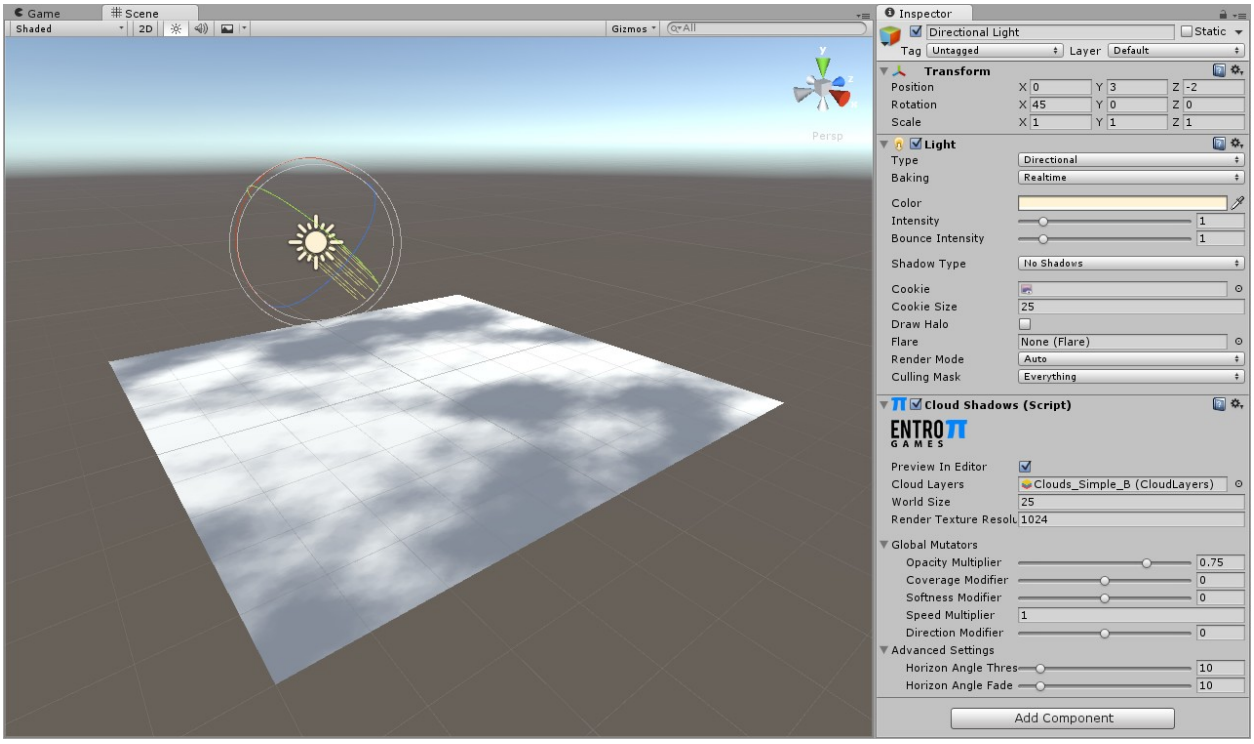
CLOUD SHADOWS

1. OVERVIEW

This system consists of a single component, the *Cloud Shadows* script component, which is attached to the *GameObject* containing the *Directional Light* illuminating the scene.

The look of the cloud shadows can be defined inside the *Cloud Layers* assets, which describe the different layers and how they should be combined together. *Cloud Layers* are similar to layers in an application such as Photoshop, except that *Cloud Layers* are blended together from top to bottom, instead of bottom to top.

The *Cloud Shadows* component merges all the layers defined inside the *Cloud Layers* asset togehter and renders them into a single render texture. This texture is then assigned to the cookie property of the *Directional Light* illuminating the scene. This cookie modulates the amount of light and projects the cloud shadows onto the scene.



2. GETTING STARTED

2.1 REQUIREMENTS

- The scene needs to contain a *Directional Light* which is used to simulate sunlight.
- This *Directional Light* needs to be rendered in real-time, baked light maps are **not** supported for this light(other lights can still use light maps).
- The rendering path must be set to either forward or deferred. Vertex lit is not supported.

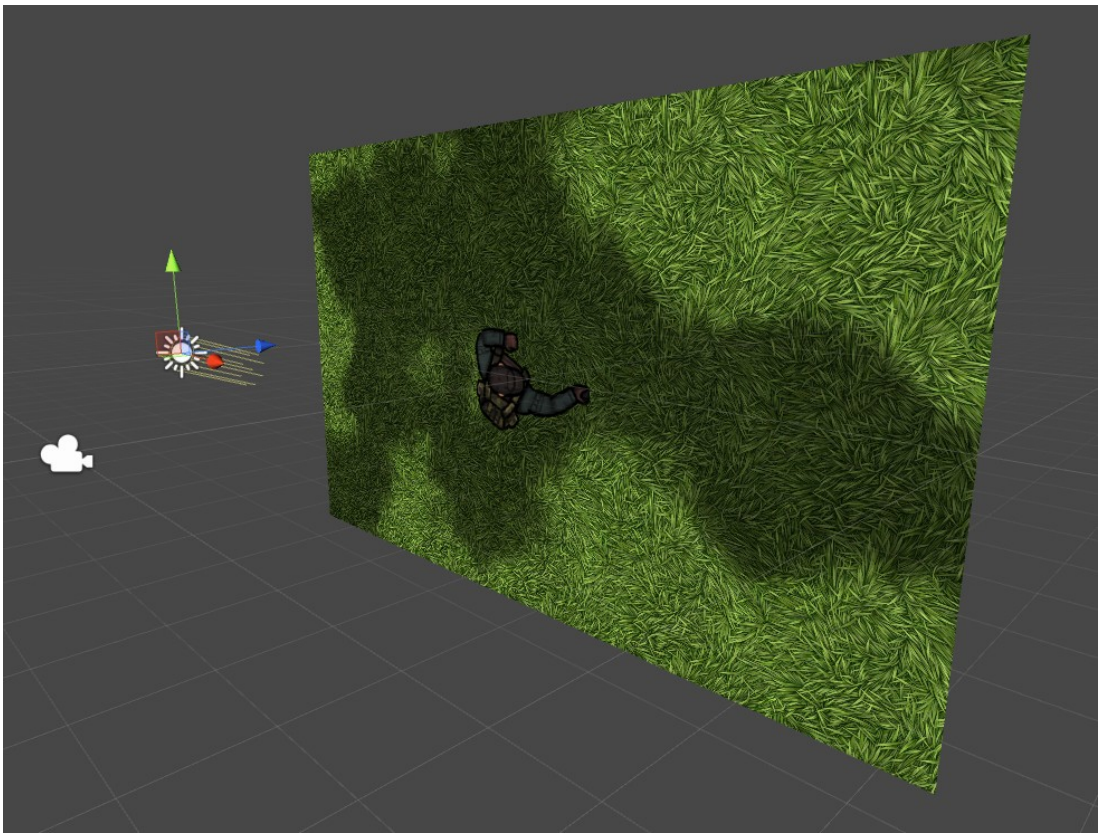
2.2 SETTING UP THE CLOUD SHADOWS COMPONENT

- Select the *GameObject* which contains the *Directional Light* component.
- Add the *Cloud Shadows* component: **Add Component ► EntroPi ► Cloud Shadows ► Cloud Shadows.**
- The cloud shadow effect should now be visible in the scene and game windows.

2.3 CHANGING THE CLOUD LAYERS ASSET IN THE CLOUD SHADOWS COMPONENT

- Select the *GameObject* which contains the *Directional Light* and *Cloud Shadows* component.
- Click the *Cloud Layer* property in the *Cloud Shadows* component and select one of the included *Cloud Layers* assets: **Assets/EntroPi/Cloud Shadows/Prefabs**

2.4 SETTING UP A 2D PROJECT



- Setting up a 2D project requires the same steps needed to set up a 3D project as outlined above.
- The sprites that need to be affected by the effect should be assigned the Diffuse sprite material: **Sprites ► Diffuse**
- Applying this material to the sprites will ensure they're lit by lights placed in the scene.

2.5 CREATING YOUR OWN CLOUD LAYERS ASSET

2.5.1 ASSET CREATION

- Navigate to the project folder where you want to store the *Cloud Layers* asset file.
- Click the RMB inside the folder and create a new Cloud Layers asset: **Create ► EntroPi ► Cloud Shadows ► Cloud Layers**
- With the asset selected, press the **+** button in the inspector window to add the first layer.
- Assign one of the included cloud textures to the *Texture* property of this newly created layer.
- Configure the cloud layer using the settings described in section 5.1.
- The order of the layers controls the order in which they are blended together. The bottom layers will be rendered on top of the top layers.

2.5.2 WORK FLOW TIPS

- You can tweak the *Cloud Layers* assets while the game is running. Because the data is serialized inside an asset file, changes you make in play mode will be stored after returning to the editor.
- Only one layer is required for the effect to render properly, but you can add multiple layers to achieve more complex effects.
- You can name the layers to keep them organized.
- You can expand, collapse and toggle the visibility of each layer using the buttons in the layer toolbar.
- You can easily move, duplicate or delete the layers using the buttons next to the layer name.

3. ADVANCED USAGE

3.1 CONVERTING PREFABS CREATED WITH OLDER VERSIONS OF THIS ASSET

Version 1.0 and 1.0.1 of this asset used *Cloud Layer* components (not to be confused with the new *Cloud Layer**S*** assets) grouped together in prefabs to configure the different layers. Version 1.1.0 includes a small tool to convert prefabs created with this legacy system to be converted into the new format.

- Follow steps 1 and 2 from section 2.5 to create a new *Cloud Layers* asset.
- Select the newly created *Cloud Layers* asset.
- Assign the prefab you want to convert to the slot next to the **Convert Prefab** button.
- Click the **Convert Prefab** button.
- All the layers should now have been copied into the new *Cloud Layers* asset.

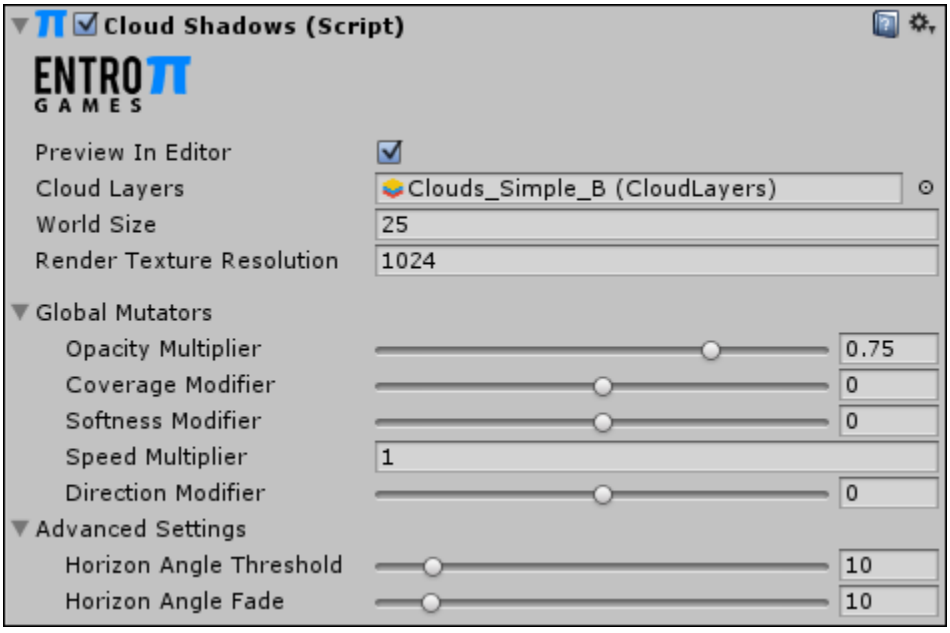
3.2 CREATING YOUR OWN CLOUD TEXTURES IN PHOTOSHOP

- Create a new image with a power of two resolution (ex. 1024x1024)
- Set the foreground color to white and the background color to black
- Render clouds using the clouds filter: **Filter ► Render ► Clouds**
- Adjust the image levels so that the grayscale values range from pure black to pure white.
- Import the texture in Unity
- Set the texture type to advanced and check the '*Bypass sRGB Sampling*' checkbox

For the coverage and softness properties of a cloud layer to work correctly, the texture needs to contain grayscale values which span the entire spectrum from black to white. The texture should also tile seamlessly.

4. COMPONENT OVERVIEW

4.1 CLOUD SHADOWS



This component combines and renders the layers defined in the *Cloud Layers assets*. This component should only be added to *GameObjects* that contain a *Directional Light* component. All the settings described below are exposed and can be accessed in code through C# properties.

The *Global Mutator* settings affect all of the layers defined in the Cloud Layers assets, but won't modify the data inside the asset itself.

4.1.1 SETTINGS

- **Preview in editor:** Toggle the visibility of the effect while in Edit Mode.
- **Cloud Layers:** Cloud Layers asset which defines the look of the effect.
- **World Size:** Size in world units that the cloud layer textures will be projected on.
- **Render texture resolution:** Resolution of the texture used to combine and render layers into.

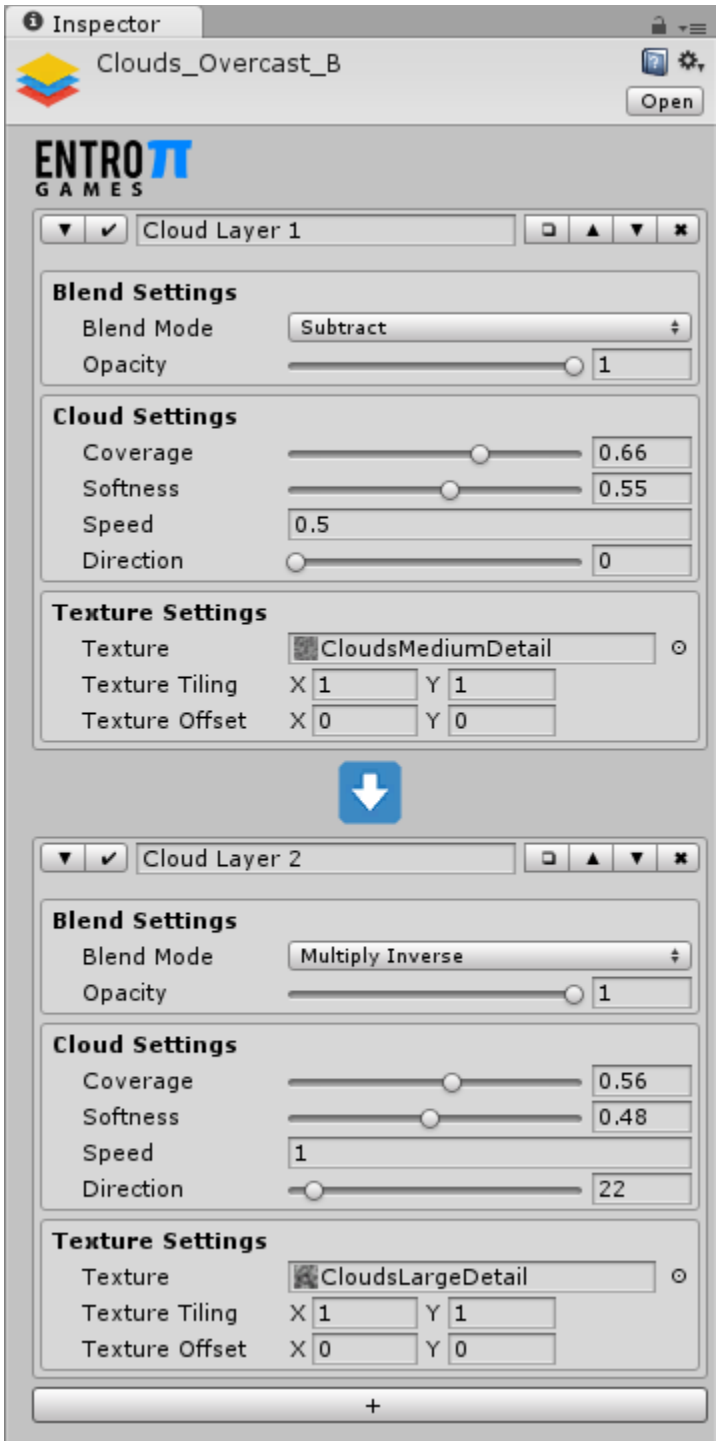
4.1.2 GLOBAL MUTATORS

- **Opacity Multiplier:** Multiplier which influences the opacity of all cloud layers
- **Coverage Modifier:** Modifies the coverage of all cloud layers
- **Softness Multiplier:** Modifies the softness of all cloud layers
- **Speed Multiplier:** Multiplier which influences the speed of all cloud layers
- **Direction Modifier:** Modifies the direction of all cloud layers

4.1.3 ADVANCED SETTINGS:

- **Horizon Angle Threshold:** The angle from the horizon at which the cloud shadows fade out completely.
- **Horizon Angle Fade:** The angle from the horizon over which the cloud shadows fade out.

4.2 CLOUD LAYERS (SCRIPTABLE OBJECT)



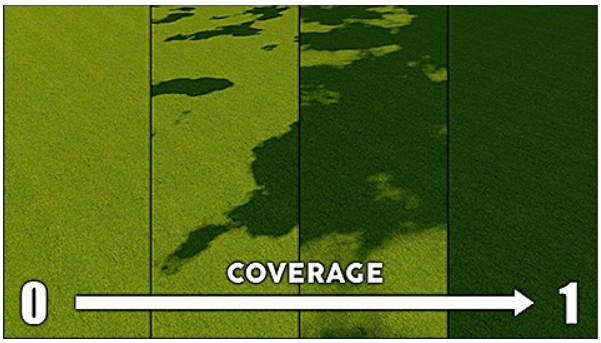
This *Scriptable Object* defines a collection of cloud layers, which is assigned to the cloud shadows component and then rendered.

Each layer can be configured using the settings described below. All of these settings are exposed and can be accessed in code through C# properties.

4.2.1 BLEND SETTINGS

- **Blend Mode:** Mode used for blending this layer. Identical to blend modes in Photoshop.
- **Opacity:** Opacity of this layer.

4.2.2 CLOUD SETTINGS



- **Coverage:** Cloud coverage (0 = no clouds, 1 = overcast)
- **Softness:** The softness of the clouds outline.
- **Speed:** Movement speed of the clouds
- **Direction:** Movement direction of the clouds in degrees

4.2.3 TEXTURE SETTINGS

- **Texture:** Cloud texture used by this layer."
- **Tiling:** Texture tiling. Values will be rounded to nearest int value.
- **Offset:** Initial texture offset.

5. DEPRECATED COMPONENTS

5.1 CLOUD LAYER (DEPRECATED IN VERSION 1.1.0)

This component defines a single cloud layer. It can either be added directly to the *GameObject* containing the Clouds Shadows component, or it can be used inside a prefab which is than assigned to the Cloud Shadows component. All the settings described below are exposed and can be accessed in code through C# properties.

5.1.1 TEXTURE SETTINGS

- **Texture:** Cloud texture used by this layer."
- **Tiling:** Texture tiling. Values will be rounded to nearest int value.
- **Offset:** Initial texture offset.

5.1.2 BLEND SETTINGS

- **Blend Mode:** Mode used for blending this layer. Identical to blend modes in Photoshop.
- **Opacity:** Opacity of this layer.

5.1.3 CLOUD SETTINGS

- **Coverage:** Cloud coverage (0 = no clouds, 1 = overcast)
- **Softness:** The softness of the clouds outline.
- **Velocity:** The velocity of the clouds in world space.

5.1.4 ADVANCED SETTINGS

- **Horizon Angle Threshold:** The angle from the horizon at which the cloud shadows fade out completely.
- **Horizon Angle Fade:** The angle from the horizon over which the cloud shadows fade out.

6. PERFORMANCE

6.1 OPTIMIZING

This system uses render textures and image blitting to animate, blend and combine several layers into a single texture. The resulting texture is than used as cookie for the Direction Light illuminating the scene. Each rendered Cloud Layer component requires a single drawcall, so adding more layers will decrease performance slightly.

Another property which has a direct impact on performance is the *Render Texture Resolution* property on the *Cloud Shadows* component. The larger the resolution, the longer rendering each layer will take. For optimal performance, limit the amount of layers used and keep the resolution of the render texture as small as possible.

7. CHANGELOG

7.1 VERSION 1.1.0

- Introduced new system for defining Cloud Layers using Scriptable Objects
- Added Global Mutators for coverage, softness, speed and direction
- Added Custom Editor script for all components
- Added extra prefabs
- Small shader fixes

8. SUPPORT

Feedback, suggestions or questions? Please send us a message using the support form on our website:

http://www.entropi-games.com/unity-asset-store-support/