Translucent Image

Release 2.0

Sep 04, 2017

Contents

Welcome to Translucent Image's documentation.

You need some basic knowledge about Unity in order to follow this documentation, if you don't, check out Unity official tutorials.

Also check out the demo (WebGL/Android) if you haven't.

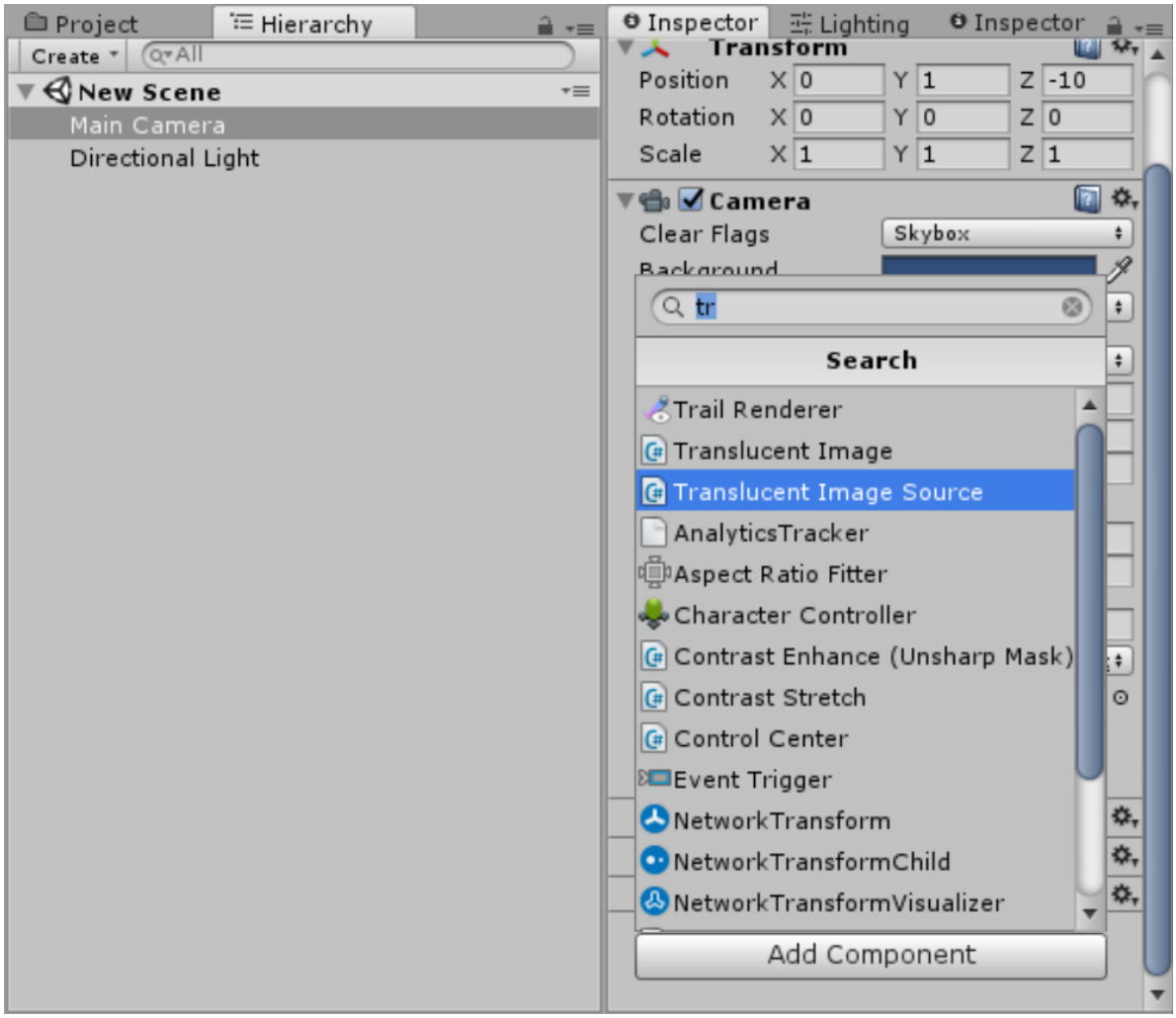
Use the sidebar or links below to browse.

CHAPTER 1

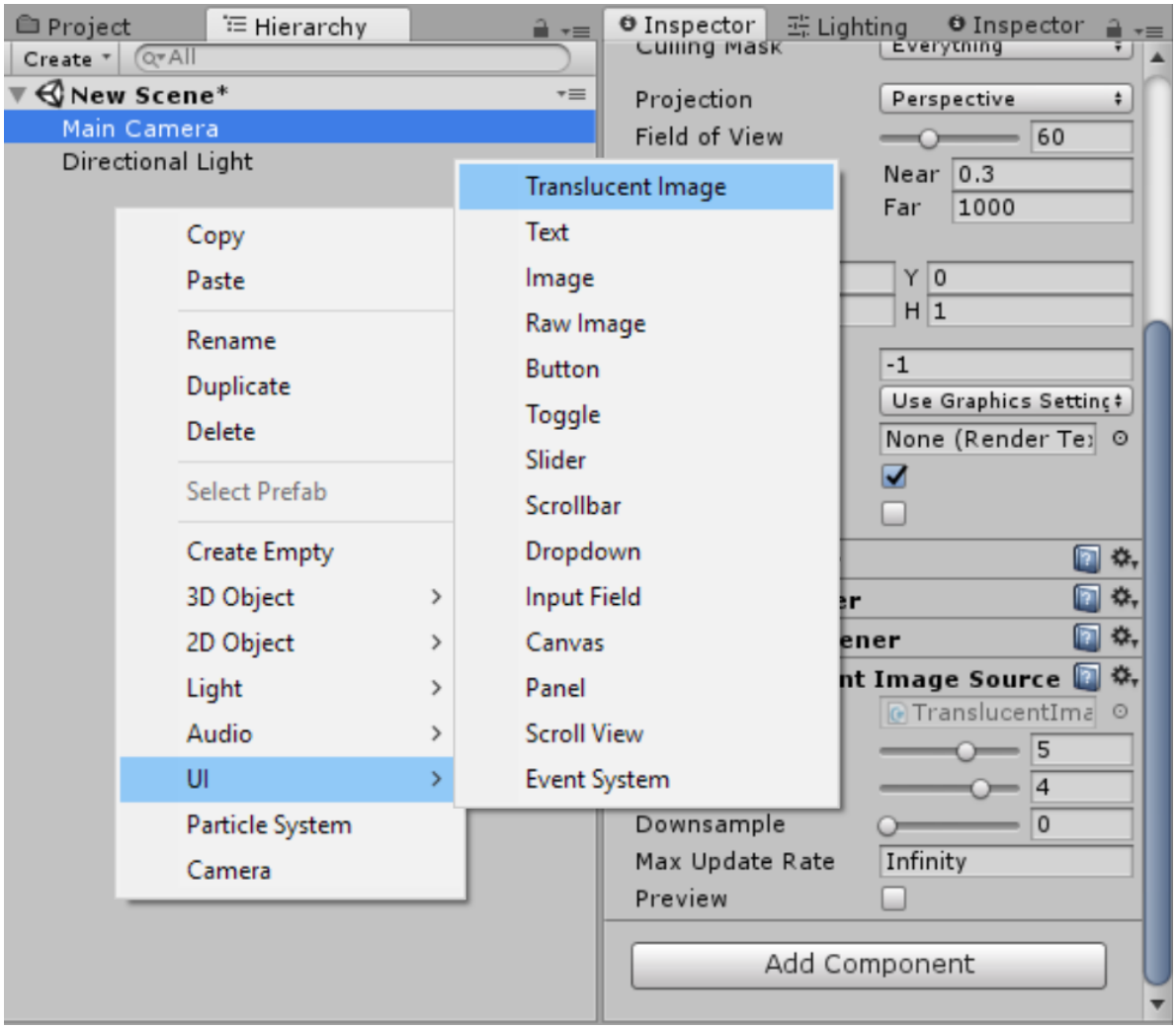
Manual

Getting Started

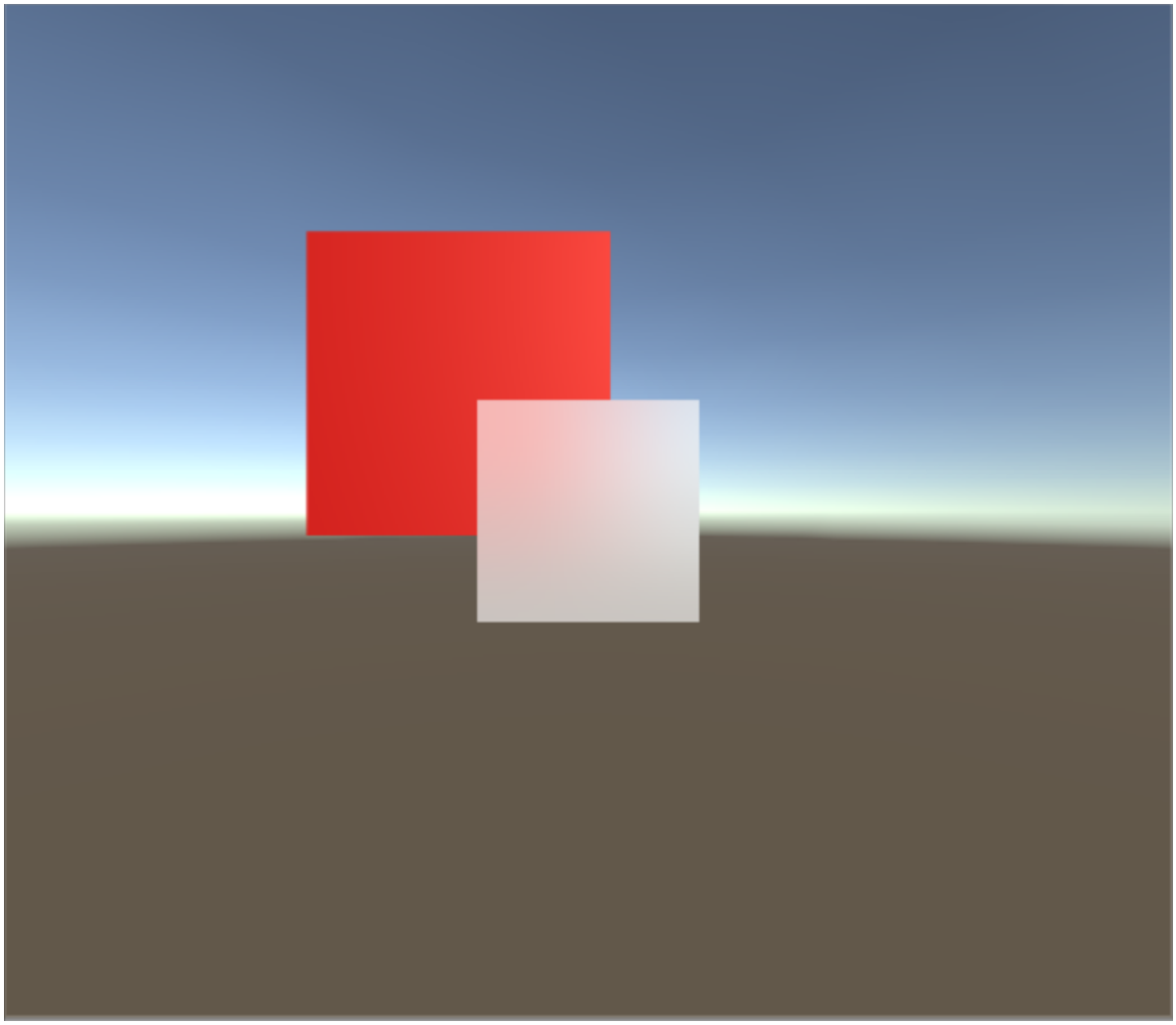
1. Add `Translucent Image Source` component to your main camera.



2. Create `UI -> Translucent Image` as you would with built-in UI.


Project
Hierarchy
Inspector
Lighting
Inspector
Create
All
New Scene*
Main Camera
Directional Light
Copy
Paste
Rename
Duplicate
Delete
Select Prefab
Create Empty
3D Object
2D Object
Light
Audio
UI
Particle System
Camera
Translucent Image
Text
Image
Raw Image
Button
Toggle
Slider
Scrollbar
Dropdown
Input Field
Canvas
Panel
Scroll View
Event System
Projection
Perspective
Field of View
60
Near
0.3
Far
1000
Y
0
H
1
-1
Use Graphics Setting
None (Render Te
Image Source
TranslucentIma
5
4
Downsample
0
Max Update Rate
Infinity
Preview
Add Component

3. That this!



Customize

Tip: This package was designed to be scalable. All properies that was said below to affect performance actually do so very little

There are 2 components that form the effect, both with their own parameter that affect the look of the effect:

Translucent Image Source

This component offers two modes of controlling the amount of blur: *Simple* and *Advanced*:

- ***Simple*:**
  - **Strength**. Using this property, you can (kinda) smoothly change the blur amount at runtime.
- ***Advanced*:**
  - **Size**: How much blurriness you want. Doesn't affect performance, but will look bad if the number too big. Also reduce flickering.

  - **Iteration**: Increase blur quality and blurriness when it is increased. The bigger it is the less performance loss when increasing further.
  - **Downsample**: Decrease the resolution before processing to increase performance. Side effect include increase blurriness and flickering.

There are also other properties that are independant of mode:

- **Max Depth**: Increase this property will:
  - Increase flickering when background moving
  - Increase blur level
  - Improve performance
- **Max Update Rate**: How many time the effect update itself per second. Use this property to increase performance and decrease power usage. Set to 0 to pause, this can reduce power usage/ prevent overheat when you don't need dynamically updating background - like in a pause menu for example.
- **Preview**: preview the effect in full-screen without creating a Translucent Image

Translucent Image

- **Source Image**: The sprite to use for this image.
- **Material**: Multiple Translucent Image using the same material can only have different color, but they can batch dynamically to only take one draw call.

> **Attention:** Material used here must use the shader UI/TranslucentImage

- **Color, Raycast Target, Image Type**: same as built-in Image.
- **Source**: Translucent Image Source component. This is where the image gets the blurred screen. It will automatically being set to the first one found, so you should make sure there one in your scene before creating any Translucent Image. You can always override this to change which camera will be blurred.
- **Vibrancy**: How colorful you want the background to be, 0 mean black and white, negative value will invert the color. This is great for enhancing the detail behind the image, or making death screen.
- **Brightness**: Brighten or darken the background.
- **Flatten**: Pull the color of the background closer together. This make it easier to ensure the legibility of your foreground element in case the background might change color a lot.

CHAPTER 2

Frequently Asked Questions

Can I blur other UI?

TL;DR: Kind of. See the demo.

If I run the blur algorithm once for each UI on the screen, it will get too expensive too fast. Infact, if you use a Mac or Window 10 computer, you will notice the blur effect is disabled for windows that aren't in focus.

For this reason, the blurring is done once per camera - it is what the TranslucentImageSource component do. The result of this is we get much higher performance, but every TranslucentImage share that same source will have the exact same background - which mean they will not "see" others below them.

If you really need to blur UIs, this can be done by setting up another camera and canvas. The new canvas will contain the UI that you want to blur. You can then set the new camera to only see that canvas with clear flag of "Depth only" or "Don't clear", and the other camera to not see it. Add a TranslucentImageSource to the new camera, and use it as the source for the top level UI. The UIs that are in the new canvas should use the old camera as source. **In short: see the demo.**

Have another question?

Contact me.

CHAPTER 3

Support

If you need assistance regarding the asset or have a feature request, feel free to contact me by the form below or through my support email.