Azure[Sky] Dynamic Skybox v7.0.2

YouTube video: [Azure[Sky] Dynamic Skybox v7.0.0 - Getting Started #1]

Getting Started Using Legacy Render Pipeline

- Drag Azure prefab to scene.
- Delete any pre-existing Directional Light in the scene.
- Add the Fog Scattering effect to the Camera. Component>Azure[Sky]>Azure Fog Scattering.
- Disable the Anti-Aliasing on the project 'Quality Settings'.

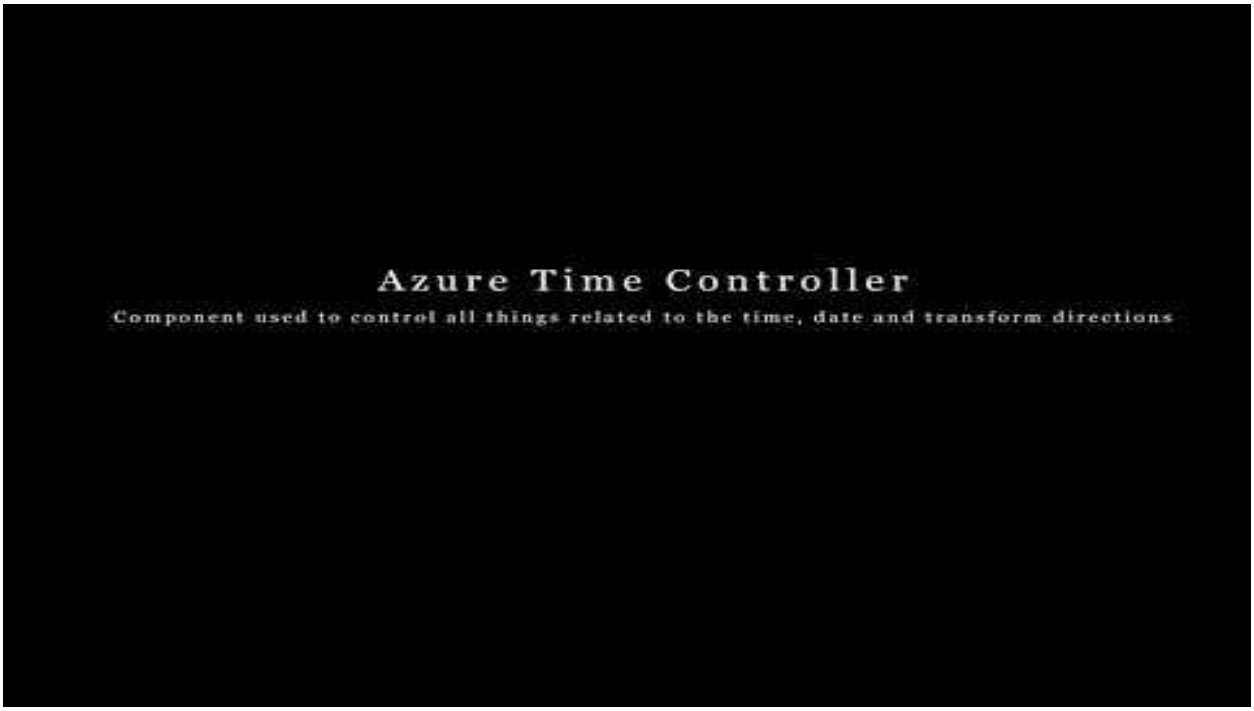
Getting Started Using Universal Render Pipeline

- Import the 'Replacements.unitypackage' from the folder 'UniversalRenderPipeline'.
- Drag Azure prefab to scene.
- Delete any pre-existing Directional Light in the scene.
- Create a new 'ForwardRender.asset' and add the 'Azure Fog Scattering Feature' to it.
- Add the 'ForwardRender.asset' to the 'Render List' of the URP.asset in use by the project.
- Set the camera 'Depth Texture' to 'On'.
- Set the camera 'Renderer' to use the 'ForwardRender.asset' with the fog feature.
- Select the URP.asset and disable the Anti-Aliasing.

Azure Time Controller

Component used to control all things related to the time, date and transform direction.

YouTube video: [Azure Time Controller Overview #2]



Azure Weather Controller

This component controls everything related to the weathers transition, global weather, local weather and override properties.

YouTube video: [Azure Weather Controller Overview #3]

Azure Sky Render Controller

This component controls the rendering of the Skybox and Fog Scattering effect.

YouTube video: [Azure Sky Render Controller Overview #4]

Azure Environment Controller

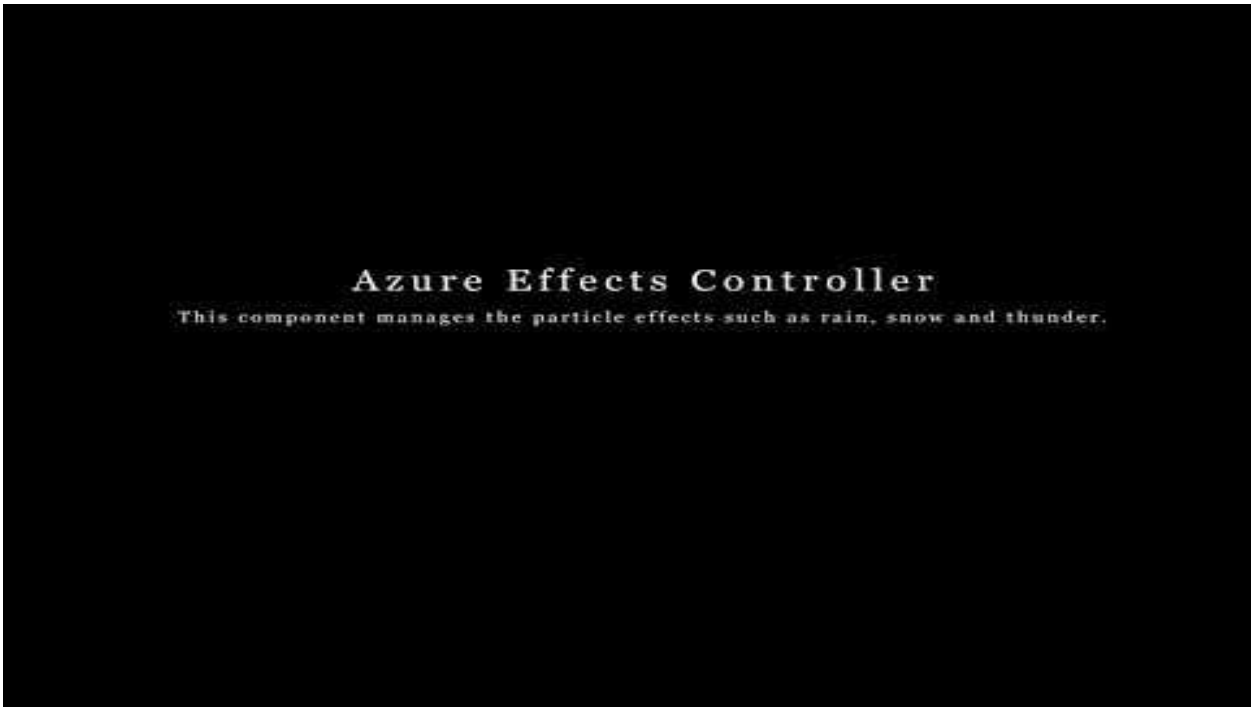
This component manages reflections and ambient colors.

YouTube video: [Azure Environment Controller Overview #5]

Azure Effects Controller

This component manages the particles effects such as rain, snow, thunder as well as the sound effects.

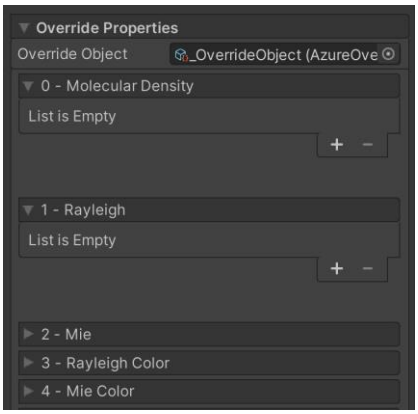
YouTube video: [Azure Effects Controller Overview #6]



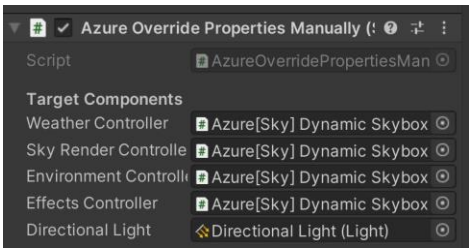
Override Properties by Scripting

Overriding a property automatically just by typing the component name and the target property name is easy to use, but it generates garbage allocation because this feature uses the FieldInfo.SetValue() and PropertyInfo.SetValue() methods from System.Reflection.

To fix this, you can simply skip the setting step for each override property from the Override Properties list of the AzureWeatherController component. Just leave each item on the list empty.



And attach the AzureOverridePropertiesManually.cs script to the prefab.



This script takes the output value of each override property and sends it to the target reference.

For each extra override property that you create you will need to edit the script by adding the reference of the component you want to access and override the target variable using the following methods of the AzureWeatherController component:

GetOverrideFloatOutput()

GetOverrideColorOutput()

See the AzureOverridePropertiesManually.cs code and just replicate the same process for your extra override property.

In the folder "Override Properties By Scripting" there is a sky prefab already configured for you to use.