Time of Day

Generated by Doxygen 1.8.13

Contents

Chapter 1

Main Page

1.1 About

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

Sky:

- Physically based sky shading
- Rayleigh & Mie scattering
- Highly customizable
- Sun and moon god rays
- Aerial perspective

Lighting:

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

Clouds:

- Physically based cloud shading
- Semi-volumetric cloud layers
- Dynamically batched cloud billboards
- Adjustable wind speed & direction
- Configurable coverage and shading

- Correctly projected cloud shadows

Time & Location:

- Dynamic day & night cycle
- Adjustable time progression curve
- Full longitude, latitude & time zone support
- Full Gregorian calendar support
- Realistic sun position
- Realistic moon position and phase
- Realistic star constellations

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
- Supports single-pass stereo rendering

[Discussion | Demo | Documentation]

You can expect a thoroughly documented, well-written and highly optimized code base with references to the scientific papers it is based on.

1.2 Getting Started

1. Drag the prefab "Time of Day/Prefabs/Sky Dome" into your scene
2. Add the Time of Day camera script to your main camera (Component -> Time of Day -> Camera Main Script)
3. Add any of the optional camera scripts to your main camera (Component -> Time of Day -> ...)
4. Enable or disable any child objects of the sky dome you do or don't want to use
5. Tweak the sky dome inspector variables until you are satisfied with the result

NOTE: 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 script TOD_Time manages the dynamic day & night cycle.

The following parameters are being set by TOD_Time:

- TOD_Sky.Cycle.Hour
- TOD_Sky.Cycle.Day
- TOD_Sky.Cycle.Month
- TOD_Sky.Cycle.Year

It also offers a time curve 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 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.

The following events are fired by TOD_Time:

- TOD_Time.OnMinute
- TOD_Time.OnHour
- TOD_Time.OnDay
- TOD_Time.OnMonth
- TOD_Time.OnYear
- TOD_Time.OnSunrise
- TOD_Time.OnSunset

All events can be subscribed to by adding a void method with no arguments via the += operator. Note that all events are only fired once per frame, which means that even if more than one minute passes in a single frame, any methods that are subscribed to OnMinute will still only be called once.

1.4 Upgrade Guide

Doing a basic upgrade: (only recommended for minor releases as this does not delete or rename files)

1. Backup the current set of parameters using the export function in the context menu of the TOD_Sky script
2. Import the new version of Time of Day from the Asset Store
3. Check the TOD_Sky script on your prefab to see if there are any setup issues that have to be addressed

Doing a full reimport: (recommended for major releases or if the basic upgrade was unsuccessful)

1. Backup the current set of parameters using the export function in the context menu of the TOD_Sky script
2. Remove the old version of Time of Day from your project
3. Restart Unity
4. Import the new version of Time of Day from the Asset Store
5. Check the TOD_Sky script on your prefab to see if there are any setup issues that have to be addressed

1.5 Clouds

Time of Day supports dynamic volumetric cloud layers and handmade cloud billboards.

Cloud billboards can be customized to fit the specific art style of a project. This includes the individual billboard instances, positions, textures and meshes - just have a look at the "Billboards" child object of the sky dome.

Cloud layers are volumetric and their density is traced using a combination of noise octaves from the RGBA channels of a lookup textures. Both coverage and shading can be dynamically adjusted to fit any scenario.

1.6 Time Zone & Location Coordinates

The TOD_Sky.World and TOD_Sky.Cycle parameter sections allow for configuration of the sky dome to simulate the exact earth, sun and moon movement for any location on the planet depending on Gregorian date, UTC/GMT time zone and geographic coordinates. This for example allows to recreate eclipses just as they would occur in real life.

It is important to manually set the correct time zone offset (TOD_Sky.World.UTC) that fits the longitude and latitude parameters in order to use local time instead of UTC.

All of those parameters are completely optional - if the sky dome should be used in a generic fantasy world they can simply be ignored and left at their default values.

1.7 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 a realtime reflection probe at runtime. Ambient light can be disabled (i.e. not managed by Time of Day), a solid color, a gradient or spherical harmonics. Reflections can be disabled (i.e. not managed by Time of Day) or a cubemap. Both ambient light and reflections contain approximations of the atmosphere in the top half and lerp to the configured ambient light color towards the bottom half. This means the ambient light color set on the Time of Day prefab can be looked at as the ground color of the scene in those cases.

Time of Day also allows you to include some or all layers of your scene in the reflection probe bake process. This should be used with care since updating a reflection probe with various reflected objects is an expensive operation. For most scenes it should be fine to only render the sky dome to the realtime reflection probe by using "Skybox" clear flags and a "Nothing" culling mask.

1.8 Image Effects

Time of Day comes with a number of optional image effects.

God Rays:

- Added via "Components -> Time of Day -> Camera God Rays"
- Adds god rays to sun and moon in a single pass
- Correctly interacts with clouds
- Ray color is managed by TOD_Sky

Atmospheric Scattering:

- Added via "Components -> Time of Day -> Camera Atmospheric Scattering"
- Calculates atmospheric sky color and fog-like aerial perspective in a single pass
- Requires Unity 5 and shader model 3.0

Cloud Shadows:

- Added via "Components -> Time of Day -> Camera Cloud Shadows"
- Adds cloud shadows to all opaque objects in the scene
- Utilizes cutoff and fadeout parameters to prevent adding cloud shadows to shadowed areas
- Requires Unity 5

1.9 Rendering Quality

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

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

1.10 Performance Remarks

Desktop:

- 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
- Realtime reflections that include objects other than the sky dome can be expensive and should be used with care

Mobile:

- Older mobile devices should choose quality settings that offer suitable performance
- Lower star qualities significantly reduce the overall vertex count
- Lower cloud qualities significantly reduce the complexity of the per-pixel calculations

1.11 Rendering Order

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

- Unity depth pass
- Unity objects (background)
- Space
- Stars
- Sun
- Moon
- Atmosphere
- Unity objects (opaque)
- Unity image effects (depth)
- Clear
- Clouds
- Billboards
- Unity objects (transparent)
- Unity image effects (remaining)

This usually leads to 3-6 draw calls to render the complete sky dome, depending on the scene setup. All billboards can be dynamically batched on Unity Pro.

1.12 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 file TOD_Clouds.cginc contains cloud-related functions to render cloud layers and cloud shadows.

1.13 Networking

- To network date and time, synchronize the property TOD_Sky.Cycle.Ticks of type long
- To network cloud movement, synchronize the property TOD_Sky.Components.Animation.CloudUV of type Vector3

1.14 Parameter Import & Export

It is possible to import and export custom presets to XML via the "Import Parameters" and "Export Parameters" options in the context menu of the TOD_Sky script. Exported parameters can be imported in other projects or even loaded at runtime by using the appropriate API calls.

1.15 Example Scripts

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

- AudioAtDay / AudioAtNight / AudioAtTime: Fade audio sources in and out according to a time of day or a specific weather type
- ParticleAtDay / ParticleAtNight / ParticleAtTime: Fade particle systems in and out according to a time of day or a specific weather type
- LightAtDay / LightAtNight / LightAtTime: Fade light intensities 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
- LoadSkyFromFile: Load exported sky dome parameters at runtime from a TextAsset that can be assigned via drag & drop
- WeatherManager: Example for a weather manager that integrates with Time of Day and manages particle effects

1.16 Frequently Asked Questions

Q: How can I get a sky dome (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 referenes 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 artifacts.

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

- Make sure the "Space" child object is enabled.
- Assign your cubemap to the "Space" material and adjust the brightness parameter.

Q: How can I add the clouds, sun, moon, or stars to reflections?

- Put any game objects you want to show up in reflections on a separate layer.
- Add that layer to the Reflection.CullingMask variable.

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.

Q: How can I adjust the color of the sky around sunrise and sunset?

- Use the Day.SkyColor gradient to adjust the base color tint of the sky at different times of day.
- The left end of the gradient is used around noon.
- The right end of the gradient is used around sunrise and sunset.

Q: My terrain or object lighting looks incorrect when the time of day changes.

- Make sure that static lightmapping is disabled when using a dynamic day / night cycle.

Q: My shadows are flickering when the time of day changes.

- This is a limitation of shadow maps in general, but there are ways to counteract.
- Shadow quality depends on the shadow resolution, shadow distance and shadow cascade quality settings.
- Shadow flickering is caused by light source movement and amplified by huge draw distances or low qualities.
- Increasing Light.UpdateInterval moves the light source less frequently and therefore reduces flickering.

Q: How can I get the god rays (TOD_Rays) to consider or ignore a certain object?

- TOD_Rays uses the depth buffer and color buffer alpha values to block light.
- Objects that should block god rays have to either write to depth or write a valid alpha value.
- Objects that should not block god rays have to use the "ZWrite Off" and "ColorMask RGB" shader features.

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 individually to disable that specific functionality.

Q: I see black lines around objects when using anti-aliasing in forward rendering.

- Disable the "Single Pass" flag on the TOD_Scattering image effect.

Q: My water reflections are missing the sky dome even though its layer is included.

- Disable the "Single Pass" flag on the TOD_Scattering image effect.

NOTE: Always disable entire child objects rather than their individual components since the enabled states of components are being modified by the sky dome scripts, which can override your changes.

1.17 Troubleshooting

If you encounter an issue, please make sure to follow these steps before contacting me:

- Check if the question you are about to ask is in the FAQ section above
- Check that the issue occurs in a new project with just the sky dome to ensure it is related to Time of Day
- Check that both Time of Day and Unity are on the latest version
- Try a full Time of Day reimport (remove the Time of Day folder from the project, restart Unity, reimport Time of Day)
- Try a Unity library rebuild to ensure the issue is reproducible

1.18 Contact Information

To contact me use one of the following methods:

- In the official forum thread of the package
- Via personal message on the Unity community forums
- Via Twitter
- Via my website

NOTE: I should always be able to reply within a couple of work days. If I have not replied after a week, try using a different contact method in case there is an issue with the one you chose.

1.19 Literature

The following literature has been used to implement physically based atmospheric scattering and aerial perspective:

1. Bruneton, Neyret
2. Preetham, Shirley, Smits
3. Hoffman, Preetham
4. Nishita, Sirai, Tadamura, Nakamae
5. Müller, Engel, Döllner

1.20 Changelog

```
VERSION 3.2.0
-------------
- Added support for Unity 5.5 and 5.6
- Added support for single-pass stereo rendering
- Added directional scattering occlusion to TOD_Scattering
- Added gradient fog mode that uses the unmodified gradient color
- Added camera movement parameter to TOD_Animation
- Added single-pass toggle to TOD_Scattering (disable when using AA / planar water reflections)
- Improved forward rendering and anti-aliasing support
- Improved overall visuals in gamma color space
- Improved referencing when instantiating camera and sky at runtime
- Fixed ambient light multiplier being ignored on some Unity versions
- Fixed issues with extreme HDR values
- Fixed several cloud shadow issues
- Fixed issues when rotating the sky dome on the X or Z axis
- Fixed weather manager error when fade time is set to zero
- Fixed various smaller issues and annoyances
- Deprecated clear alpha pass for better compatibility
```

```
VERSION 3.1.0
-------------
- Added star catalog support (data for over 9000 real-life stars is included)
- Revamped cloud density tracing for much more realistic, semi-volumetric clouds
- Revamped cloud shading using a physically based backend
- Replaced cloud shadow projector with the new TOD_Shadows image effect
- Added LDR support to the TOD_Scattering image effect (no longer forces HDR)
- Fixed TOD_Scattering being brighter towards screen edges
- Fixed reflection probe bake ignoring sky dome rotation
- Improved sun spot shape calculation
- Added TOD_Sky.Ambient.Saturation parameter (saturate or desaturate ambient light)
- Added TOD_Sky.ColorOutput selection (toggles anti-banding dithering)
- Added TOD_Sky.Day/Night.FogColor (Atmosphere.Fogginess uses this instead of the cloud color)
- Added TOD_Time.ProgressTime toggle
- Added TOD_Time.OnSunrise and TOD_Time.OnSunset events
- Made TOD_Time.DayLengthInMinutes accept values that are less than one
- Added sky dome setup issue detection to TOD_Sky
- Added more tooltips to the TOD_Sky inspector and made various tooltips more descriptive
- Made day / night gradients span the entire day / night rather than just the transitional periods
- Optimized various recently added features for very old mobile devices like the iPhone 4
- Made shader keywords global (fixes materials sometimes showing up as changed in version control)
- Changed cloud UVs to Vector3 (X and Z are layer coordinates, Y is billboard rotation)
- Always set scene skybox material (not just if reflections or ambient light are enabled)
- Force solid black clear flags since the "Space" child object is now optional
- Moved parameter import and export to TOD_Sky context menu
- Made weather manager an example script instead of a sky dome component
- Added TOD_ prefix to all example scripts
- Removed Clouds.Billboards parameter (billboards can now be placed and tweaked by hand)
- Removed "Space (Cube)" shader (the default space shader now uses a cubemap)

VERSION 3.0.2
-------------
- Added UseDeviceDate to TOD_Time (UseDeviceTime now only sets the time of day, not the date)
- Tweaked sun spot calculation to yield more consistent sizes with increasing mesh brightnesses
- Fixed that ambient colors would be multiplied with their brightnesses twice
- Tweaked some of the default internal night multipliers
- Removed offset of 1000 in all sky dome shaders (causing issues on some mobile devices)
- Changed "RenderType" tag of all sky dome shaders to "Background"
- Adjusted render queue positions of the cloud shaders (minor tweak by -10)
- Replaced TOD_TRANSFORM_VERT macro with direct MVP mul in cloud shadow and skybox shaders

VERSION 3.0.1
-------------

- Removed directional light from reflection probe pass to avoid double specular highlights
- Changed all internal variables to public properties to allow script access when compiled as library
- Forced TOD_Time.DayLengthInMinutes to be greater than or equal to one
- Removed the custom TOD_Camera inspector script
- Adjusted render queue positions of all shaders
- Added global density and height falloff to the TOD_Scattering image effect
- Added depth offset of 1000 to all sky dome shaders
- Added example scripts to adjust audio, light or particle intensities over the course of a day
- Updated some image effect shader syntax to Unity 4 or later
- Replaced the two cloud layer alpha maps with a single RGB texture (one channel per layer)
- Added a third layer to the cloud layer shader
- Replaced the two cloud layer normal maps with a single tiled normal map (same as billboards)
- Tweaked billboard cloud normal map tiling

VERSION 3.0.0
-------------
- Added new atmospheric scattering model (supports planet shadowing)
- Added ColorRange parameter to specify whether or not to output colors in high dynamic range
- Added SkyQuality parameter variable (can be per-vertex and per-pixel)
- Added dynamically batched normal mapped billboard horizon clouds (see Clouds.Billboards)
- Added inspector variable tooltips
- Added events that are fired when a year, month, day, hour or minute have passed to TOD_Time
- Added an image effect that renders atmospheric scattering and aerial perspective in a single pass
- Added profiler samples to TOD_Sky
- Improved inspector variable interface by using property drawers
- Improved inspector variable verification by using property attributes
- Improved cloud layer shading
- Improved shader property update performance
- Improved space rotation by using local sidereal time
- Fixed errors in Unity 5 Beta 21 (this means Beta 20 is no longer supported)
- Changed all textures from PNG to TGA
- Changed all color inspector variables to gradients
- Changed sun shader to a procedural shape instead of a texture
- Removed a number of now unused parameters

VERSION 2.3.5
-------------
- Fixed inaccuracy issues with the time curve approximation
- Fixed possible gimbal lock in the space dome rotation
- Tweaked the default space texture to be more resistant to tiling
- Made all example scripts initialize in Start() instead of OnEnable()
- Made the Space (Cube) shader fade to black in the bottom half of the sky dome
- Made Clouds.Density clamp between 0 and 1

VERSION 2.3.4
-------------
- Fixed moon position being vastly off
- Fixed space texture tiling to infinity towards the horizon (could cause issues when rotating)
- Tweaked horizon line for low haziness values
- Tweaked the default prefab parameters
- Disabled headless mode detection in-editor
- Simplified and optimized TOD_Time calculations
- Changed rendering order of sun and moon to support eclipses
- Made inspector adjustments to the cycle properties correctly progress day, month and year
- Made moon phase get calculated directly from the sun position
- Removed Moon.Phase inspector variable (no longer required)
- Removed Progress* fields from TOD_Time (no longer required)
- Removed Moon (Flat) shader (adjusting Moon.Contrast now has the same effect)

VERSION 2.3.3
-------------

- Added TOD_Sky.LoadParameters(...) to load exported parameters at runtime
- Added LoadSkyFromFile example script
- Added skybox material that is assigned to the render settings skybox for dynamic GI
- Added TOD_Sky.Moon.HaloSize to increase or decrease the size of the moon halo
- Added TOD_Sky.Reflection.ClearFlags to specify which clear flags to use for the reflection cubemap
- Added TOD_Sky.Reflection.CullingMask to specify which layers to include in the reflection cubemap
- Added warning to TOD_Camera if skybox clear flags are used (redundant with a sky dome)
- Made parameter export and import remember the most recently specified path
- Made the reflection cubemap less bright in the bottom hemisphere
- Made light source color fall off to black before switching positions
- Changed reflection baking to use a native Unity 5 realtime reflection probe (better quality)
- Changed TOD_Sky.RenderToSphericalHarmonics(...) and TOD_Sky.RenderToCubemap(...) APIs
- Renamed TOD_Components.*Shader to TOD_Components.*Material
- Removed TOD_Sky.Fog.UpdateInterval (it's fast enough to update every frame anyhow)
- Removed TOD_Sky.Fog/Ambient/Reflection.Directional (now part of fog mode, unused for the others)
- Removed some parameters that are unused on Unity 3 and Unity 4 if running those versions

VERSION 2.3.2
-------------

- Fixed that the sky dome would go into headless mode (i.e. black) on mobile
- Fixed an error in Unity 5 Beta 14 (this means Beta 13 is no longer supported)
- Made sky fogginess correctly affect the light intensity
- Optimized coloring calculations
- Renamed TOD_AmbientType.Flat to TOD_AmbientType.Color
- Renamed TOD_AmbientType.Trilight to TOD_AmbientType.Gradient
- Renamed TOD_Sky.RenderToSH3(...) to TOD_Sky.RenderToSphericalHarmonics(...)

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 paramter "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 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 use
```

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

Chapter 3

Class Index

3.1 Class List

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

Chapter 4

Class Documentation

4.1 TOD_AmbientParameters Class Reference

Parameters of the ambient mode. This is the type of the TOD_Sky.Ambient inspector variable group.

Public Attributes

- TOD_AmbientType Mode = TOD_AmbientType.Color

  Ambient light mode.
- float Saturation = 1.0f

  Saturation of the ambient light.
- float UpdateInterval = 1.0f

  Refresh interval of the ambient light probe in seconds.

4.1.1 Detailed Description

Parameters of the ambient mode. This is the type of the TOD_Sky.Ambient inspector variable group.

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 CameraMovement = 1.0f

  How much to move the clouds when the camera moves.
- 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 = 1.0f

  Speed of the wind that is acting on the clouds.

Properties

- Vector3 CloudUV `[get, set]`

  Current cloud UV coordinates. Can be synchronized between multiple game clients to guarantee identical cloud positions.
- Vector3 OffsetUV `[get]`

  Current offset UV coordinates. 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. This is the type of the TOD_Sky.Atmosphere inspector variable group.

Public Attributes

- float RayleighMultiplier = 1.0f

  [0, ∞] Intensity of the atmospheric Rayleigh scattering.
- float MieMultiplier = 1.0f

  [0, ∞] Intensity of the atmospheric Mie scattering.
- float Brightness = 1.5f

  [0, ∞] Overall brightness of the atmosphere.
- float Contrast = 1.5f

  [0, ∞] Overall contrast of the atmosphere.
- float Directionality = 0.7f

  [0, 1] Directionality factor that determines the size of the glow around the sun.
- float Fogginess = 0.0f

  [0, 1] Density of the fog covering the sky.

4.3.1 Detailed Description

Parameters of the atmosphere. This is the type of the TOD_Sky.Atmosphere inspector variable group.

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

- TOD_Parameters.cs

4.4 TOD_Billboard Class Reference

Public Attributes

- float **Altitude** = 0
- float **Azimuth** = 0
- float **Distance** = 1
- float **Size** = 1

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

- TOD_Billboard.cs

4.5 TOD_Camera Class Reference

Sky dome management camera component.

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.

Properties

- bool **HDR** `[get]`
- float **NearClipPlane** `[get]`
- float **FarClipPlane** `[get]`
- Color **BackgroundColor** `[get]`

4.5.1 Detailed Description

Sky dome management camera component.

Move and scale the sky dome every frame after the rest of the scene has fully updated.

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

- TOD_Camera.cs

4.6 TOD_CloudParameters Class Reference

Parameters of the clouds. This is the type of the TOD_Sky.Clouds inspector variable group.

Public Attributes

- float Size = 2.0f

  [1, ∞] Size of the clouds.
- float Opacity = 1.0f

  [0, 1] Opacity of the clouds.
- float Coverage = 0.5f

  [0, 1] How much sky is covered by clouds.
- float Sharpness = 0.5f

  [0, 1] Sharpness of the cloud to sky transition.
- float Coloring = 0.5f

  [0, 1] Coloring of the clouds.
- float Attenuation = 0.5f

  [0, 1] Amount of skylight that is blocked.
- float Saturation = 0.5f

  [0, 1] Amount of sunlight that is blocked. Only affects the highest cloud quality setting.
- float Scattering = 1.0f

  [0, ∞] Intensity of the cloud translucency glow. Only affects the highest cloud quality setting.
- float Brightness = 1.5f

  [0, ∞] Brightness of the clouds.

4.6.1 Detailed Description

Parameters of the clouds. This is the type of the TOD_Sky.Clouds inspector variable group.

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

- TOD_Parameters.cs

4.7 TOD_Components Class Reference

Component manager class.

Public Member Functions

- void Initialize ()

  Initializes all component references.

Public Attributes

- GameObject Space = null

  Space object.
- GameObject Stars = null

  Star object.
- GameObject Sun = null

  Sun object.
- GameObject Moon = null

  Moon object.
- GameObject Atmosphere = null

  Atmosphere object.
- GameObject Clear = null

  Clear object.
- GameObject Clouds = null

  Cloud object.
- GameObject Billboards = null

  Billboard parent.
- GameObject Light = null

  Light object.

Properties

- Transform DomeTransform `[get, set]`

  Dome transform.
- Transform SpaceTransform `[get, set]`

  Space transform.
- Transform StarTransform `[get, set]`

  Star transform.
- Transform SunTransform `[get, set]`

  Sun transform.
- Transform MoonTransform `[get, set]`

  Moon transform.
- Transform AtmosphereTransform `[get, set]`

  Atmosphere transform.
- Transform ClearTransform `[get, set]`

  Clear transform.
- Transform CloudTransform `[get, set]`

  Cloud transform.
- Transform BillboardTransform `[get, set]`

  Billboard transform.
- Transform LightTransform `[get, set]`

  Light transform.
- Renderer SpaceRenderer `[get, set]`

Space renderer.

- Renderer StarRenderer `[get, set]`

  Star renderer.
- Renderer SunRenderer `[get, set]`

  Sun renderer.
- Renderer MoonRenderer `[get, set]`

  Moon renderer.
- Renderer AtmosphereRenderer `[get, set]`

  Atmosphere renderer.
- Renderer ClearRenderer `[get, set]`

  Clear renderer.
- Renderer CloudRenderer `[get, set]`

  Cloud renderer.
- Renderer [ ] BillboardRenderers `[get, set]`

  Billboard renderers.
- MeshFilter SpaceMeshFilter `[get, set]`

  Space mesh filter.
- MeshFilter StarMeshFilter `[get, set]`

  Star mesh filter.
- MeshFilter SunMeshFilter `[get, set]`

  Sun mesh filter.
- MeshFilter MoonMeshFilter `[get, set]`

  Moon mesh filter.
- MeshFilter AtmosphereMeshFilter `[get, set]`

  Atmosphere mesh filter.
- MeshFilter ClearMeshFilter `[get, set]`

  Clear mesh filter.
- MeshFilter CloudMeshFilter `[get, set]`

  Cloud mesh filter.
- MeshFilter [ ] BillboardMeshFilters `[get, set]`

  Billboard mesh filters.
- Material SpaceMaterial `[get, set]`

  Space material.
- Material StarMaterial `[get, set]`

  Star material.
- Material SunMaterial `[get, set]`

  Sun material.
- Material MoonMaterial `[get, set]`

  Moon material.
- Material AtmosphereMaterial `[get, set]`

  Atmosphere material.
- Material ClearMaterial `[get, set]`

  Clear material.
- Material CloudMaterial `[get, set]`

  Cloud material.
- Material [ ] BillboardMaterials `[get, set]`

  Billboard materials.
- Light LightSource `[get, set]`

  Light source.
- TOD_Sky Sky `[get, set]`

  Dome sky script.

- TOD_Animation Animation `[get, set]`

  Dome animation script.
- TOD_Time Time `[get, set]`

  Dome time script.
- TOD_Camera Camera `[get, set]`

  Camera script.
- TOD_Rays Rays `[get, set]`

  Camera god ray script.
- TOD_Scattering Scattering `[get, set]`

  Camera atmospheric scattering script.
- TOD_Shadows Shadows `[get, set]`

  Camera cloud shadow script.

4.7.1 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

4.8 TOD_CycleParameters Class Reference

Parameters of the day and night cycle. This is the type of the TOD_Sky.Cycle inspector variable group.

Public Attributes

- float Hour = 12

  [0, 24] Current hour of the day.
- int Day = 15

  [1, 31] Current day of the month.
- int Month = 6

  [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.

4.8.1 Detailed Description

Parameters of the day and night cycle. This is the type of the TOD_Sky.Cycle inspector variable group.

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

- TOD_Parameters.cs

4.9 TOD_DayParameters Class Reference

Parameters that are unique to the day. This is the type of the TOD_Sky.Day inspector variable group.

Public Attributes

- Gradient SunColor

  Color of the sun spot. Left value: Sun at zenith. Right value: Sun at horizon.
- Gradient LightColor

  Color of the light that hits the ground. Left value: Sun at zenith. Right value: Sun at horizon.
- Gradient RayColor

  Color of the god rays. Left value: Sun at zenith. Right value: Sun at horizon.
- Gradient SkyColor

  Color of the light that hits the atmosphere. Left value: Sun at zenith. Right value: Sun at horizon.
- Gradient CloudColor

  Color of the clouds. Left value: Sun at zenith. Right value: Sun at horizon.
- Gradient FogColor

  Color of the atmosphere fog. Left value: Sun at zenith. Right value: Sun at horizon.
- Gradient AmbientColor

  Color of the ambient light. Left value: Sun at zenith. Right value: Sun at horizon.
- float LightIntensity = 1.0f

  [0, 8] Intensity of the light source.
- float ShadowStrength = 1.0f

  [0, 1] Opacity of the shadows dropped by the light source.
- float AmbientMultiplier = 1.0f

  [0, 1] Brightness multiplier of the ambient light.
- float ReflectionMultiplier = 1.0f

  [0, 1] Brightness multiplier of the reflection probe.

4.9.1 Detailed Description

Parameters that are unique to the day. This is the type of the TOD_Sky.Day inspector variable group.

4.9.2 Member Data Documentation

4.9.2.1 AmbientColor

```
Gradient TOD_DayParameters.AmbientColor
```

Initial value:

```
= new Gradient()
    {
        alphaKeys = new GradientAlphaKey[] {
            new GradientAlphaKey(1.0f, 0.0f),
            new GradientAlphaKey(1.0f, 0.5f),
            new GradientAlphaKey(1.0f, 1.0f)
        },
        colorKeys = new GradientColorKey[] {
            new GradientColorKey(new Color32(094, 089, 087, 255), 0.0f),
            new GradientColorKey(new Color32(094, 089, 087, 255), 0.5f),
            new GradientColorKey(new Color32(094, 089, 087, 255), 1.0f)
        }
    }
```

Color of the ambient light. Left value: Sun at zenith. Right value: Sun at horizon.

4.9.2.2 CloudColor

```
Gradient TOD_DayParameters.CloudColor
```

Initial value:

```
= new Gradient()
    {
        alphaKeys = new GradientAlphaKey[] {
            new GradientAlphaKey(1.0f, 0.0f),
            new GradientAlphaKey(1.0f, 0.5f),
            new GradientAlphaKey(1.0f, 1.0f)
        },
        colorKeys = new GradientColorKey[] {
            new GradientColorKey(new Color32(255, 255, 255, 255), 0.0f),
            new GradientColorKey(new Color32(255, 255, 255, 255), 0.5f),
            new GradientColorKey(new Color32(255, 195, 145, 255), 1.0f)
        }
    }
```

Color of the clouds. Left value: Sun at zenith. Right value: Sun at horizon.

4.9.2.3 FogColor

```
Gradient TOD_DayParameters.FogColor
```

Initial value:

```
= new Gradient()
    {
        alphaKeys = new GradientAlphaKey[] {
            new GradientAlphaKey(1.0f, 0.0f),
            new GradientAlphaKey(1.0f, 0.5f),
            new GradientAlphaKey(1.0f, 1.0f)
        },
        colorKeys = new GradientColorKey[] {
            new GradientColorKey(new Color32(191, 191, 191, 255), 0.0f),
            new GradientColorKey(new Color32(191, 191, 191, 255), 0.5f),
            new GradientColorKey(new Color32(127, 127, 127, 255), 1.0f)
        }
    }
```

Color of the atmosphere fog. Left value: Sun at zenith. Right value: Sun at horizon.

4.9.2.4 LightColor

```
Gradient TOD_DayParameters.LightColor
```

Initial value:

```
= new Gradient()
    {
        alphaKeys = new GradientAlphaKey[] {
            new GradientAlphaKey(1.0f, 0.0f),
            new GradientAlphaKey(1.0f, 0.5f),
            new GradientAlphaKey(1.0f, 1.0f)
        },
        colorKeys = new GradientColorKey[] {
            new GradientColorKey(new Color32(255, 243, 234, 255), 0.0f),
            new GradientColorKey(new Color32(255, 243, 234, 255), 0.5f),
            new GradientColorKey(new Color32(255, 154, 000, 255), 1.0f)
        }
    }
```

Color of the light that hits the ground. Left value: Sun at zenith. Right value: Sun at horizon.

4.9.2.5 RayColor

```
Gradient TOD_DayParameters.RayColor
```

Initial value:

```
= new Gradient()
    {
        alphaKeys = new GradientAlphaKey[] {
            new GradientAlphaKey(1.0f, 0.0f),
            new GradientAlphaKey(1.0f, 0.5f),
            new GradientAlphaKey(1.0f, 1.0f)
        },
        colorKeys = new GradientColorKey[] {
            new GradientColorKey(new Color32(255, 243, 234, 255), 0.0f),
            new GradientColorKey(new Color32(255, 243, 234, 255), 0.5f),
            new GradientColorKey(new Color32(255, 154, 000, 255), 1.0f)
        }
    }
```

Color of the god rays. Left value: Sun at zenith. Right value: Sun at horizon.

4.9.2.6 SkyColor

```
Gradient TOD_DayParameters.SkyColor
```

Initial value:

```
= new Gradient()
    {
        alphaKeys = new GradientAlphaKey[] {
            new GradientAlphaKey(1.0f, 0.0f),
            new GradientAlphaKey(1.0f, 0.5f),
            new GradientAlphaKey(1.0f, 1.0f)
        },
        colorKeys = new GradientColorKey[] {
            new GradientColorKey(new Color32(255, 243, 234, 255), 0.0f),
            new GradientColorKey(new Color32(255, 243, 234, 255), 0.5f),
            new GradientColorKey(new Color32(255, 243, 234, 255), 1.0f)
        }
    }
```

Color of the light that hits the atmosphere. Left value: Sun at zenith. Right value: Sun at horizon.

4.9.2.7 SunColor

`Gradient TOD_DayParameters.SunColor`

Initial value:

```
= new Gradient()
    {
        alphaKeys = new GradientAlphaKey[] {
            new GradientAlphaKey(1.0f, 0.0f),
            new GradientAlphaKey(1.0f, 0.5f),
            new GradientAlphaKey(1.0f, 1.0f)
        },
        colorKeys = new GradientColorKey[] {
            new GradientColorKey(new Color32(253, 171, 050, 255), 0.0f),
            new GradientColorKey(new Color32(253, 171, 050, 255), 0.5f),
            new GradientColorKey(new Color32(253, 171, 050, 255), 1.0f)
        }
    }
```

Color of the sun spot. Left value: Sun at zenith. Right value: Sun at horizon.

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

- TOD_Parameters.cs

4.10 TOD_FogParameters Class Reference

Parameters of the fog mode. This is the type of the TOD_Sky.Fog inspector variable group.

Public Attributes

- TOD_FogType Mode = TOD_FogType.Atmosphere

  Fog color mode.
- float HeightBias = 0.0f

  [0, 1] Fog color sampling height.
  = 0 fog is atmosphere color at horizon.
  = 1 fog is atmosphere color at zenith.

4.10.1 Detailed Description

Parameters of the fog mode. This is the type of the TOD_Sky.Fog inspector variable group.

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

- TOD_Parameters.cs

4.11 TOD_ImageEffect Class Reference

Image effect base class.

Public Types

- enum ResolutionType { **Low**, **Normal**, **High** }

  Image effect resolutions.

Public Attributes

- TOD_Sky sky = null

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

4.11.1 Detailed Description

Image effect base class.

Based on PostEffectsBase from the Unity Standard Assets. Extended for image effects that depend on a TOD_Sky reference.

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

- TOD_ImageEffect.cs

4.12 TOD_LightParameters Class Reference

Parameters of the light source. This is the type of the TOD_Sky.Light inspector variable group.

Public Attributes

- float UpdateInterval = 0.0f

  [0, ∞] Refresh interval of the light source position in seconds.
- float MinimumHeight = 0.0f

  [-1, 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.

4.12.1 Detailed Description

Parameters of the light source. This is the type of the TOD_Sky.Light inspector variable group.

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

- TOD_Parameters.cs

4.13 TOD_MaxAttribute Class Reference

Public Member Functions

- **TOD_MaxAttribute** (float max)

Public Attributes

- float **max**

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

- TOD_Attributes.cs

4.14 TOD_MinAttribute Class Reference

Public Member Functions

- **TOD_MinAttribute** (float min)

Public Attributes

- float **min**

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

- TOD_Attributes.cs

4.15 TOD_MoonParameters Class Reference

Parameters that are unique to the moon. This is the type of the TOD_Sky.Moon inspector variable group.

Public Attributes

- float MeshSize = 1.0f

  [0, ∞] Diameter of the moon in degrees. The diameter as seen from earth is 0.5 degrees.
- float MeshBrightness = 2.0f

  [0, ∞] Brightness of the moon.
- float MeshContrast = 1.0f

  [0, ∞] Contrast of the moon.
- float HaloSize = 0.1f

  [0, ∞] Size of the moon halo.
- float HaloBrightness = 1.0f

  [0, ∞] Brightness of the moon halo.
- TOD_MoonPositionType Position = TOD_MoonPositionType.Realistic

  Type of the moon position calculation.

4.15.1 Detailed Description

Parameters that are unique to the moon. This is the type of the TOD_Sky.Moon inspector variable group.

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

- TOD_Parameters.cs

4.16 TOD_NightParameters Class Reference

Parameters that are unique to the night. This is the type of the TOD_Sky.Night inspector variable group.

Public Attributes

- Gradient MoonColor

  Color of the moon mesh. Left value: Sun at horizon. Right value: Sun opposite to zenith.
- Gradient LightColor

  Color of the light that hits the ground. Left value: Sun at horizon. Right value: Sun opposite to zenith.
- Gradient RayColor

  Color of the god rays. Left value: Sun at horizon. Right value: Sun opposite to zenith.
- Gradient SkyColor

  Color of the light that hits the atmosphere. Left value: Sun at horizon. Right value: Sun opposite to zenith.
- Gradient CloudColor

  Color of the clouds. Left value: Sun at horizon. Right value: Sun opposite to zenith.
- Gradient FogColor

  Color of the atmosphere fog. Left value: Sun at horizon. Right value: Sun opposite to zenith.
- Gradient AmbientColor

  Color of the ambient light. Left value: Sun at horizon. Right value: Sun opposite to zenith.
- float LightIntensity = 0.1f

  [0, 8] Intensity of the light source.
- float ShadowStrength = 1.0f

  [0, 1] Opacity of the shadows dropped by the light source.
- float AmbientMultiplier = 1.0f

  [0, 1] Brightness multiplier of the ambient light.
- float ReflectionMultiplier = 1.0f

  [0, 1] Brightness multiplier of the reflection probe.

4.16.1 Detailed Description

Parameters that are unique to the night. This is the type of the TOD_Sky.Night inspector variable group.

4.16.2 Member Data Documentation

4.16.2.1 AmbientColor

```
Gradient TOD_NightParameters.AmbientColor
```

Initial value:

```
= new Gradient()
    {
        alphaKeys = new GradientAlphaKey[] {
            new GradientAlphaKey(1.0f, 0.0f),
            new GradientAlphaKey(0.2f, 0.5f),
            new GradientAlphaKey(0.2f, 1.0f)
        },
        colorKeys = new GradientColorKey[] {
            new GradientColorKey(new Color32(025, 040, 065, 255), 0.0f),
            new GradientColorKey(new Color32(025, 040, 065, 255), 0.5f),
            new GradientColorKey(new Color32(025, 040, 065, 255), 1.0f)
        }
    }
```

Color of the ambient light. Left value: Sun at horizon. Right value: Sun opposite to zenith.

4.16.2.2 CloudColor

```
Gradient TOD_NightParameters.CloudColor
```

Initial value:

```
= new Gradient()
    {
        alphaKeys = new GradientAlphaKey[] {
            new GradientAlphaKey(1.0f, 0.0f),
            new GradientAlphaKey(0.1f, 0.5f),
            new GradientAlphaKey(0.1f, 1.0f)
            },
        colorKeys = new GradientColorKey[] {
            new GradientColorKey(new Color32(025, 040, 065, 255), 0.0f),
            new GradientColorKey(new Color32(025, 040, 065, 255), 0.5f),
            new GradientColorKey(new Color32(025, 040, 065, 255), 1.0f)
        }
    }
```

Color of the clouds. Left value: Sun at horizon. Right value: Sun opposite to zenith.

4.16.2.3 FogColor

```
Gradient TOD_NightParameters.FogColor
```

Initial value:

```
= new Gradient()
    {
        alphaKeys = new GradientAlphaKey[] {
            new GradientAlphaKey(1.0f, 0.0f),
            new GradientAlphaKey(0.2f, 0.5f),
            new GradientAlphaKey(0.2f, 1.0f)
        },
        colorKeys = new GradientColorKey[] {
            new GradientColorKey(new Color32(025, 040, 065, 255), 0.0f),
            new GradientColorKey(new Color32(025, 040, 065, 255), 0.5f),
            new GradientColorKey(new Color32(025, 040, 065, 255), 1.0f)
        }
    }
```

Color of the atmosphere fog. Left value: Sun at horizon. Right value: Sun opposite to zenith.

4.16.2.4 LightColor

```
Gradient TOD_NightParameters.LightColor
```

Initial value:

```
= new Gradient()
    {
        alphaKeys = new GradientAlphaKey[] {
            new GradientAlphaKey(1.0f, 0.0f),
            new GradientAlphaKey(1.0f, 0.5f),
            new GradientAlphaKey(1.0f, 1.0f)
        },
        colorKeys = new GradientColorKey[] {
            new GradientColorKey(new Color32(025, 040, 065, 255), 0.0f),
            new GradientColorKey(new Color32(025, 040, 065, 255), 0.5f),
            new GradientColorKey(new Color32(025, 040, 065, 255), 1.0f)
        }
    }
```

Color of the light that hits the ground. Left value: Sun at horizon. Right value: Sun opposite to zenith.

4.16.2.5 MoonColor

```
Gradient TOD_NightParameters.MoonColor
```

Initial value:

```
= new Gradient()
    {
        alphaKeys = new GradientAlphaKey[] {
            new GradientAlphaKey(1.0f, 0.0f),
            new GradientAlphaKey(1.0f, 0.5f),
            new GradientAlphaKey(1.0f, 1.0f)
        },
        colorKeys = new GradientColorKey[] {
            new GradientColorKey(new Color32(255, 233, 200, 255), 0.0f),
            new GradientColorKey(new Color32(255, 233, 200, 255), 0.5f),
            new GradientColorKey(new Color32(255, 233, 200, 255), 1.0f)
        }
    }
```

Color of the moon mesh. Left value: Sun at horizon. Right value: Sun opposite to zenith.

4.16.2.6 RayColor

```
Gradient TOD_NightParameters.RayColor
```

Initial value:

```
= new Gradient()
    {
        alphaKeys = new GradientAlphaKey[] {
            new GradientAlphaKey(1.0f, 0.0f),
            new GradientAlphaKey(0.2f, 0.5f),
            new GradientAlphaKey(0.2f, 1.0f)
        },
        colorKeys = new GradientColorKey[] {
            new GradientColorKey(new Color32(025, 040, 065, 255), 0.0f),
            new GradientColorKey(new Color32(025, 040, 065, 255), 0.5f),
            new GradientColorKey(new Color32(025, 040, 065, 255), 1.0f)
        }
    }
```

Color of the god rays. Left value: Sun at horizon. Right value: Sun opposite to zenith.

4.16.2.7 SkyColor

```
Gradient TOD_NightParameters.SkyColor
```

Initial value:

```
= new Gradient()
    {
        alphaKeys = new GradientAlphaKey[] {
            new GradientAlphaKey(1.0f, 0.0f),
            new GradientAlphaKey(0.2f, 0.5f),
            new GradientAlphaKey(0.2f, 1.0f)
        },
        colorKeys = new GradientColorKey[] {
            new GradientColorKey(new Color32(025, 040, 065, 255), 0.0f),
            new GradientColorKey(new Color32(025, 040, 065, 255), 0.5f),
            new GradientColorKey(new Color32(025, 040, 065, 255), 1.0f)
        }
    }
```

Color of the light that hits the atmosphere. Left value: Sun at horizon. Right value: Sun opposite to zenith.

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

- TOD_Parameters.cs

4.17 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**

4.17.1 Detailed Description

All parameters of the sky dome.

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

- TOD_Parameters.cs

4.18 TOD_RangeAttribute Class Reference

Public Member Functions

- **TOD_RangeAttribute** (float min, float max)

Public Attributes

- float **min**
- float **max**

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

- TOD_Attributes.cs

4.19 TOD_Rays Class Reference

God ray camera component.

Public Types

- enum BlendModeType { **Screen**, **Add** }

  Methods to blend the god rays with the image.

Public Attributes

- Shader **GodRayShader** = null
- Shader **ScreenClearShader** = null
- Shader **SkyMaskShader** = null
- bool UseDepthTexture = true

  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.
- BlendModeType BlendMode = BlendModeType.Screen

  The god ray rendering blend mode.
- float Intensity = 1

  The intensity of the god rays.
- ResolutionType Resolution = ResolutionType.Normal

  The god ray rendering resolution.
- int BlurIterations = 2

  The number of blur iterations to be performed.
- float BlurRadius = 2

  The radius to blur filter applied to the god rays.
- float MaxRadius = 0.5f

  The maximum radius of the god rays.

4.19.1 Detailed Description

God ray camera component.

Based on SunShafts from the Unity Standard Assets. 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

4.20 TOD_ReflectionParameters Class Reference

Parameters of the reflection mode. This is the type of the TOD_Sky.Reflection inspector variable group.

Public Attributes

- TOD_ReflectionType Mode = TOD_ReflectionType.None

  Reflection probe mode.
- ReflectionProbeClearFlags ClearFlags = ReflectionProbeClearFlags.Skybox

  Clear flags to use for the reflection.
- LayerMask CullingMask = 0

  Layers to include in the reflection.
- ReflectionProbeTimeSlicingMode TimeSlicing = ReflectionProbeTimeSlicingMode.AllFacesAtOnce

  Time slicing behaviour to spread out rendering cost over multiple frames.
- int Resolution = 128

  Resolution of the reflection bake.
- float UpdateInterval = 1.0f

  Refresh interval of the reflection cubemap in seconds.

4.20.1 Detailed Description

Parameters of the reflection mode. This is the type of the TOD_Sky.Reflection inspector variable group.

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

- TOD_Parameters.cs

4.21 TOD_Resources Class Reference

Material and mesh wrapper class.

Public Member Functions

- void Initialize ()

  Initializes all resource references.

Public Attributes

- Material **Skybox**
- Mesh **MoonLOD0**
- Mesh **MoonLOD1**
- Mesh **MoonLOD2**
- Mesh **SkyLOD0**
- Mesh **SkyLOD1**
- Mesh **SkyLOD2**
- Mesh **CloudsLOD0**
- Mesh **CloudsLOD1**
- Mesh **CloudsLOD2**
- Mesh **StarsLOD0**
- Mesh **StarsLOD1**
- Mesh **StarsLOD2**

Properties

- int **ID_SunLightColor** `[get]`
- int **ID_MoonLightColor** `[get]`
- int **ID_SunSkyColor** `[get]`
- int **ID_MoonSkyColor** `[get]`
- int **ID_SunMeshColor** `[get]`
- int **ID_MoonMeshColor** `[get]`
- int **ID_SunCloudColor** `[get]`
- int **ID_MoonCloudColor** `[get]`
- int **ID_FogColor** `[get]`
- int **ID_GroundColor** `[get]`
- int **ID_AmbientColor** `[get]`
- int **ID_SunDirection** `[get]`
- int **ID_MoonDirection** `[get]`
- int **ID_LightDirection** `[get]`
- int **ID_LocalSunDirection** `[get]`
- int **ID_LocalMoonDirection** `[get]`
- int **ID_LocalLightDirection** `[get]`
- int **ID_Contrast** `[get]`
- int **ID_Brightness** `[get]`
- int **ID_Fogginess** `[get]`
- int **ID_Directionality** `[get]`
- int **ID_MoonHaloPower** `[get]`
- int **ID_MoonHaloColor** `[get]`
- int **ID_CloudSize** `[get]`
- int **ID_CloudOpacity** `[get]`
- int **ID_CloudCoverage** `[get]`
- int **ID_CloudSharpness** `[get]`
- int **ID_CloudDensity** `[get]`
- int **ID_CloudColoring** `[get]`
- int **ID_CloudAttenuation** `[get]`
- int **ID_CloudSaturation** `[get]`
- int **ID_CloudScattering** `[get]`
- int **ID_CloudBrightness** `[get]`
- int **ID_CloudMultiplier** `[get]`
- int **ID_CloudOffset** `[get]`
- int **ID_CloudWind** `[get]`

- int **ID_StarSize** `[get]`
- int **ID_StarBrightness** `[get]`
- int **ID_StarVisibility** `[get]`
- int **ID_SunMeshContrast** `[get]`
- int **ID_SunMeshBrightness** `[get]`
- int **ID_MoonMeshContrast** `[get]`
- int **ID_MoonMeshBrightness** `[get]`
- int **ID_kBetaMie** `[get]`
- int **ID_kSun** `[get]`
- int **ID_k4PI** `[get]`
- int **ID_kRadius** `[get]`
- int **ID_kScale** `[get]`
- int **ID_World2Sky** `[get]`
- int **ID_Sky2World** `[get]`

4.21.1 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

4.22 TOD_Scattering Class Reference

Atmospheric scattering and aerial perspective camera component.

Public Attributes

- Shader **ScatteringShader** = null
- Shader **ScreenClearShader** = null
- Shader **SkyMaskShader** = null
- Texture2D **DitheringTexture** = null
- bool **SinglePass** = true
- float GlobalDensity = 0.01f

  How quickly the fog thickens with increasing distance.
- float HeightFalloff = 0.01f

  How quickly the fog falls off with increasing altitude.
- float ZeroLevel = 0.0f

  The height where fog reaches its maximum density.
- ResolutionType Resolution = ResolutionType.Normal

  The scattering resolution.
- int BlurIterations = 2

  The number of blur iterations to be performed.
- float BlurRadius = 2

  The radius to blur filter applied to the directional scattering.
- float MaxRadius = 1

  The maximum radius of the directional scattering.

Additional Inherited Members

4.22.1 Detailed Description

Atmospheric scattering and aerial perspective camera component.

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

- TOD_Scattering.cs

4.23 TOD_Shadows Class Reference

Cloud shadow camera component.

Public Attributes

- Shader **ShadowShader** = null
- Texture2D **CloudTexture** = null
- float **Cutoff** = 0.0f
- float **Fade** = 0.0f
- float **Intensity** = 0.5f

Additional Inherited Members

4.23.1 Detailed Description

Cloud shadow camera component.

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

- TOD_Shadows.cs

4.24 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.
- SphericalHarmonicsL2 RenderToSphericalHarmonics ()

  Render the sky dome to 3rd order spherical harmonics.
- SphericalHarmonicsL2 RenderToSphericalHarmonics (float intensity, float saturation)

  Render the sky dome to 3rd order spherical harmonics.
- void RenderToCubemap (RenderTexture targetTexture=null)

  Render the sky dome to a cubemap render texture.
- Color SampleFogColor (bool directLight=true)

  Calculate the fog color.
- Color SampleSkyColor ()

  Calculate the sky color.
- Color SampleEquatorColor ()

  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.
- void LoadParameters (string xml)

  Load parameters at runtime.

Public Attributes

- TOD_ColorSpaceType ColorSpace = TOD_ColorSpaceType.Auto

  Auto: Use the player settings. Linear: Force linear color space. Gamma: Force gamma color space.
- TOD_ColorRangeType ColorRange = TOD_ColorRangeType.Auto

  Auto: Use the camera settings. HDR: Force high dynamic range. LDR: Force low dynamic range.
- TOD_ColorOutputType ColorOutput = TOD_ColorOutputType.Dithered

  Raw: Write color without modifications. Dithered: Add dithering to reduce banding.
- TOD_SkyQualityType SkyQuality = TOD_SkyQualityType.PerVertex

  Per Vertex: Calculate sky color per vertex. Per Pixel: Calculate sky color per pixel.
- TOD_CloudQualityType CloudQuality = TOD_CloudQualityType.High

  Low: Only recommended for very old mobile devices. Medium: Simplified cloud shading. High: Physically based cloud shading.
- TOD_MeshQualityType MeshQuality = TOD_MeshQualityType.High

  Low: Only recommended for very old mobile devices. Medium: Simplified mesh geometry. High: Detailed mesh geometry.
- TOD_StarQualityType StarQuality = TOD_StarQualityType.High

  Low: Recommended for most mobile devices. Medium: Includes most visible stars. High: Includes all visible stars.
- TOD_CycleParameters Cycle

  Parameters of the day and night cycle.
- TOD_WorldParameters World

  Parameters of the world.

- TOD_AtmosphereParameters Atmosphere

  Parameters of the atmosphere.
- TOD_DayParameters Day

  Parameters of the day.
- TOD_NightParameters Night

  Parameters of the night.
- TOD_SunParameters Sun

  Parameters of the sun.
- TOD_MoonParameters Moon

  Parameters of the moon.
- TOD_StarParameters Stars

  Parameters of the stars.
- TOD_CloudParameters Clouds

  Parameters of the cloud layers.
- TOD_LightParameters Light

  Parameters of the light source.
- TOD_FogParameters Fog

  Parameters of the fog.
- TOD_AmbientParameters Ambient

  Parameters of the ambient light.
- TOD_ReflectionParameters Reflection

  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.
- bool Headless `[get]`

  Whether or not the sky dome is running in headless mode.
- TOD_Components Components `[get]`

  Containins references to all components.
- TOD_Resources Resources `[get]`

  Containins references to all resources.
- 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 Diameter `[get]`

  Diameter of the sky dome.
- 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.
= 90 if the sun is exactly at the horizon.
= 180 if the sun is exactly opposize to zenith.

- float SunAltitude `[get]`

  Sun altitude angle in degrees.
  = -90 if the sun is exactly opposite to zenith.
  = 0 if the sun is exactly at the horizon.
  = 90 if the sun is exactly at zenith.

- float SunAzimuth `[get]`

  Sun azimuth angle in degrees.
  = 0 if the sun is exactly in the north.
  = 90 if the sun is exactly in the east.
  = 180 if the sun is exactly in the south.
  = 270 if the sun is exactly in the west.

- 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 MoonAltitude `[get]`

  Moon altitude angle in degrees.
  = -90 if the moon is exactly opposite to zenith.
  = 0 if the moon is exactly at the horizon.
  = 90 if the moon is exactly at zenith.

- float MoonAzimuth `[get]`

  Moon azimuth angle in degrees.
  = 0 if the moon is exactly in the north.
  = 90 if the moon is exactly in the east.
  = 180 if the moon is exactly in the south.
  = 270 if the moon is exactly in the west.

- float SunsetTime `[get]`

  Time the sun sets.

- float SunriseTime `[get]`

  Time the sun rises.

- float LocalSiderealTime `[get]`

  The current local sidereal time.

- 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.

- float SunVisibility `[get]`

  Current sun visibility.

- float MoonVisibility `[get]`

  Current moon visibility.

- Vector3 SunDirection `[get]`

  Sun direction vector in world space.

- Vector3 MoonDirection `[get]`

  Moon 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 LocalSunDirection `[get]`

  Sun direction vector in sky dome object space.

- Vector3 LocalMoonDirection `[get]`

  Moon 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 SunLightColor `[get]`

  Current sun light color.
- Color MoonLightColor `[get]`

  Current moon light color.
- Color LightColor `[get]`

  Current light color. The color of TOD_Sky.Components.LightSource.
- Color SunRayColor `[get]`

  Current sun ray color.
- Color MoonRayColor `[get]`

  Current moon ray color.
- Color SunSkyColor `[get]`

  Current sun sky color.
- Color MoonSkyColor `[get]`

  Current moon sky color.
- Color SunMeshColor `[get]`

  Current sun mesh color.
- Color MoonMeshColor `[get]`

  Current moon mesh color.
- Color SunCloudColor `[get]`

  Current sun cloud color.
- Color MoonCloudColor `[get]`

  Current moon cloud color.
- Color FogColor `[get]`

  Current fog color.
- Color GroundColor `[get]`

  Current ground color.
- Color AmbientColor `[get]`

  Current ambient light color.
- Color MoonHaloColor `[get]`

  Current moon halo color.
- ReflectionProbe Probe `[get]`

  Current reflection probe.

4.24.1 Detailed Description

Main sky dome management class.

Component of the sky dome parent game object.

4.24.2 Member Function Documentation

4.24.2.1 LoadParameters()

```
void TOD_Sky.LoadParameters (
            string xml )  [inline]
```

Load parameters at runtime.

Parameters

xml	The parameters to load, serialized to XML.

4.24.2.2 OrbitalToLocal()

```
Vector3 TOD_Sky.OrbitalToLocal (
            float theta,
            float phi )  [inline]
```

Convert spherical coordinates to cartesian coordinates.

Parameters

theta	Spherical coordinates theta.
phi	Spherical coordinates phi.

Returns

Unity position in local space.

4.24.2.3 OrbitalToUnity()

```
Vector3 TOD_Sky.OrbitalToUnity (
            float radius,
            float theta,
            float phi )  [inline]
```

Convert spherical coordinates to cartesian coordinates.

Parameters

radius	Spherical coordinates radius.
theta	Spherical coordinates theta.
phi	Spherical coordinates phi.

Returns

Unity position in world space.

4.24.2.4 RenderToCubemap()

```
void TOD_Sky.RenderToCubemap (
            RenderTexture targetTexture = null )  [inline]
```

Render the sky dome to a cubemap render texture.

Parameters

targetTexture	Target RenderTexture in which rendering should be done.

4.24.2.5 SampleAtmosphere()

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

Sample atmosphere colors from the sky dome.

Parameters

direction	View direction in world space.
directLight	Whether or not to include direct light.

Returns

Color of the atmosphere in the specified direction.

4.24.2.6 SampleFogColor()

```
Color TOD_Sky.SampleFogColor (
            bool directLight = true )  [inline]
```

Calculate the fog color.

Parameters

directLight	Whether or not to include direct light.

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

- TOD_Sky+API.cs
- TOD_Sky+Settings.cs
- TOD_Sky+Shader.cs

- TOD_Sky.cs
- TOD_Sky+Unity.cs

4.25 TOD_StarParameters Class Reference

Parameters of the stars. This is the type of the TOD_Sky.Stars inspector variable group.

Public Attributes

- float Size = 1.0f

  [0, ∞] Size of the stars.
- float Brightness = 1.0f

  [0, ∞] Brightness of the stars.
- TOD_StarsPositionType Position = TOD_StarsPositionType.Rotating

  Type of the stars position calculation.

4.25.1 Detailed Description

Parameters of the stars. This is the type of the TOD_Sky.Stars inspector variable group.

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

- TOD_Parameters.cs

4.26 TOD_SunParameters Class Reference

Parameters that are unique to the sun. This is the type of the TOD_Sky.Sun inspector variable group.

Public Attributes

- float MeshSize = 1.0f

  [0, ∞] Diameter of the sun in degrees. The diameter as seen from earth is 0.5 degrees.
- float MeshBrightness = 2.0f

  [0, ∞] Brightness of the sun.
- float MeshContrast = 1.0f

  [0, ∞] Contrast of the sun.

4.26.1 Detailed Description

Parameters that are unique to the sun. This is the type of the TOD_Sky.Sun inspector variable group.

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

- TOD_Parameters.cs

4.27 TOD_Time Class Reference

Time iteration class.

Public Member Functions

- void RefreshTimeCurve ()

  Apply changes made to TimeCurve.
- float ApplyTimeCurve (float deltaTime)

  Apply the time curve to a time span.
- void AddHours (float hours, bool adjust=true)

  Add hours and fractions of hours to the current time.
- void AddSeconds (float seconds, bool adjust=true)

  Add seconds and fractions of seconds to the current time.

Public Attributes

- float DayLengthInMinutes = 30

  Length of one day in minutes.
- bool ProgressTime = true

  Progress time at runtime.
- bool UseDeviceDate = false

  Set the date to the current device date on start.
- bool UseDeviceTime = false

  Set the time to the current device time on start.
- bool UseTimeCurve = false

  Apply the time curve when progressing time.
- AnimationCurve TimeCurve = AnimationCurve.Linear(0, 0, 24, 24)

  Time progression curve.

Events

- Action OnSecond

  Fired whenever the second value is incremented.
- Action OnMinute

  Fired whenever the minute value is incremented.
- Action OnHour

  Fired whenever the hour value is incremented.
- Action OnDay

  Fired whenever the day value is incremented.
- Action OnMonth

  Fired whenever the month value is incremented.
- Action OnYear

  Fired whenever the year value is incremented.
- Action OnSunrise

  Fired whenever the sun rises.
- Action OnSunset

  Fired whenever the sun sets.

4.27.1 Detailed Description

Time iteration class.

Component of the sky dome parent game object.

4.27.2 Member Function Documentation

4.27.2.1 AddHours()

```
void TOD_Time.AddHours (
             float hours,
             bool adjust = true )  [inline]
```

Add hours and fractions of hours to the current time.

Parameters

hours	The hours to add.
adjust	Whether or not to apply the time curve.

4.27.2.2 AddSeconds()

```
void TOD_Time.AddSeconds (
             float seconds,
             bool adjust = true )  [inline]
```

Add seconds and fractions of seconds to the current time.

Parameters

seconds	The seconds to add.
adjust	Whether or not to apply the time curve.

4.27.2.3 ApplyTimeCurve()

```
float TOD_Time.ApplyTimeCurve (
             float deltaTime )  [inline]
```

Apply the time curve to a time span.

Parameters

deltaTime	The time span to adjust.

Returns

The adjusted time span.

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

- TOD_Time.cs

4.28 TOD_WorldParameters Class Reference

Parameters of the world. This is the type of the TOD_Sky.World inspector variable group.

Public Attributes

- float Latitude = 0

  [-90, +90] Latitude of the current location in degrees.
- float Longitude = 0

  [-180, +180] Longitude of the current location in degrees.
- float UTC = 0

  [-14, +14] UTC/GMT time zone of the current location in hours.

4.28.1 Detailed Description

Parameters of the world. This is the type of the TOD_Sky.World inspector variable group.

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

- TOD_Parameters.cs

Index