float **Phase** = 0.0f
  - [-1, 1] Phase of the moon.*
  - = 0 full moon.*
  - ± 1 no moon.*
- TOD\_MoonPositionType **Position** = TOD\_MoonPositionType.OppositeToSun
  - Type of the moon position calculation.*

#### 4.11.1 Detailed Description

Parameters that are unique to the moon.

The documentation for this class was generated from the following file:

- TOD\_Parameters.cs

## 4.12 TOD\_NightParameters Class Reference

Parameters that are unique to the night.

### Public Member Functions

- void **CheckRange** ()
  - Assures that all parameters are within a reasonable range.*

### Public Attributes

- Color **AdditiveColor** = new Color32(0, 0, 0, 255)
  - Artistic value for an additive color at night.*
- Color **AmbientColor** = new Color32(50, 62, 81, 255)
  - Color of the ambient light at night.*
- Color **CloudColor** = new Color32(47, 73, 137, 255)
  - Color of the clouds at night.*
- float **SkyMultiplier** = 0.1f
  - [0, 1] Sky opacity multiplier at night.*
- float **CloudMultiplier** = 0.1f
  - [0, 1] Cloud tone multiplier at night.*
- float **AmbientMultiplier** = 1.0f
  - [0, ∞] Brightness of ambient light at night.*
- float **ReflectionMultiplier** = 1.0f
  - [0, ∞] Brightness of reflected light at night.*

#### 4.12.1 Detailed Description

Parameters that are unique to the night.

The documentation for this class was generated from the following file:

- [TOD\\_Parameters.cs](#)

### 4.13 TOD\_Parameters Class Reference

All parameters of the sky dome.

#### Public Member Functions

- **TOD\_Parameters** ([TOD\\_Sky](#) sky)
- void **ToSky** ([TOD\\_Sky](#) sky)

#### Public Attributes

- [TOD\\_CycleParameters](#) **Cycle**
- [TOD\\_WorldParameters](#) **World**
- [TOD\\_AtmosphereParameters](#) **Atmosphere**
- [TOD\\_DayParameters](#) **Day**
- [TOD\\_NightParameters](#) **Night**
- [TOD\\_SunParameters](#) **Sun**
- [TOD\\_MoonParameters](#) **Moon**
- [TOD\\_LightParameters](#) **Light**
- [TOD\\_StarParameters](#) **Stars**
- [TOD\\_CloudParameters](#) **Clouds**
- [TOD\\_FogParameters](#) **Fog**
- [TOD\\_AmbientParameters](#) **Ambient**
- [TOD\\_ReflectionParameters](#) **Reflection**

#### 4.13.1 Detailed Description

All parameters of the sky dome.

The documentation for this class was generated from the following file:

- [TOD\\_Parameters.cs](#)

### 4.14 TOD\_PostEffectsBase Class Reference

Post effects base class.

#### Public Attributes

- [TOD\\_Sky](#) sky = null

*Sky dome reference inspector variable. Will automatically be searched in the scene if not set in the inspector.*

#### 4.14.1 Detailed Description

Post effects base class.

Based on PostEffectsBase from the default Unity image effects. Extended for image effects that depend on a [TOD\\_Sky](#) reference.

The documentation for this class was generated from the following file:

- [TOD\\_PostEffectsBase.cs](#)

### 4.15 TOD\_Rays Class Reference

God ray class.

#### Public Types

- enum [ResolutionType](#) { **Low**, **Normal**, **High** }  
*Resolutions for the god rays. High is full, Normal is half and Low is quarter the screen resolution.*
- enum [BlendModeType](#) { **Screen**, **Add** }  
*Methods to blend the god rays with the image.*

#### Public Attributes

- [ResolutionType](#) [Resolution](#) = ResolutionType.Normal  
*Inspector variable to define the god ray rendering resolution.*
- [BlendModeType](#) [BlendMode](#) = BlendModeType.Screen  
*Inspector variable to define the god ray rendering blend mode.*
- int [BlurIterations](#) = 2  
*Inspector variable to define the number of blur iterations to be performed.*
- float [BlurRadius](#) = 2  
*Inspector variable to define the radius to blur filter applied to the god rays.*
- float [Intensity](#) = 1  
*Inspector variable to define the intensity of the god rays.*
- float [MaxRadius](#) = 0.5f  
*Inspector variable to define the maximum radius of the god rays.*
- bool [UseDepthTexture](#) = true  
*Inspector variable to define whether or not to use the depth buffer. If enabled, requires the target platform to allow the camera to create a depth texture. Unity always creates this depth texture if deferred lighting is enabled. Otherwise this script will enable it for the camera it is attached to. If disabled, requires all shaders writing to the depth buffer to also write to the frame buffer alpha channel. Only the frame buffer alpha channel will then be used to check for ray blockers in the image effect. However, a lot of the built-in Unity shaders do not write correct alpha values for legacy reasons. It is unknown when this will be fixed, which is why it is recommended to use a depth texture in most cases.*
- Shader [GodRayShader](#) = null  
*Inspector variable pointing to the god ray rendering shader.*
- Shader [ScreenClearShader](#) = null  
*Inspector variable pointing to the clear rendering shader.*

### 4.15.1 Detailed Description

God ray class.

Component of the main camera of the scene to render god rays. Based on the sun shafts from the default Unity image effects. Extended to get the god ray color from [TOD\\_Sky](#) and properly handle transparent meshes like clouds.

The documentation for this class was generated from the following file:

- [TOD\\_Rays.cs](#)

## 4.16 TOD\_ReflectionParameters Class Reference

Parameters of the reflection mode.

### Public Member Functions

- void [CheckRange](#) ()  
*Assures that all parameters are within a reasonable range.*

### Public Attributes

- TOD\_ReflectionType [Mode](#) = TOD\_ReflectionType.None  
*Reflection cubemap mode.*
- bool [Directional](#) = true  
*Whether or not to include directional light.*
- float [UpdateInterval](#) = 1.0f  
*Refresh interval of the reflection cubemap in seconds.*
- int [Resolution](#) = 8  
*Resolution of the reflection cubemap.*

### 4.16.1 Detailed Description

Parameters of the reflection mode.

The documentation for this class was generated from the following file:

- [TOD\\_Parameters.cs](#)

## 4.17 TOD\_Resources Class Reference

Material and mesh wrapper class.

### Public Attributes

- Mesh **Quad**
- Mesh **SphereHigh**
- Mesh **SphereMedium**
- Mesh **SphereLow**
- Mesh **IcosphereHigh**

- Mesh **IcosphereMedium**
- Mesh **IcosphereLow**
- Mesh **HalfIcosphereHigh**
- Mesh **HalfIcosphereMedium**
- Mesh **HalfIcosphereLow**
- Material **CloudMaterialBumped**
- Material **CloudMaterialDensity**
- Material **CloudMaterialFastest**
- Material **ShadowMaterialBumped**
- Material **ShadowMaterialDensity**
- Material **ShadowMaterialFastest**
- Material **SpaceMaterial**
- Material **AtmosphereMaterial**
- Material **SunMaterial**
- Material **MoonMaterial**
- Material **ClearMaterial**

#### 4.17.1 Detailed Description

Material and mesh wrapper class.

Component of the sky dome parent game object.

The documentation for this class was generated from the following file:

- TOD\_Resources.cs

### 4.18 TOD\_Sky Class Reference

Main sky dome management class.

#### Public Member Functions

- Vector3 [OrbitalToUnity](#) (float radius, float theta, float phi)  
*Convert spherical coordinates to cartesian coordinates.*
- Vector3 [OrbitalToLocal](#) (float theta, float phi)  
*Convert spherical coordinates to cartesian coordinates.*
- Color [SampleAtmosphere](#) (Vector3 direction, bool directLight=true)  
*Sample atmosphere colors from the sky dome.*
- SphericalHarmonics3 [RenderToSH3](#) (int size, float exposure=1, bool directLight=true)  
*Render the sky dome to 3rd order spherical harmonics.*
- Cubemap [RenderToCubemap](#) (int size, int faceMask=63, float exposure=1, bool directLight=true)  
*Render the sky dome to a cubemap.*
- void [RenderToCubemap](#) (Cubemap cubemap, int faceMask=63, float exposure=1, bool directLight=true)  
*Render the sky dome to a cubemap.*
- Color [SampleFogColor](#) (bool directLight=true)  
*Calculate the fog color.*
- Color [SampleSkyColor](#) (float exposure=1)  
*Calculate the sky color.*
- Color [SampleEquatorColor](#) (float exposure=1)  
*Calculate the equator color.*
- void [UpdateFog](#) ()

Update the RenderSettings fog color according to [TOD\\_FogParameters](#).

- void [UpdateAmbient](#) ()

Update the RenderSettings ambient light according to [TOD\\_AmbientParameters](#).

- void [UpdateReflection](#) ()

Update the RenderSettings reflection probe according to [TOD\\_ReflectionParameters](#).

- Color [FakeHDR2LDR](#) (Color color)
- Color [FakeHDR2LDR](#) (Color color, float factor)
- Vector3 [FakeHDR2LDR](#) (Vector3 vector)
- Vector3 [FakeHDR2LDR](#) (Vector3 vector, float factor)

## Public Attributes

- TOD\_ColorSpaceDetection [UnityColorSpace](#) = TOD\_ColorSpaceDetection.Auto

Inspector variable to adjust the color space.

- TOD\_CloudQualityType [CloudQuality](#) = TOD\_CloudQualityType.Bumped

Inspector variable to adjust the cloud quality.

- TOD\_MeshQualityType [MeshQuality](#) = TOD\_MeshQualityType.High

Inspector variable to adjust the mesh quality.

- [TOD\\_CycleParameters](#) Cycle

Inspector variable containing parameters of the day and night cycle.

- [TOD\\_WorldParameters](#) World

Inspector variable containing parameters of the world.

- [TOD\\_AtmosphereParameters](#) Atmosphere

Inspector variable containing parameters of the atmosphere.

- [TOD\\_DayParameters](#) Day

Inspector variable containing parameters of the day.

- [TOD\\_NightParameters](#) Night

Inspector variable containing parameters of the night.

- [TOD\\_SunParameters](#) Sun

Inspector variable containing parameters of the sun.

- [TOD\\_MoonParameters](#) Moon

Inspector variable containing parameters of the moon.

- [TOD\\_LightParameters](#) Light

Inspector variable containing parameters of the light source.

- [TOD\\_StarParameters](#) Stars

Inspector variable containing parameters of the stars.

- [TOD\\_CloudParameters](#) Clouds

Inspector variable containing parameters of the cloud layers.

- [TOD\\_FogParameters](#) Fog

Inspector variable containing parameters of the fog.

- [TOD\\_AmbientParameters](#) Ambient

Inspector variable containing parameters of the ambient light.

- [TOD\\_ReflectionParameters](#) Reflection

Inspector variable containing parameters of the reflection cubemap.

## Properties

- static List< [TOD\\_Sky](#) > [Instances](#) [get]  
All currently active sky dome instances.
- static [TOD\\_Sky](#) [Instance](#) [get]  
The most recently created sky dome instance.
- bool [Initialized](#) [get]  
Whether or not the sky dome was successfully initialized.
- [TOD\\_Components](#) [Components](#) [get]  
Contains references to all components.
- bool [IsDay](#) [get]  
Boolean to check if it is day.
- bool [IsNight](#) [get]  
Boolean to check if it is night.
- float [Radius](#) [get]  
Radius of the sky dome.
- float [Level](#) [get]  
Height level of the sky dome.
- float [Gamma](#) [get]  
Gamma value that is being used in the shaders.
- float [OneOverGamma](#) [get]  
Inverse of the gamma value ( $1 / \text{Gamma}$ ) that is being used in the shaders.
- float [LerpValue](#) [get]  
Falls off the darker the sunlight gets. Can for example be used to lerp between day and night values in shaders.  
= +1 at day  
= 0 at night.
- float [SunZenith](#) [get]  
Sun zenith angle in degrees.  
= 0 if the sun is exactly at zenith.  
= 180 if the sun is exactly below the ground.
- float [MoonZenith](#) [get]  
Moon zenith angle in degrees.  
= 0 if the moon is exactly at zenith.  
= 180 if the moon is exactly below the ground.
- float [HorizonAngle](#) [get]  
Horizon angle in degrees.  
= 90 if the horizon is exactly in the middle of the sky dome.  
= 180 if the horizon is exactly at the bottom of the sky dome.
- float [HorizonOffset](#) [get]  
Relative horizon offset.  
= 0 if the horizon is exactly in the middle of the sky dome.  
= 1 if the horizon is exactly at the bottom of the sky dome.
- float [HorizonLevel](#) [get]  
Absolute horizon height level in world space.
- float [LightZenith](#) [get]  
Currently active light source zenith angle in degrees.  
= 0 if the currently active light source (sun or moon) is exactly at zenith.  
= 90 if the currently active light source (sun or moon) is exactly at the horizon.
- float [LightIntensity](#) [get]  
Current light intensity.
- Vector3 [MoonDirection](#) [get]  
Moon direction vector in world space.
- Vector3 [SunDirection](#) [get]



- Sun direction vector in world space.*

  - Vector3 [LightDirection](#) [get]

*Current directional light vector in world space. Lerps between [TOD\\_Sky.SunDirection](#) and [TOD\\_Sky.MoonDirection](#) at dusk and dawn.*
- Vector3 [LocalMoonDirection](#) [get]

*Moon direction vector in sky dome object space.*
- Vector3 [LocalSunDirection](#) [get]

*Sun direction vector in sky dome object space.*
- Vector3 [LocalLightDirection](#) [get]

*Current directional light vector in sky dome object space. Lerps between [TOD\\_Sky.LocalSunDirection](#) and [TOD\\_Sky.LocalMoonDirection](#) at dusk and dawn.*
- Color [LightColor](#) [get]

*Current light color. Returns the color of [TOD\\_Sky.Components.LightSource](#).*
- Color [RayColor](#) [get]

*Current ray color.*
- Color [SunColor](#) [get]

*Current sun color.*
- Color [MoonColor](#) [get]

*Current moon color.*
- Color [MoonHaloColor](#) [get]

*Current moon halo color.*
- Color [CloudColor](#) [get]

*Current cloud color.*
- Color [AdditiveColor](#) [get]

*Current additive color.*
- Color [AmbientColor](#) [get]

*Current ambient color.*

### 4.18.1 Detailed Description

Main sky dome management class.

Component of the sky dome parent game object.

### 4.18.2 Member Function Documentation

#### 4.18.2.1 Color TOD\_Sky.FakeHDR2LDR ( Color color ) [inline]

Parameters

<i>color</i>	The HDR color.
--------------	----------------

Returns

The LDR color.

#### 4.18.2.2 Color TOD\_Sky.FakeHDR2LDR ( Color color, float factor ) [inline]

## Parameters

<i>color</i>	The HDR color.
<i>factor</i>	The amount of fake HDR to apply.

## Returns

The LDR color.

4.18.2.3 Vector3 TOD\_Sky.FakeHDR2LDR ( Vector3 *vector* ) [inline]

## Parameters

<i>vector</i>	The HDR vector.
---------------	-----------------

## Returns

The LDR vector.

4.18.2.4 Vector3 TOD\_Sky.FakeHDR2LDR ( Vector3 *vector*, float *factor* ) [inline]

## Parameters

<i>vector</i>	The HDR vector.
<i>factor</i>	The amount of fake HDR to apply.

## Returns

The LDR vector.

4.18.2.5 Vector3 TOD\_Sky.OrbitalToLocal ( float *theta*, float *phi* ) [inline]

Convert spherical coordinates to cartesian coordinates.

## Parameters

<i>theta</i>	Spherical coordinates theta.
<i>phi</i>	Spherical coordinates phi.

## Returns

Unity position in local space.

4.18.2.6 Vector3 TOD\_Sky.OrbitalToUnity ( float *radius*, float *theta*, float *phi* ) [inline]

Convert spherical coordinates to cartesian coordinates.

## Parameters

<i>radius</i>	Spherical coordinates radius.
---------------	-------------------------------

<i>theta</i>	Spherical coordinates theta.
<i>phi</i>	Spherical coordinates phi.

**Returns**

Unity position in world space.

**4.18.2.7** `Cubemap TOD_Sky.RenderToCubemap ( int size, int faceMask = 63, float exposure = 1, bool directLight = true ) [inline]`

Render the sky dome to a cubemap.

**Parameters**

<i>size</i>	Resolution of the cubemap.
<i>faceMask</i>	Mask to determine which faces of the cubemap to update.
<i>exposure</i>	Camera exposure, determines brightness.
<i>directLight</i>	Whether or not to include direct light.

**Returns**

Freshly allocated cubemap with the sky rendered to it.

**4.18.2.8** `void TOD_Sky.RenderToCubemap ( Cubemap cubemap, int faceMask = 63, float exposure = 1, bool directLight = true ) [inline]`

Render the sky dome to a cubemap.

**Parameters**

<i>cubemap</i>	Cubemap target to render the sky to.
<i>faceMask</i>	Mask to determine which faces of the cubemap to update.
<i>exposure</i>	Camera exposure, determines brightness.
<i>directLight</i>	Whether or not to include direct light.

**4.18.2.9** `SphericalHarmonics3 TOD_Sky.RenderToSH3 ( int size, float exposure = 1, bool directLight = true ) [inline]`

Render the sky dome to 3rd order spherical harmonics.

**Parameters**

<i>size</i>	Number of samples to do in every direction.
<i>exposure</i>	Camera exposure, determines brightness.
<i>directLight</i>	Whether or not to include direct light.

**4.18.2.10** `Color TOD_Sky.SampleAtmosphere ( Vector3 direction, bool directLight = true ) [inline]`

Sample atmosphere colors from the sky dome.

**Parameters**

<i>direction</i>	View direction in world space.
<i>directLight</i>	Whether or not to include direct light.

**Returns**

Color of the atmosphere in the specified direction.

#### 4.18.2.11 Color TOD\_Sky.SampleEquatorColor ( float *exposure* = 1 ) [inline]

Calculate the equator color.

**Parameters**

<i>exposure</i>	Camera exposure, determines brightness.
-----------------	---

#### 4.18.2.12 Color TOD\_Sky.SampleFogColor ( bool *directLight* = true ) [inline]

Calculate the fog color.

**Parameters**

<i>directLight</i>	Whether or not to include direct light.
--------------------	---

#### 4.18.2.13 Color TOD\_Sky.SampleSkyColor ( float *exposure* = 1 ) [inline]

Calculate the sky color.

**Parameters**

<i>exposure</i>	Camera exposure, determines brightness.
-----------------	---

The documentation for this class was generated from the following files:

- TOD\_Sky+API.cs
- TOD\_Sky+Settings.cs
- TOD\_Sky.cs
- TOD\_Sky+Unity.cs

## 4.19 TOD\_StarParameters Class Reference

Parameters of the stars.

**Public Member Functions**

- void [CheckRange](#) ()  
*Assures that all parameters are within a reasonable range.*

**Public Attributes**

- float [Tiling](#) = 3.0f  
*[0, ∞] Texture tiling of the stars texture. Determines how often the texture is tiled accross the sky and therefore the size of the stars.*

- float [Brightness](#) = 3.0f  
*[0, ∞] Brightness of the stars.*
- TOD\_StarsPositionType [Position](#) = TOD\_StarsPositionType.Rotating  
*Type of the stars position calculation.*

#### 4.19.1 Detailed Description

Parameters of the stars.

The documentation for this class was generated from the following file:

- TOD\_Parameters.cs

## 4.20 TOD\_SunParameters Class Reference

Parameters that are unique to the sun.

### Public Member Functions

- void [CheckRange](#) ()  
*Assures that all parameters are within a reasonable range.*

### Public Attributes

- Color [LightColor](#) = new Color32(255, 243, 234, 255)  
*Color of the light emitted by the sun.*
- Color [MeshColor](#) = new Color32(255, 160, 25, 255)  
*Color of the sun material.*
- Color [RayColor](#) = new Color32(255, 243, 234, 255)  
*Color of the god rays cast by the sun.*
- float [MeshSize](#) = 1.0f  
*[0, ∞] Size of the sun mesh in degrees.*
- float [MeshBrightness](#) = 1.0f  
*[0, ∞] Brightness of the sun mesh.*
- float [MeshContrast](#) = 1.0f  
*[0, ∞] Contrast of the sun mesh.*
- float [LightIntensity](#) = 1.0f  
*[0, ∞] Intensity of the sun light source.*
- float [ShadowStrength](#) = 1.0f  
*[0, 1] Opacity of the object shadows dropped by the sun light source.*

#### 4.20.1 Detailed Description

Parameters that are unique to the sun.

The documentation for this class was generated from the following file:

- TOD\_Parameters.cs

## 4.21 TOD\_Time Class Reference

Time iteration class.

### Public Member Functions

- void [ApplyTimeCurve](#) ()  
*Apply changes made to TimeCurve.*
- void [AddTime](#) (float delta, bool adjust=false)  
*Adds delta to the sky dome time. Ensures that all values remain in a valid range.*
- void [AddMoon](#) (float delta)  
*Adds delta to the moon phase. Ensures that all values remain in a valid range.*
- void [CheckTimeRange](#) ()  
*Forces the time values to be within a valid range.*
- void [CheckMoonRange](#) ()  
*Forces the moon phase to be within a valid range.*

### Public Attributes

- float [DayLengthInMinutes](#) = 30  
*Day length inspector variable. Length of one day in minutes.*
- bool [ProgressTime](#) = true  
*Time progression inspector variable. Automatically updates Cycle.Hour if enabled.*
- bool [ProgressDate](#) = true  
*Date progression inspector variable. Automatically updates Cycle.Day if enabled.*
- bool [ProgressMoonPhase](#) = true  
*Moon phase progression inspector variable. Automatically updates Moon.Phase if enabled.*
- bool [UseTimeCurve](#) = false  
*Adjust the time progress according to the time curve.*
- AnimationCurve [TimeCurve](#) = AnimationCurve.Linear(0, 0, 24, 24)  
*Time of day progression curve. Can be used to make days longer and nights shorter.*

### 4.21.1 Detailed Description

Time iteration class.

Component of the sky dome parent game object.

The documentation for this class was generated from the following file:

- TOD\_Time.cs

## 4.22 TOD\_Weather Class Reference

Weather management class.

## Public Attributes

- float **FadeTime** = 10f  
*Fade time inspector variable. Time to fade from one weather type to the other.*
- TOD\_CloudType **Clouds** = TOD\_CloudType.Custom  
*Currently selected cloud type.*
- TOD\_WeatherType **Weather** = TOD\_WeatherType.Custom  
*Currently selected weather type.*

### 4.22.1 Detailed Description

Weather management class.

Component of the sky dome parent game object.

The documentation for this class was generated from the following file:

- TOD\_Weather.cs

## 4.23 TOD\_WorldParameters Class Reference

Parameters of the world.

## Public Member Functions

- void **CheckRange** ()  
*Assures that all parameters are within a reasonable range.*

## Public Attributes

- float **ZeroLevel** = 0  
*The zero / water level of the scene in world space. The horizon offset is automatically adjusted whenever the sky dome is above this level.*
- float **Latitude** = 0  
*[-90, 90] Latitude of your position in degrees.  
= -90 at the south pole.  
= 0 at the equator.  
= 90 at the north pole.*
- float **Longitude** = 0  
*[-180, 180] Longitude of your position in degrees.  
= -180 at 180 degrees in the west of Greenwich, England.  
= 0 at Greenwich, England.  
= 180 at 180 degrees in the east of Greenwich, England.*
- float **UTC** = 0  
*UTC/GMT time zone of the current location.  
= 0 for Greenwich, England.*
- TOD\_HorizonType **Horizon** = TOD\_HorizonType.Static  
*Type of the horizon offset.*

### 4.23.1 Detailed Description

Parameters of the world.

The documentation for this class was generated from the following file:

- TOD\_Parameters.cs