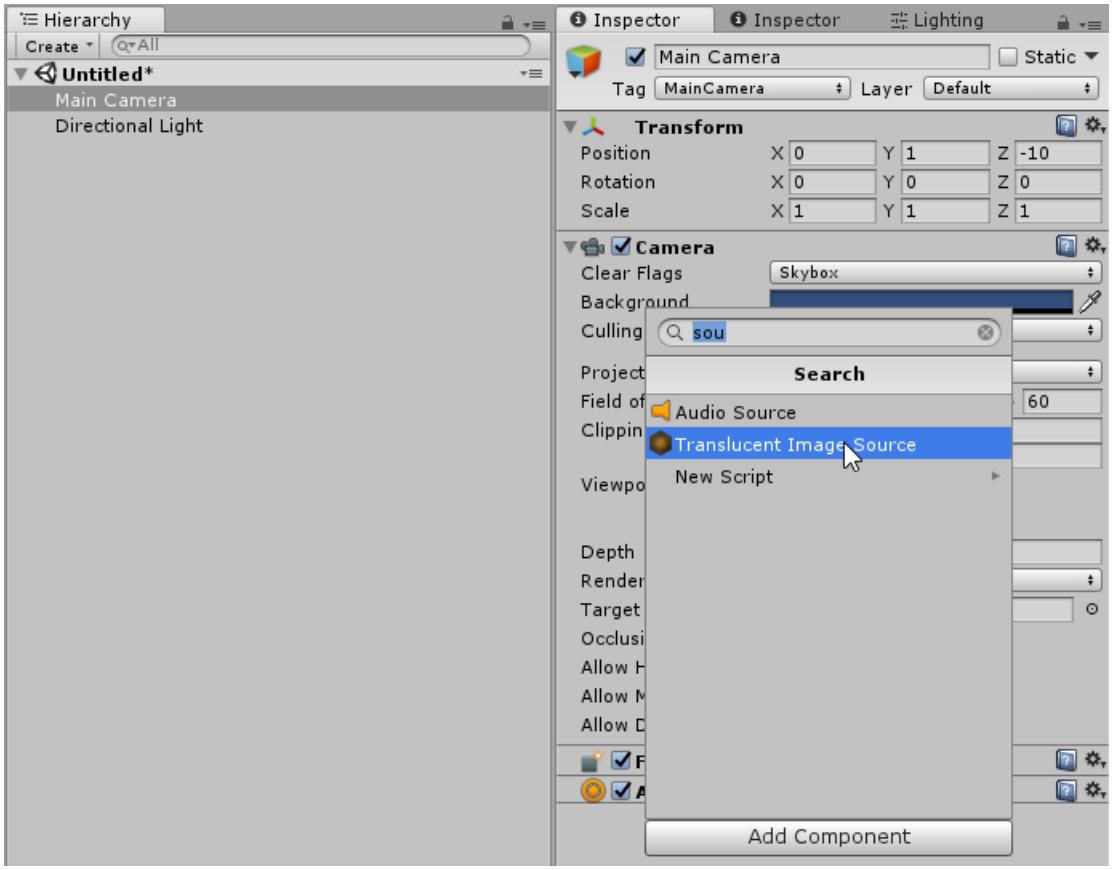
Table of Contents

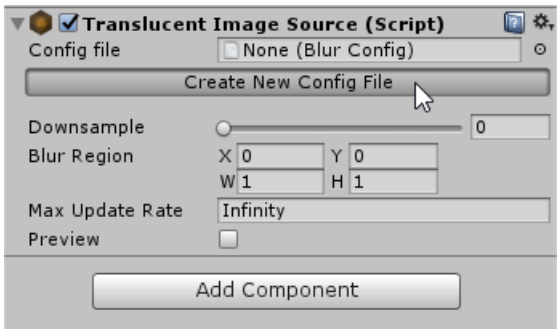
Getting Started

0. If you are using URP, follow the setup steps first.

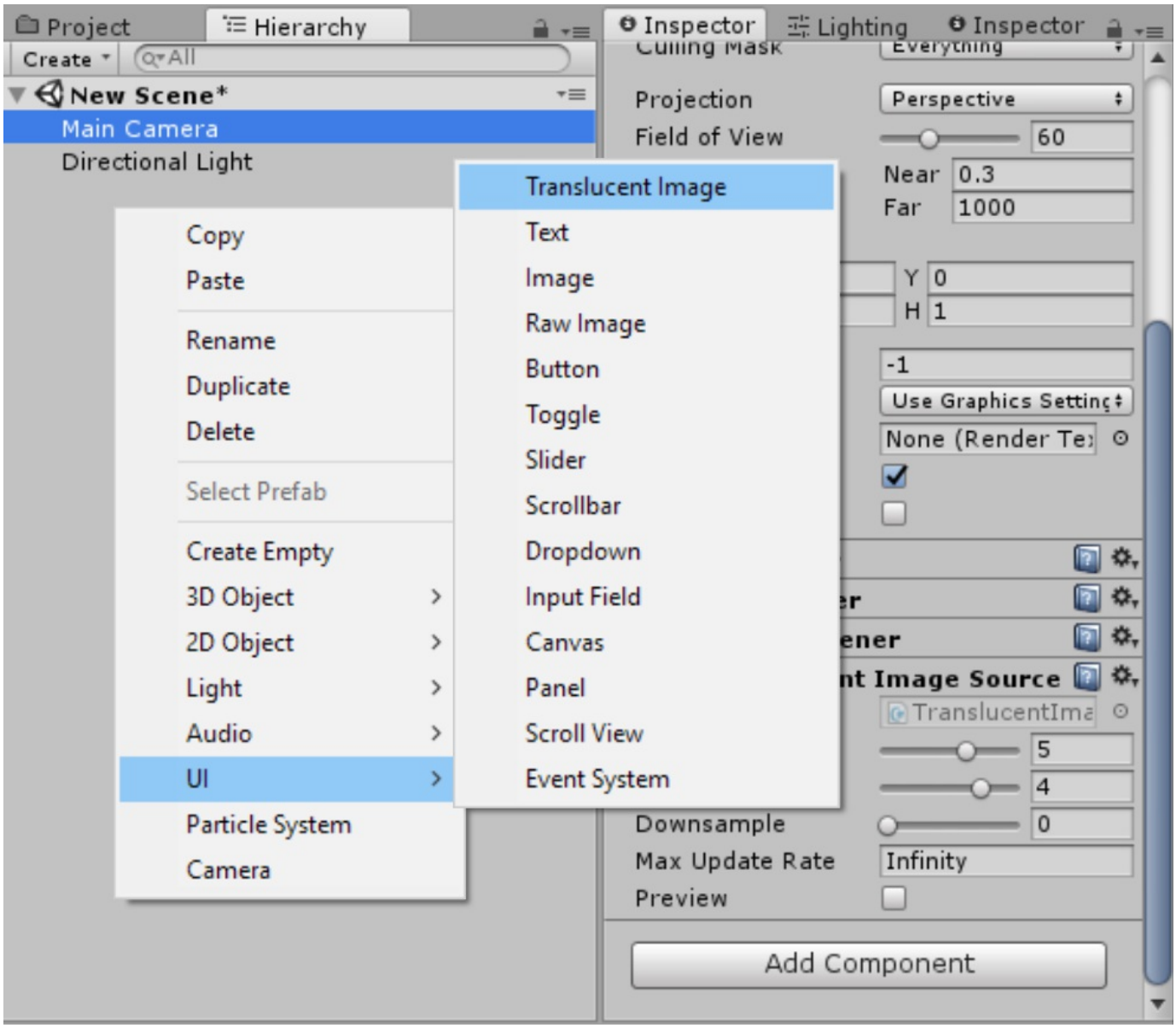
1. Add **Translucent Image Source** to your main camera.



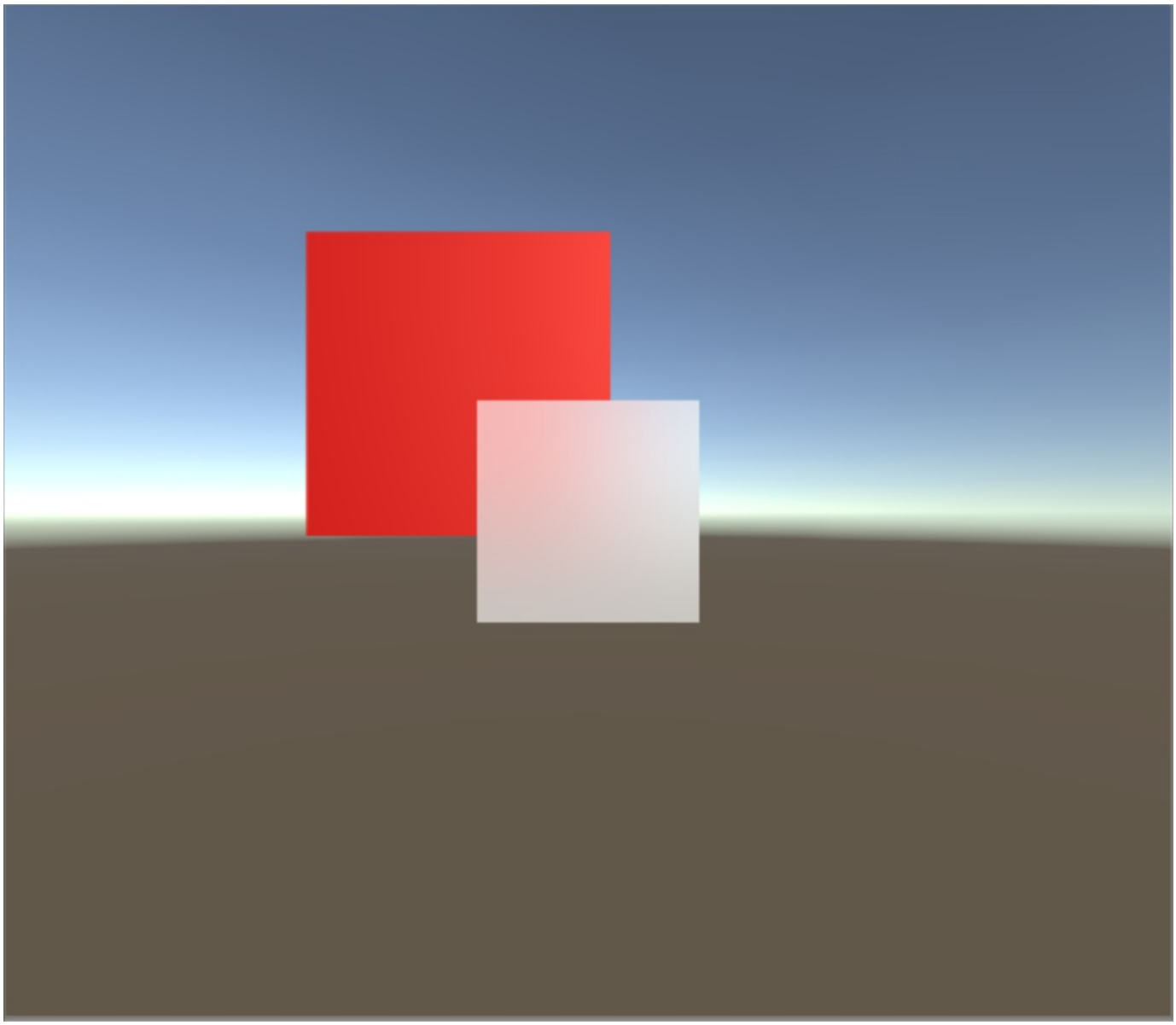
2. Create a ***Blur Config*** asset (or assign an existing one).



3. Create a ***UI > Translucent Image***, as you would with normal Image.


Project
Hierarchy
Inspector
Lighting
Inspector
Create
All
New Scene*
Main Camera
Directional Light
Culling Mask
Everything
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
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

4. That's it!



WARNING

By default, Translucent Image will use a default Material. To make sure your Translucent Image are not affected by asset update, create your own Material. See Customize section for more info.

NOTE

Sometime the effect does not shown up immediately. If that happen, just switch to play mode. The effect will continue to show even when exit play mode.

Customize

Translucent Image requires 2 components in a scene: 1 or more Sources that control the blur amount, and multiple Translucent Images that replace the built-in Image component.

Translucent Image Source

Translucent Image Source generate the blurred background. It lets you control how much blur is present, the quality of blur, and thus the performance trade-off.

Blur Amount

There are 2 methods of controlling the amount of blur, *Simple* and *Advanced*:

Simple:

- **Strength**. Using this single property, you can (mostly) smoothly change the blur amount at runtime.

Advanced:

- **Size**: Increase blur amount without affecting performance, but will look bad if set too high or too low. Reduce flickering when increased. Allow smooth interpolating between blur amounts, but can't go down to 0.
- **Iteration**: Increase blur quality and blurriness. Affects performance slightly. Animating this will create noncontinuous change in blur amount.

Performance:

These allow you to fine-tune the performance-quality trade-off. They're more useful for less capable mobile devices or high-resolution tablets.

- **Max Depth**: Increasing this property will:
    - Improves performance.
    - Increases blur amount if Iteration is high.
    - May cause flickering when the background moves.
- **Downsample**: Increasing to reduce the internal resolution and improve performance. At the cost of reduced quality, and may cause flickering.
- **Blur Region**: Limit the blur effect to a region of the screen. If your UI does not span the entire screen, it is a good idea to use this to increase performance and reduce power usage.

> TIP
>
> You can visualize and visually edit this by turning on Preview. The number here works the same as the Camera component Viewport. It's easier to wield if you change x and y before w and h.

- **Max Update Rate**: How many times the screen is blurred per second. Use this to improve performance and decrease power usage.

> TIP
>
> Setting this to 0 will pause the effect completely. This can reduce power usage/ prevent overheat when you don't need a dynamically updating background, for example, in a pause menu.

Other:

- **Background Fill**: Fill the background where the frame buffer alpha is 0. Useful for VR Underlay and Passthrough, where these areas would otherwise be black.
- **Preview**: Preview the effect in full-screen. It also shows the Blur Region as a resizable rectangle.

Translucent Image

- **Source Image, Color, Raycast Target, Image Type**: same as built-in Image.
- **Material**: Multiple Translucent Images using the same material share some settings. They will batch dynamically into a single draw call.

> **WARNING**
> - Materials used here must use the shader UI/TranslucentImage.
> - You should create your own Material instead of the default to avoid changes to look after asset updates.

- **Source**: a Translucent Image Source component. Will be automatically set to the first one found, so you should make sure there's one in your scene before creating any Translucent Image. You can change this to select which camera will provide the background.
- **Sprite Blending**: Mix between the Source Image and the blurred background. This should be what you would use instead of the alpha channel of the Color property most of the time.

Shared settings

The following settings are shared across Translucent Images using the same material:

- **Vibrancy**: Colorfulness of the background. 0 means monochrome, and a negative value will invert the color.
- **Brightness**: Brighten or darken the background.
- **Flatten**: Reduce the background contrast. Useful when you can't predict the color of the background but want to keep the content on top of it legible.

Controls Translucent Image from script

You can control all of the settings available in the inspector in C# through the exposed properties. See: TranslucentImage and TranslucentImageSource

The blur settings can't be accessed directly from the TranslucentImageSource class. They're are stored in a ScriptableObject. An easy way to access these settings is to create a public/serialized field. You can then assign the setting asset to it in the inspector:

```
public ScalableBlurConfig blurSettings;
```

Alternatively, if you have a reference to a TranslucentImageSource component, you can access the blur settings by casting it to a ScalableBlurConfig

```
void Start(){
    var theSource = FindObjectOfType<TranslucentImageSource>();
    var blurSettings = (ScalableBlurConfig) theSource.BlurConfig; // No other blur algorithms are available at the moment, so the cast will always success

    blurSettings.Strength = 42;
}
```

Universal Render Pipeline

Requirements

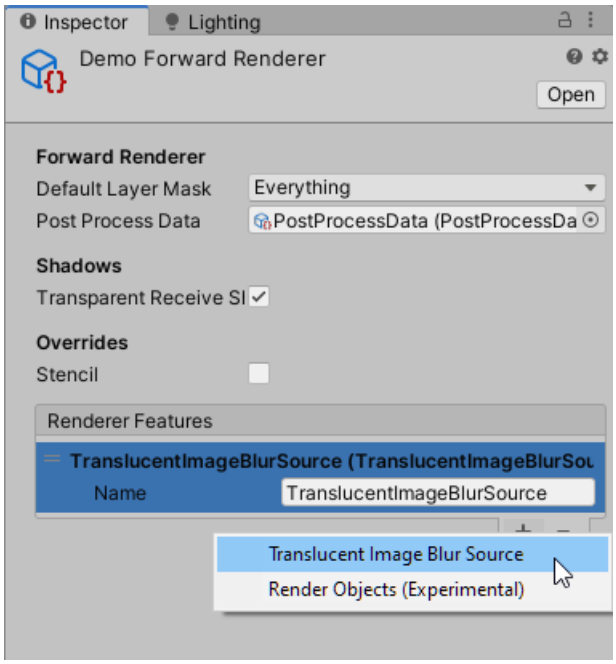
Only non-preview, non-beta version of URP and Unity will be supported.

The files required to support URP can be imported from the unitypackage at: `TranslucentImage/UniversalRP support`. They're not included by default, as that would produce errors in projects without URP.

Some demo scene does not work in URP. You can find demo scenes dedicated for URP under `TranslucentImage/Demo/UniversalRP`, after you import the support package.

Setup

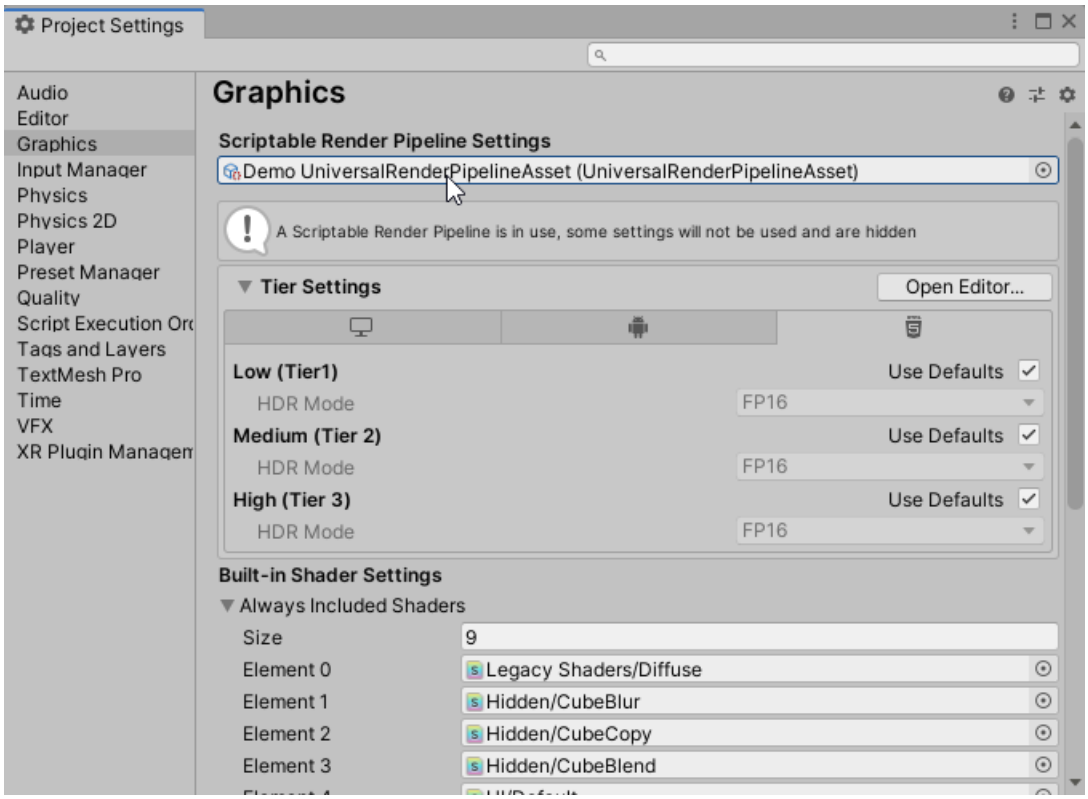
1. Import the package at `TranslucentImage/UniversalRP support`.
2. Find your **Forward Renderer Assets**, and add **TranslucentImageBlurSource** to its list of **Renderer Features**:



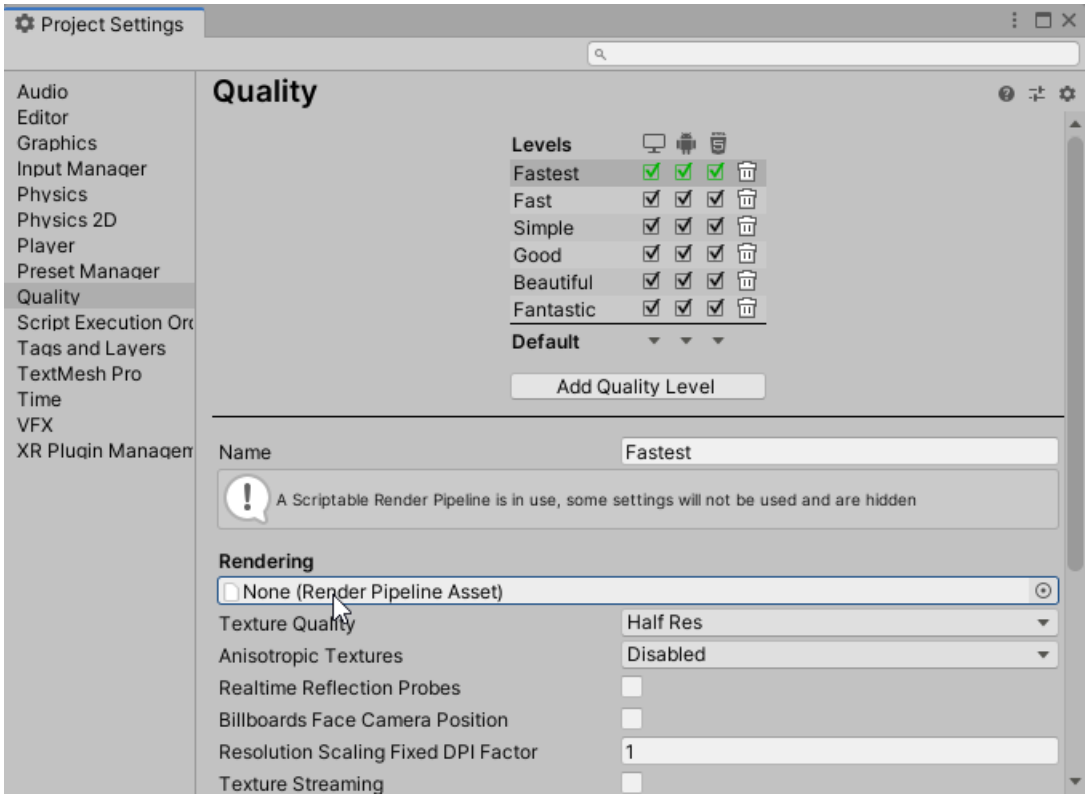
> **NOTE**
>
> You may have multiple **Forward Renderer Assets**. In which case you have to add the **Renderer Feature** to all of them.

Finding the Forward Renderer Assets

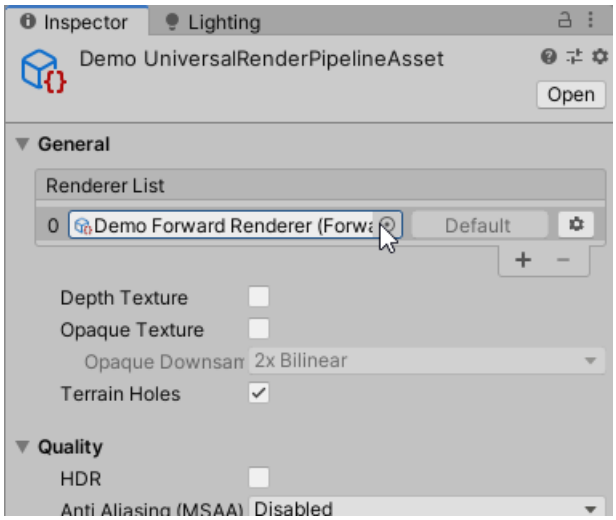
1. You can find the **Forward Renderer asset(s)** you're using by finding the **Render Pipeline Settings asset** in **Graphic Settings**:



2. You may also have more **Quality Setting**. Be sure to check all Quality Levels that you use:



3. Double click the field under the cursor in the above images will take you to the **Render Pipeline Settings asset**, where you can find your **Forward Renderer asset(s)** in the list of **Renderer**:


Inspector
Lighting
Demo UniversalRenderPipelineAsset
Open
General
Renderer List
0
Demo Forward Renderer (Forwa
Default
Depth Texture
Opaque Texture
Opaque Downsam
2x Bilinear
Terrain Holes
Quality
HDR
Anti Aliasing (MSAA)
Disabled

Blurring other UI elements

To achieve the best possible performance, Translucent Image does not support blurring UI elements in arbitrary order. However, UIs in a Canvas can blur UIs in different Canvas, by using additional Cameras.

> **TIP**
>
> The best way to learn is by example! Check out the included demo scenes that contain examples of blurring UIs:
>
> - For Builtin Render Pipeline: `Demo/_Scenes/Demo.unity`
> - For Universal Render Pipeline: `Demo/UniversalRP/UniversalRP Demo UI Blur.unity`

The idea

To understand the setup, it's useful to understand how Translucent Image works.

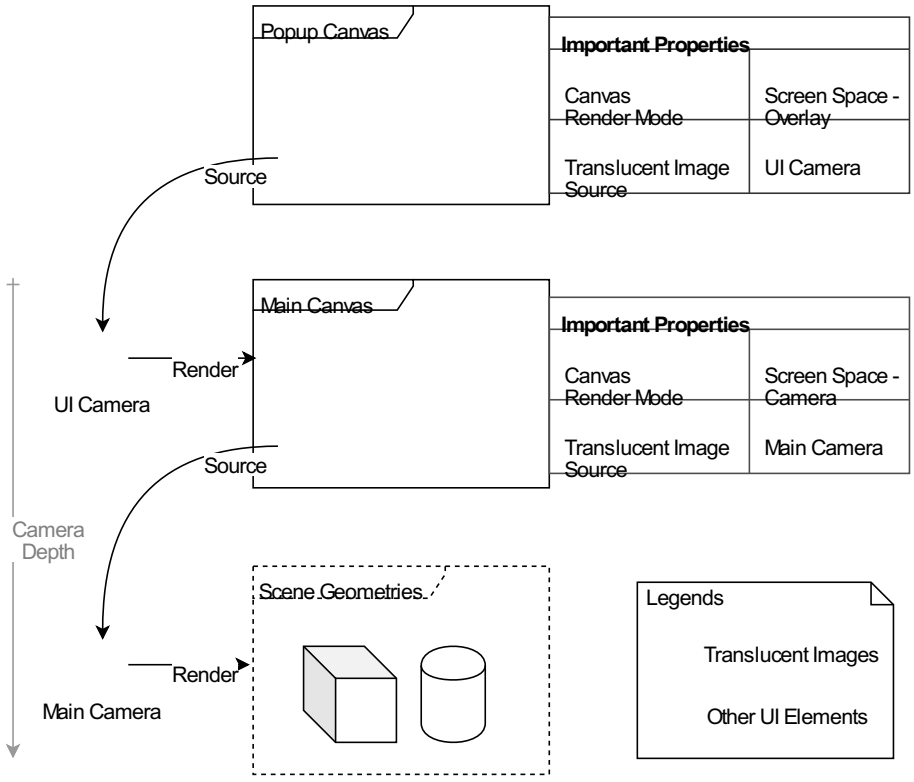
Translucent Images **do not blur** things by themselves. Instead, the Translucent Image Source component blurs the **entire** "screen" at once, then shares the result with multiple Translucent Images. This massively **reduces** the number of pixels that have to be blurred. It's one reason why Translucent Image is so fast.

The Translucent Image Source component "see" and blur what the Camera they attached to render. If the Source "sees" the Translucent Images, these Images will appear in their own background, causing a feedback loop and making the Images fade to opaque. It's important to make sure Translucent Images are **not** "seen" by their own Source.

So, to blur UIs, you need at least 2 Canvases: a lower one containing the UIs to be blurred, an upper one for the Translucent Images. Then, set the Camera with the Translucent Image Source to only render the lower Canvas.

An example setup

A common use of UI blurring is for pop-up panels. The following setup lets the Translucent Images in the Main Canvas blur the scene geometries. Those in the Popup Canvas will blur both the scene geometries, as well as UIs in the Main Canvas.



An example setup. What a Camera render can be controlled using its Culling Mask property

For reference, these are the important objects and settings:

```
├── Popup Canvas
│   ├── Layer: UI
│   └── Render Mode: Screen Space - Overlay
│
├── Main Canvas
│   ├── Layer: UI
│   ├── Render Mode: Screen Space - Camera
│   └── Render Camera: UI Camera
│
├── UI Camera
│   ├── Clear Flags: Depth only
│   ├── Culling Mask: UI
│   └── Depth: 2
│
└── Main Camera
    ├── Clear Flags: any
    ├── Culling Mask: Everything but UI
    └── Depth: 1
```

Performance implication

You can stack as many cameras and canvases as you like. However, with each extra Translucent Image Source you use, the GPU will have to do more work. A workaround is to disable the Source that is not the top-most. In fact, both Windows 10 and macOS do this:

Windows 10 only use blur on the top-most UI

World Space UI

World Space UI face the same problem as blurring other UIs - if we want to batch blurring operations to achieve high performance, we cannot have Translucent Images interleaved between what they want to blur. If you simply put Translucent Images in world space, they will continuously blur themselves, causing an "overexposed" effect.

To work around this, use a separated Camera for Translucent Images, an example of this setup available at: `Le Tai Asset/TranlucentImage/Demo/World Space UI`. Particularly, the World UI Camera should: - Have higher Depth than your Main Camera. - Have Culling Mask set to UI layer only. - Have `Depth only` clear Flags. - Other properties should match your Main Camera setting. - Be in the same position as your Main Camera - setting it as children with position and rotation of (0,0,0) is the simplest way.

Also, your Main Camera should have Culling Mask set to *exclude* UI layer.

Now, your Translucent Images should appear on top of scene geometry all the time, even if they are further away. While this is not ideal, it satisfies many use case, for example, world-space HUD, and allow for far better performance.

Frequently Asked Questions

Will this asset works well on my device?

The asset should run on any device. Performance-wise, it depends on your project's existing GPU consumption, but here some general rule of thumb:

- PC/Mac/Console: Should run well on almost everything except very old integrated GPU.
- Android: There're too many of them with too much difference in capability. The only way to know for sure is to test the demo on your target devices. On a Samsung Galaxy S7 Edge, the demo run at 60FPS with any setting.
- IOS:
    - Iphone: Apple A8 and later should hit 60FPS. A7 can hit 30FPS.
    - Ipad: Because of the higher pixel count, you'll need to use the resolution scale features to hit 60fps on A9 and below.

The blur does not work. UI in the demo scene just all white.

This usually due to one of the following:

- Sometime the effect only so up after switching to Play mode.
- If you're using URP, you have to do some setup, as detailed in the Universal Render Pipeline section first.
- Sometime Unity's import process break some random things. Try delete the whole folder and re-import the asset.

Can I smoothly animate the blur level?

The way the blur algorithm work make it difficult to smoothly animate the blur amount. The Strength property allows for mostly smooth change of blurriness, but there will still be some abrupt jump that is noticeable when interpolating slowly.

If you just need to fade in and out, you can use the alpha component of the Color property. You can also use Canvas Group as with normal Images.

Have another question?

Contact me

Support

If you need assistance regarding the asset or have a feature request, feel free to contact me by the form below or at:
https://letai.freshdesk.com/support/tickets/new

Support request Search Articles

Namespace LeTai.Asset.TranslucentImage

Classes

BackgroundFill

BlurConfig

Extensions

ScalableBlur

ScalableBlurConfig

ShaderId

Shims

TranslucentImage

Dynamic blur-behind UI element

TranslucentImageSource

Common source of blur for Translucent Images.

Interfaces

IBlurAlgorithm

Enums

BackgroundFillMode

BlurAlgorithmType

Class BackgroundFill

Inheritance

object

BackgroundFill

Inherited Members

object.Equals(object)
object.Equals(object, object)
object.GetHashCode()
object.GetType()
object.MemberwiseClone()
object.ReferenceEquals(object, object)
object.ToString()

Namespace: LeTai.Asset.TranslucentImage

Assembly: LeTai.TranslucentImage.dll

Syntax

```
[Serializable]
public class BackgroundFill
```

Fields

color

Declaration

```
public Color color
```

Field Value

TYPE	DESCRIPTION
Color	

mode

Declaration

```
public BackgroundFillMode mode
```

Field Value

TYPE	DESCRIPTION
BackgroundFillMode	

Enum BackgroundFillMode

Namespace: LeTai.Asset.TranslucentImage

Assembly: LeTai.TranslucentImage.dll

Syntax

```
public enum BackgroundFillMode
```

Fields

NAME	DESCRIPTION
Color	
None	

Enum BlurAlgorithmType

Namespace: LeTai.Asset.TranslucentImage

Assembly: LeTai.TranslucentImage.dll

Syntax

```
public enum BlurAlgorithmType
```

Fields

NAME	DESCRIPTION
ScalableBlur	

Class BlurConfig

Inheritance

object

Object

ScriptableObject

BlurConfig

ScalableBlurConfig

Inherited Members

ScriptableObject.SetDirty()

ScriptableObject.CreateInstance(string)

ScriptableObject.CreateInstance(Type)

ScriptableObject.CreateInstance<T>()

Object.GetInstanceID()

Object.GetHashCode()

Object.Equals(object)

Object.Instantiate(Object, Vector3, Quaternion)

Object.Instantiate(Object, Vector3, Quaternion, Transform)

Object.Instantiate(Object)

Object.Instantiate(Object, Transform)

Object.Instantiate(Object, Transform, bool)

Object.Instantiate<T>(T)

Object.Instantiate<T>(T, Vector3, Quaternion)

Object.Instantiate<T>(T, Vector3, Quaternion, Transform)

Object.Instantiate<T>(T, Transform)

Object.Instantiate<T>(T, Transform, bool)

Object.Destroy(Object, float)

Object.Destroy(Object)

Object.DestroyImmediate(Object, bool)

Object.DestroyImmediate(Object)

Object.FindObjectsOfType(Type)

Object.DontDestroyOnLoad(Object)

Object.DestroyObject(Object, float)

Object.DestroyObject(Object)

Object.FindSceneObjectsOfType(Type)

Object.FindObjectsOfTypeIncludingAssets(Type)

Object.FindObjectsOfType<T>()

Object.FindObjectOfType<T>()

Object.FindObjectsOfTypeAll(Type)

Object.FindObjectOfType(Type)

Object.ToString()

Object.name

Object.hideFlags

object.Equals(object, object)

object.GetType()

object.MemberwiseClone()

object.ReferenceEquals(object, object)

Namespace: LeTai.Asset.TranslucentImage

Assembly: LeTai.TranslucentImage.dll

Syntax

```
public class BlurConfig : ScriptableObject
```

Class Extensions

Inheritance

object

Extensions

Inherited Members

object.Equals(object)
object.Equals(object, object)
object.GetHashCode()
object.GetType()
object.MemberwiseClone()
object.ReferenceEquals(object, object)
object.ToString()

Namespace: LeTai.Asset.TranslucentImage

Assembly: LeTai.TranslucentImage.dll

Syntax

```
public static class Extensions
```

Methods

BlitCustom(CommandBuffer, RenderTargetIdentifier, RenderTargetIdentifier, Material, int, bool)

Declaration

```
public static void BlitCustom(this CommandBuffer cmd, RenderTargetIdentifier source, RenderTargetIdentifier destination, Material material, int passIndex, bool useBuiltin = false)
```

Parameters

TYPE	NAME	DESCRIPTION
CommandBuffer	cmd	
RenderTargetIdentifier	source	
RenderTargetIdentifier	destination	
Material	material	
int	passIndex	
bool	useBuiltin	

BlitFullscreenTriangle(CommandBuffer, RenderTargetIdentifier, RenderTargetIdentifier, Material, int)

Declaration

```
public static void BlitFullscreenTriangle(this CommandBuffer cmd, RenderTargetIdentifier source, RenderTargetIdentifier destination, Material material, int pass)
```

Parameters

TYPE	NAME	DESCRIPTION
CommandBuffer	cmd	

TYPE	NAME	DESCRIPTION
RenderTargetIdentifier	source	
RenderTargetIdentifier	destination	
Material	material	
int	pass	

BlitProcedural(CommandBuffer, RenderTargetIdentifier, RenderTargetIdentifier, Material, int)

Declaration

```
public static void BlitProcedural(this CommandBuffer cmd, RenderTargetIdentifier source, RenderTargetIdentifier destination, Material material, int passIndex)
```

Parameters

TYPE	NAME	DESCRIPTION
CommandBuffer	cmd	
RenderTargetIdentifier	source	
RenderTargetIdentifier	destination	
Material	material	
int	passIndex	

ToMinMaxVector(Rect)

Declaration

```
public static Vector4 ToMinMaxVector(this Rect self)
```

Parameters

TYPE	NAME	DESCRIPTION
Rect	self	

Returns

TYPE	DESCRIPTION
Vector4	

ToVector4(Rect)

Declaration

```
public static Vector4 ToVector4(this Rect self)
```

Parameters

TYPE	NAME	DESCRIPTION
Rect	self	

Returns

TYPE	DESCRIPTION
Vector4	

Interface IBlurAlgorithm

Namespace: LeTai.Asset.TranslucentImage

Assembly: LeTai.TranslucentImage.dll

Syntax

```
public interface IBlurAlgorithm
```

Methods

Blur(CommandBuffer, RenderTargetIdentifier, Rect, BackgroundFill, RenderTexture)

Declaration

```
void Blur(CommandBuffer cmd, RenderTargetIdentifier src, Rect srcCropRegion, BackgroundFill backgroundFill, RenderTexture target)
```

Parameters

TYPE	NAME	DESCRIPTION
CommandBuffer	cmd	
RenderTargetIdentifier	src	
Rect	srcCropRegion	
BackgroundFill	backgroundFill	
RenderTexture	target	

Init(BlurConfig, bool)

Declaration

```
void Init(BlurConfig config, bool isBirp)
```

Parameters

TYPE	NAME	DESCRIPTION
BlurConfig	config	
bool	isBirp	

Class ScalableBlur

Inheritance

object

ScalableBlur

Implements

IBlurAlgorithm

Inherited Members

object.Equals(object)
object.Equals(object, object)
object.GetHashCode()
object.GetType()
object.MemberwiseClone()
object.ReferenceEquals(object, object)
object.ToString()

Namespace: LeTai.Asset.TranslucentImage

Assembly: LeTai.TranslucentImage.dll

Syntax

```
public class ScalableBlur : IBlurAlgorithm
```

Methods

Blur(CommandBuffer, RenderTargetIdentifier, Rect, BackgroundFill, RenderTexture)

Declaration

```
public void Blur(CommandBuffer cmd, RenderTargetIdentifier src, Rect srcCropRegion, BackgroundFill backgroundFill, RenderTexture target)
```

Parameters

TYPE	NAME	DESCRIPTION
CommandBuffer	cmd	
RenderTargetIdentifier	src	
Rect	srcCropRegion	
BackgroundFill	backgroundFill	
RenderTexture	target	

BlurAtDepth(CommandBuffer, int, RenderTexture)

Declaration

```
protected virtual void BlurAtDepth(CommandBuffer cmd, int depth, RenderTexture baseTexture)
```

Parameters

TYPE	NAME	DESCRIPTION
CommandBuffer	cmd	

TYPE	NAME	DESCRIPTION
int	depth	
RenderTexture	baseTexture	

ConfigMaterial(float, Vector4, BackgroundFill)

Declaration

```
protected void ConfigMaterial(float radius, Vector4 cropRegion, BackgroundFill backgroundFill)
```

Parameters

TYPE	NAME	DESCRIPTION
float	radius	
Vector4	cropRegion	
BackgroundFill	backgroundFill	

Init(BlurConfig, bool)

Declaration

```
public void Init(BlurConfig config, bool isBirp)
```

Parameters

TYPE	NAME	DESCRIPTION
BlurConfig	config	
bool	isBirp	

SimplePingPong(int, int)

Declaration

```
public static int SimplePingPong(int t, int max)
```

Parameters

TYPE	NAME	DESCRIPTION
int	t	
int	max	

Returns

TYPE	DESCRIPTION
int	

Implements

IBlurAlgorithm

Class ScalableBlurConfig

Inheritance

object

Object

ScriptableObject

BlurConfig

ScalableBlurConfig

Inherited Members

ScriptableObject.SetDirty()

ScriptableObject.CreateInstance(string)

ScriptableObject.CreateInstance(Type)

ScriptableObject.CreateInstance<T>()

Object.GetInstanceID()

Object.GetHashCode()

Object.Equals(object)

Object.Instantiate(Object, Vector3, Quaternion)

Object.Instantiate(Object, Vector3, Quaternion, Transform)

Object.Instantiate(Object)

Object.Instantiate(Object, Transform)

Object.Instantiate(Object, Transform, bool)

Object.Instantiate<T>(T)

Object.Instantiate<T>(T, Vector3, Quaternion)

Object.Instantiate<T>(T, Vector3, Quaternion, Transform)

Object.Instantiate<T>(T, Transform)

Object.Instantiate<T>(T, Transform, bool)

Object.Destroy(Object, float)

Object.Destroy(Object)

Object.DestroyImmediate(Object, bool)

Object.DestroyImmediate(Object)

Object.FindObjectsOfType(Type)

Object.DontDestroyOnLoad(Object)

Object.DestroyObject(Object, float)

Object.DestroyObject(Object)

Object.FindSceneObjectsOfType(Type)

Object.FindObjectsOfTypeIncludingAssets(Type)

Object.FindObjectsOfType<T>()

Object.FindObjectOfType<T>()

Object.FindObjectsOfTypeAll(Type)

Object.FindObjectOfType(Type)

Object.ToString()

Object.name

Object.hideFlags

object.Equals(object, object)

object.GetType()

object.MemberwiseClone()

object.ReferenceEquals(object, object)

Namespace: LeTai.Asset.TranslucentImage

Assembly: LeTai.TranslucentImage.dll

Syntax

```
[CreateAssetMenu(fileName = "New Scalable Blur Config", menuName = "Translucent Image/ Scalable Blur Config", order = 100)]
public class ScalableBlurConfig : BlurConfig
```

Properties

Iteration

Half the number of time to process the image. It is half because the real number of iteration must alway be even. Using half also make calculation simpler

Declaration

```
public int Iteration { get; set; }
```

Property Value

TYPE	DESCRIPTION
int	Must be non-negative

MaxDepth

Clamp the minimum size of the intermediate texture. Reduce flickering and blur

Declaration

```
public int MaxDepth { get; set; }
```

Property Value

TYPE	DESCRIPTION
int	Must larger than 0

Radius

Distance between the base texel and the texel to be sampled.

Declaration

```
public float Radius { get; set; }
```

Property Value

TYPE	DESCRIPTION
float	

Strength

User friendly property to control the amount of blur

Declaration

```
public float Strength { get; set; }
```

Property Value

TYPE	DESCRIPTION
float	Must be non-negative

Methods

SetAdvancedFieldFromSimple()

Calculate size and iteration from strength

Declaration

```
protected virtual void SetAdvancedFieldFromSimple()
```

Class ShaderId

Inheritance

object

ShaderId

Inherited Members

object.Equals(object)

object.Equals(object, object)

object.GetHashCode()

object.GetType()

object.MemberwiseClone()

object.ReferenceEquals(object, object)

object.ToString()

Namespace: LeTai.Asset.TranslucentImage

Assembly: LeTai.TranslucentImage.dll

Syntax

```
public static class ShaderId
```

Fields

BACKGROUND_COLOR

Declaration

```
public static readonly int BACKGROUND_COLOR
```

Field Value

TYPE	DESCRIPTION
int	

COLOR

Declaration

```
public static readonly int COLOR
```

Field Value

TYPE	DESCRIPTION
int	

CROP_REGION

Declaration

```
public static readonly int CROP_REGION
```

Field Value

TYPE	DESCRIPTION
int	

MAIN_TEX

Declaration

```
public static readonly int MAIN_TEX
```

Field Value

TYPE	DESCRIPTION
int	

RADIUS

Declaration

```
public static readonly int RADIUS
```

Field Value

TYPE	DESCRIPTION
int	

intermediateRT

Declaration

```
public static int[] intermediateRT
```

Field Value

TYPE	DESCRIPTION
int[]	

Methods

Init(int)

Declaration

```
public static void Init(int stackDepth)
```

Parameters

TYPE	NAME	DESCRIPTION
int	stackDepth	

Class Shims

Inheritance

object

Shims

Inherited Members

object.Equals(object)

object.Equals(object, object)

object.GetHashCode()

object.GetType()

object.MemberwiseClone()

object.ReferenceEquals(object, object)

object.ToString()

Namespace: LeTai.Asset.TranslucentImage

Assembly: LeTai.TranslucentImage.dll

Syntax

```
public static class Shims
```

Methods

FindObjectOfType<T>(bool, bool)

Declaration

```
public static T FindObjectOfType<T>(bool includeInactive = false, bool sorted = true) where T : Object
```

Parameters

TYPE	NAME	DESCRIPTION
bool	includeInactive	
bool	sorted	

Returns

TYPE	DESCRIPTION
T	

Type Parameters

NAME	DESCRIPTION
T	

FindObjectsOfType<T>(bool)

Declaration

```
public static T[] FindObjectsOfType<T>(bool includeInactive = false) where T : Object
```

Parameters

TYPE	NAME	DESCRIPTION
bool	includeInactive	

Returns

TYPE	DESCRIPTION
T[]	

Type Parameters

NAME	DESCRIPTION
T	

Class TranslucentImage

Dynamic blur-behind UI element

Inheritance

object
Object
Component
Behaviour
MonoBehaviour
UIBehaviour
Graphic
MaskableGraphic
Image
TranslucentImage

Implements

ICanvasElement
IClippable
IMaskable
IMaterialModifier
ISerializationCallbackReceiver
ILayoutElement
ICanvasRaycastFilter
IMeshModifier

Inherited Members

Image.s_ETC1DefaultUI
Image.sprite
Image.DisableSpriteOptimizations()
Image.overrideSprite
Image.type
Image.preserveAspect
Image.fillCenter
Image.fillMethod
Image.fillAmount
Image.fillClockwise
Image.fillOrigin
Image.eventAlphaThreshold
Image.alphaHitTestMinimumThreshold
Image.useSpriteMesh
Image.defaultETC1GraphicMaterial
Image.mainTexture
Image.hasBorder
Image.pixelsPerUnitMultiplier
Image.pixelsPerUnit
Image.multipliedPixelsPerUnit
Image.material
Image.OnBeforeSerialize()
Image.OnAfterDeserialize()
Image.SetNativeSize()
Image.OnPopulateMesh(VertexHelper)
Image.UpdateMaterial()

Image.OnCanvasHierarchyChanged()
Image.CalculateLayoutInputHorizontal()
Image.CalculateLayoutInputVertical()
Image.minWidth
Image.preferredWidth
Image.flexibleWidth
Image.minHeight
Image.preferredHeight
Image.flexibleHeight
Image.layoutPriority
Image.IsRaycastLocationValid(Vector2, Camera)
MaskableGraphic.m_ShouldRecalculateStencil
MaskableGraphic.m_MaskMaterial
MaskableGraphic.onCullStateChanged
MaskableGraphic.maskable
MaskableGraphic.isMaskingGraphic
MaskableGraphic.m_StencilValue
MaskableGraphic.GetModifiedMaterial(Material)
MaskableGraphic.Cull(Rect, bool)
MaskableGraphic.SetClipRect(Rect, bool)
MaskableGraphic.SetClipSoftness(Vector2)
MaskableGraphic.OnTransformParentChanged()
MaskableGraphic.RecalculateClipping()
MaskableGraphic.RecalculateMasking()
Graphic.s_DefaultUI
Graphic.s_WhiteTexture
Graphic.defaultGraphicMaterial
Graphic.m_Material
Graphic.m_SkipLayoutUpdate
Graphic.m_SkipMaterialUpdate
Graphic.color
Graphic.raycastTarget
Graphic.m_OnDirtyLayoutCallback
Graphic.m_OnDirtyVertsCallback
Graphic.m_OnDirtyMaterialCallback
Graphic.s_Mesh
Graphic.m_CachedMesh
Graphic.m_CachedUvs
Graphic.useLegacyMeshGeneration
Graphic.SetAllDirty()
Graphic.SetLayoutDirty()
Graphic.SetVerticesDirty()
Graphic.SetMaterialDirty()
Graphic.OnRectTransformDimensionsChange()
Graphic.OnBeforeTransformParentChanged()
Graphic.depth
Graphic.rectTransform
Graphic.canvas
Graphic.canvasRenderer
Graphic.defaultMaterial
Graphic.materialForRendering
Graphic.OnDestroy()

Graphic.OnCullingChanged()
Graphic.Rebuild(CanvasUpdate)
Graphic.LayoutComplete()
Graphic.GraphicUpdateComplete()
Graphic.UpdateGeometry()
Graphic.workerMesh
Graphic.OnPopulateMesh(Mesh)
Graphic.Raycast(Vector2, Camera)
Graphic.PixelAdjustPoint(Vector2)
Graphic.GetPixelAdjustedRect()
Graphic.CrossFadeColor(Color, float, bool, bool)
Graphic.CrossFadeColor(Color, float, bool, bool, bool)
Graphic.CrossFadeAlpha(float, float, bool)
Graphic.RegisterDirtyLayoutCallback(UnityAction)
Graphic.UnregisterDirtyLayoutCallback(UnityAction)
Graphic.RegisterDirtyVerticesCallback(UnityAction)
Graphic.UnregisterDirtyVerticesCallback(UnityAction)
Graphic.RegisterDirtyMaterialCallback(UnityAction)
Graphic.UnregisterDirtyMaterialCallback(UnityAction)
UIBehaviour.Awake()
UIBehaviour.IsActive()
UIBehaviour.OnCanvasGroupChanged()
UIBehaviour.IsDestroyed()
MonoBehaviour.IsInvoking()
MonoBehaviour.CancelInvoke()
MonoBehaviour.Invoke(string, float)
MonoBehaviour.InvokeRepeating(string, float, float)
MonoBehaviour.CancelInvoke(string)
MonoBehaviour.IsInvoking(string)
MonoBehaviour.StartCoroutine(string)
MonoBehaviour.StartCoroutine(string, object)
MonoBehaviour.StartCoroutine(IEnumerator)
MonoBehaviour.StartCoroutine_Auto(IEnumerator)
MonoBehaviour.StopCoroutine(IEnumerator)
MonoBehaviour.StopCoroutine(Coroutine)
MonoBehaviour.StopCoroutine(string)
MonoBehaviour.StopAllCoroutines()
MonoBehaviour.print(object)
MonoBehaviour.useGUILayout
MonoBehaviour.runInEditMode
Behaviour.enabled
Behaviour.isActiveAndEnabled
Component.GetComponent(Type)
Component.GetComponent<T>()
Component.TryGetComponent(Type, out Component)
Component.TryGetComponent<T>(out T)
Component.GetComponent(string)
Component.GetComponentInChildren(Type, bool)
Component.GetComponentInChildren(Type)
Component.GetComponentInChildren<T>(bool)
Component.GetComponentInChildren<T>()
Component.GetComponentsInChildren(Type, bool)

Component.GetComponentsInChildren(Type)
Component.GetComponentsInChildren<T>(bool)
Component.GetComponentsInChildren<T>(bool, List<T>)
Component.GetComponentsInChildren<T>()
Component.GetComponentsInChildren<T>(List<T>)
Component.GetComponentInParent(Type)
Component.GetComponentInParent<T>()
Component.GetComponentsInParent(Type, bool)
Component.GetComponentsInParent(Type)
Component.GetComponentsInParent<T>(bool)
Component.GetComponentsInParent<T>(bool, List<T>)
Component.GetComponentsInParent<T>()
Component.GetComponents(Type)
Component.GetComponents(Type, List<Component>)
Component.GetComponents<T>(List<T>)
Component.GetComponents<T>()
Component.CompareTag(string)
Component.SendMessageUpwards(string, object, SendMessageOptions)
Component.SendMessageUpwards(string, object)
Component.SendMessageUpwards(string)
Component.SendMessageUpwards(string, SendMessageOptions)
Component.SendMessage(string, object)
Component.SendMessage(string)
Component.SendMessage(string, object, SendMessageOptions)
Component.SendMessage(string, SendMessageOptions)
Component.BroadcastMessage(string, object, SendMessageOptions)
Component.BroadcastMessage(string, object)
Component.BroadcastMessage(string)
Component.BroadcastMessage(string, SendMessageOptions)
Component.transform
Component.gameObject
Component.tag
Object.GetInstanceID()
Object.GetHashCode()
Object.Equals(object)
Object.Instantiate(Object, Vector3, Quaternion)
Object.Instantiate(Object, Vector3, Quaternion, Transform)
Object.Instantiate(Object)
Object.Instantiate(Object, Transform)
Object.Instantiate(Object, Transform, bool)
Object.Instantiate<T>(T)
Object.Instantiate<T>(T, Vector3, Quaternion)
Object.Instantiate<T>(T, Vector3, Quaternion, Transform)
Object.Instantiate<T>(T, Transform)
Object.Instantiate<T>(T, Transform, bool)
Object.Destroy(Object, float)
Object.Destroy(Object)
Object.DestroyImmediate(Object, bool)
Object.DestroyImmediate(Object)
Object.FindObjectsOfType(Type)
Object.DontDestroyOnLoad(Object)
Object.DestroyObject(Object, float)

Object.DestroyObject(Object)
Object.FindSceneObjectsOfType(Type)
Object.FindObjectsOfTypeIncludingAssets(Type)
Object.FindObjectsOfType<T>()
Object.FindObjectOfType<T>()
Object.FindObjectsOfTypeAll(Type)
Object.FindObjectOfType(Type)
Object.ToString()
Object.name
Object.hideFlags
object.Equals(object, object)
object.GetType()
object.MemberwiseClone()
object.ReferenceEquals(object, object)

Namespace: LeTai.Asset.TranslucentImage

Assembly: LeTai.TranslucentImage.dll

Syntax

```
[HelpURL("https://leloctai.com/asset/translucentimage/docs/articles/customize.html#translucent-image")]
public class TranslucentImage : Image, ICanvasElement, IClippable, IMaskable, IMaterialModifier, ISerializationCallbackReceiver, ILayoutElement, ICanvasRaycastFilter, IMeshModifier
```

Fields

brightness

Brighten/darken them image

Declaration

```
[Tooltip("Brighten/darken them image")]
[Range(-1, 1)]
public float brightness
```

Field Value

TYPE	DESCRIPTION
float	

flatten

Flatten the color behind to help keep contrast on varying background

Declaration

```
[Tooltip("Flatten the color behind to help keep contrast on varying background")]
[Range(0, 1)]
public float flatten
```

Field Value

TYPE	DESCRIPTION
float	

m_spriteBlending

Declaration

```
[Tooltip("Blend between the sprite and background blur")]
[Range(0, 1)]
[FormerlySerializedAs("spriteBlending")]
public float m_spriteBlending
```

Field Value

TYPE	DESCRIPTION
float	

source

Source of blur for this image

Declaration

```
public TranslucentImageSource source
```

Field Value

TYPE	DESCRIPTION
TranslucentImageSource	

vibrancy

(De)Saturate them image, 1 is normal, 0 is grey scale, below zero make the image negative

Declaration

```
[Tooltip("(De)Saturate them image, 1 is normal, 0 is black and white, below zero make the image negative")]
[Range(-1, 3)]
public float vibrancy
```

Field Value

TYPE	DESCRIPTION
float	

Properties

spriteBlending

Declaration

```
public float spriteBlending { get; set; }
```

Property Value

TYPE	DESCRIPTION
float	

Methods

ModifyMesh(Mesh)

Declaration

```
public virtual void ModifyMesh(Mesh mesh)
```

Parameters

TYPE	NAME	DESCRIPTION
Mesh	mesh	

ModifyMesh(VertexHelper)

Call used to modify mesh. Place any custom mesh processing in this function.

Declaration

```
public virtual void ModifyMesh(VertexHelper vh)
```

Parameters

TYPE	NAME	DESCRIPTION
VertexHelper	vh	

OnDidApplyAnimationProperties()

Declaration

```
protected override void OnDidApplyAnimationProperties()
```

Overrides

UnityEngine.UI.Image.OnDidApplyAnimationProperties()

OnDisable()

Clear references.

Declaration

```
protected override void OnDisable()
```

Overrides

UnityEngine.UI.Image.OnDisable()

OnEnable()

Mark the Graphic and the canvas as having been changed.

Declaration

```
protected override void OnEnable()
```

Overrides

UnityEngine.UI.Image.OnEnable()

Start()

Declaration

```
protected override void Start()
```

Overrides

UnityEngine.EventSystems.UIBehaviour.Start()

Implements

UnityEngine.UI.ICanvasElement
UnityEngine.UI.IClippable
UnityEngine.UI.IMaskable
UnityEngine.UI.IMaterialModifier
UnityEngine.ISerializationCallbackReceiver
UnityEngine.UI.ILayoutElement
UnityEngine.ICanvasRaycastFilter
UnityEngine.UI.IMeshModifier

Class TranslucentImageSource

Common source of blur for Translucent Images.

Inheritance

object
Object
Component
Behaviour
MonoBehaviour
TranslucentImageSource

Inherited Members

MonoBehaviour.IsInvoking()
MonoBehaviour.CancelInvoke()
MonoBehaviour.Invoke(string, float)
MonoBehaviour.InvokeRepeating(string, float, float)
MonoBehaviour.CancelInvoke(string)
MonoBehaviour.IsInvoking(string)
MonoBehaviour.StartCoroutine(string)
MonoBehaviour.StartCoroutine(string, object)
MonoBehaviour.StartCoroutine(IEnumerator)
MonoBehaviour.StartCoroutine_Auto(IEnumerator)
MonoBehaviour.StopCoroutine(IEnumerator)
MonoBehaviour.StopCoroutine(Coroutine)
MonoBehaviour.StopCoroutine(string)
MonoBehaviour.StopAllCoroutines()
MonoBehaviour.print(object)
MonoBehaviour.useGUILayout
MonoBehaviour.runInEditMode
Behaviour.enabled
Behaviour.isActiveAndEnabled
Component.GetComponent(Type)
Component.GetComponent<T>()
Component.TryGetComponent(Type, out Component)
Component.TryGetComponent<T>(out T)
Component.GetComponent(string)
Component.GetComponentInChildren(Type, bool)
Component.GetComponentInChildren(Type)
Component.GetComponentInChildren<T>(bool)
Component.GetComponentInChildren<T>()
Component.GetComponentsInChildren(Type, bool)
Component.GetComponentsInChildren(Type)
Component.GetComponentsInChildren<T>(bool)
Component.GetComponentsInChildren<T>(bool, List<T>)
Component.GetComponentsInChildren<T>()
Component.GetComponentsInChildren<T>(List<T>)
Component.GetComponentInParent(Type)
Component.GetComponentInParent<T>()
Component.GetComponentsInParent(Type, bool)
Component.GetComponentsInParent(Type)
Component.GetComponentsInParent<T>(bool)
Component.GetComponentsInParent<T>(bool, List<T>)

Component.GetComponentsInParent<T>()
Component.GetComponents(Type)
Component.GetComponents(Type, List<Component>)
Component.GetComponents<T>(List<T>)
Component.GetComponents<T>()
Component.CompareTag(string)
Component.SendMessageUpwards(string, object, SendMessageOptions)
Component.SendMessageUpwards(string, object)
Component.SendMessageUpwards(string)
Component.SendMessageUpwards(string, SendMessageOptions)
Component.SendMessage(string, object)
Component.SendMessage(string)
Component.SendMessage(string, object, SendMessageOptions)
Component.SendMessage(string, SendMessageOptions)
Component.BroadcastMessage(string, object, SendMessageOptions)
Component.BroadcastMessage(string, object)
Component.BroadcastMessage(string)
Component.BroadcastMessage(string, SendMessageOptions)
Component.transform
Component.gameObject
Component.tag
Object.GetInstanceID()
Object.GetHashCode()
Object.Equals(object)
Object.Instantiate(Object, Vector3, Quaternion)
Object.Instantiate(Object, Vector3, Quaternion, Transform)
Object.Instantiate(Object)
Object.Instantiate(Object, Transform)
Object.Instantiate(Object, Transform, bool)
Object.Instantiate<T>(T)
Object.Instantiate<T>(T, Vector3, Quaternion)
Object.Instantiate<T>(T, Vector3, Quaternion, Transform)
Object.Instantiate<T>(T, Transform)
Object.Instantiate<T>(T, Transform, bool)
Object.Destroy(Object, float)
Object.Destroy(Object)
Object.DestroyImmediate(Object, bool)
Object.DestroyImmediate(Object)
Object.FindObjectsOfType(Type)
Object.DontDestroyOnLoad(Object)
Object.DestroyObject(Object, float)
Object.DestroyObject(Object)
Object.FindSceneObjectsOfType(Type)
Object.FindObjectsOfTypeIncludingAssets(Type)
Object.FindObjectsOfType<T>()
Object.FindObjectOfType<T>()
Object.FindObjectsOfTypeAll(Type)
Object.FindObjectOfType(Type)
Object.ToString()
Object.name
Object.hideFlags
object.Equals(object, object)

object.GetType()
object.MemberwiseClone()
object.ReferenceEquals(object, object)

Namespace: LeTai.Asset.TranslucentImage

Assembly: LeTai.TranslucentImage.dll

Syntax

```
[ExecuteAlways]
[RequireComponent(typeof(Camera))]
[AddComponentMenu("Image Effects/Tai Le Assets/Translucent Image Source")]
[HelpURL("https://leloctai.com/asset/translucentimage/docs/articles/customize.html#translucent-image-source")]
public class TranslucentImageSource : MonoBehaviour
```

Remarks

It is an Image effect that blur the render target of the Camera it attached to, then save the result to a global read-only Render Texture

Properties

BackgroundFill

Declaration

```
public BackgroundFill BackgroundFill { get; set; }
```

Property Value

TYPE	DESCRIPTION
BackgroundFill	

BlurAlgorithmSelection

Declaration

```
public BlurAlgorithmType BlurAlgorithmSelection { get; set; }
```

Property Value

TYPE	DESCRIPTION
BlurAlgorithmType	

BlurConfig

Declaration

```
public BlurConfig BlurConfig { get; set; }
```

Property Value

TYPE	DESCRIPTION
BlurConfig	

BlurRegion

Define the rectangular area on screen that will be blurred.

Declaration

```
public Rect BlurRegion { get; set; }
```

Property Value

TYPE	DESCRIPTION
Rect	Between 0 and 1

BlurRegionNormalizedScreenSpace

Blur Region rect is relative to Cam.rect . This is relative to the full screen

Declaration

```
public Rect BlurRegionNormalizedScreenSpace { get; set; }
```

Property Value

TYPE	DESCRIPTION
Rect	

BlurredScreen

Result of the image effect. Translucent Image use this as their content (read-only)

Declaration

```
public RenderTexture BlurredScreen { get; set; }
```

Property Value

TYPE	DESCRIPTION
RenderTexture	

CamRectOverride

Set in SRP to provide Cam.rect for overlay cameras

Declaration

```
public Rect CamRectOverride { get; set; }
```

Property Value

TYPE	DESCRIPTION
Rect	

Downsample

The rendered image will be shrinked by a factor of 2^{{this}} before bluring to reduce processing time

Declaration

```
public int Downsample { get; set; }
```

Property Value

TYPE	DESCRIPTION
int	Must be non-negative. Default to 0

MaxUpdateRate

Maximum number of times to update the blurred image each second

Declaration

```
public float MaxUpdateRate { get; set; }
```

Property Value

TYPE	DESCRIPTION
float	

Preview

Render the blurred result to the render target

Declaration

```
public bool Preview { get; set; }
```

Property Value

TYPE	DESCRIPTION
bool	

Methods

CreateNewBlurredScreen(Vector2Int)

Declaration

```
protected virtual void CreateNewBlurredScreen(Vector2Int camPixelSize)
```

Parameters

TYPE	NAME	DESCRIPTION
Vector2Int	camPixelSize	

OnBeforeBlur(Vector2Int)

Declaration

```
public void OnBeforeBlur(Vector2Int camPixelSize)
```

Parameters

TYPE	NAME	DESCRIPTION
Vector2Int	camPixelSize	

Request(bool)

Declaration

```
public void Request(bool isForOverlayCanvas)
```

Parameters

TYPE	NAME	DESCRIPTION
bool	isForOverlayCanvas	

Start()

Declaration

```
protected virtual void Start()
```

shouldUpdateBlur()

Declaration

```
public bool shouldUpdateBlur()
```

Returns

TYPE	DESCRIPTION
bool	

Namespace LeTai.Asset.TranslucentImage.UniversalRP

Classes

TranslucentImageBlurRenderPass

TranslucentImageBlurSource

Enums

RenderOrder

Enum RenderOrder

Namespace: LeTai.Asset.TranslucentImage.UniversalRP

Assembly: LeTai.TranslucentImage.UniversalRP.dll

Syntax

```
public enum RenderOrder
```

Fields

NAME	DESCRIPTION
AfterPostProcessing	
BeforePostProcessing	

Class TranslucentImageBlurRenderPass

Inheritance

object

ScriptableRenderPass

TranslucentImageBlurRenderPass

Inherited Members

ScriptableRenderPass.ConfigureTarget(RenderTargetIdentifier, RenderTargetIdentifier)
ScriptableRenderPass.ConfigureTarget(RenderTargetIdentifier[], RenderTargetIdentifier)
ScriptableRenderPass.ConfigureTarget(RenderTargetIdentifier)
ScriptableRenderPass.ConfigureTarget(RenderTargetIdentifier[])
ScriptableRenderPass.ConfigureClear(ClearFlag, Color)
ScriptableRenderPass.Configure(CommandBuffer, RenderTextureDescriptor)
ScriptableRenderPass.FrameCleanup(CommandBuffer)
ScriptableRenderPass.Blit(CommandBuffer, RenderTargetIdentifier, RenderTargetIdentifier, Material, int)
ScriptableRenderPass.RenderPostProcessing(CommandBuffer, ref CameraData, RenderTextureDescriptor, RenderTargetIdentifier, RenderTargetIdentifier, bool, bool)
ScriptableRenderPass.CreateDrawingSettings(ShaderTagId, ref RenderingData, SortingCriteria)
ScriptableRenderPass.CreateDrawingSettings(List<ShaderTagId>, ref RenderingData, SortingCriteria)
ScriptableRenderPass.renderPassEvent
ScriptableRenderPass.colorAttachments
ScriptableRenderPass.colorAttachment
ScriptableRenderPass.depthAttachment
ScriptableRenderPass.clearFlag
ScriptableRenderPass.clearColor
object.Equals(object)
object.Equals(object, object)
object.GetHashCode()
object.GetType()
object.MemberwiseClone()
object.ReferenceEquals(object, object)
object.ToString()

Namespace: LeTai.Asset.TranslucentImage.UniversalRP

Assembly: LeTai.TranslucentImage.UniversalRP.dll

Syntax

```
[MovedFrom("LeTai.Asset.TranslucentImage.LWRP")]
public class TranslucentImageBlurRenderPass : ScriptableRenderPass
```

Properties

PreviewMaterial

Declaration

```
public Material PreviewMaterial { get; }
```

Property Value

TYPE	DESCRIPTION
Material	

Methods

Execute(ScriptableRenderContext, ref RenderingData)

Declaration

```
public override void Execute(ScriptableRenderContext context, ref RenderingData renderingData)
```

Parameters

TYPE	NAME	DESCRIPTION
ScriptableRenderContext	context	
RenderingData	renderingData	

Overrides

UnityEngine.Rendering.Universal.ScriptableRenderPass.Execute(UnityEngine.Rendering.ScriptableRenderContext, ref UnityEngine.Rendering.Universal.RenderingData)

~TranslucentImageBlurRenderPass()

Declaration

```
protected ~TranslucentImageBlurRenderPass()
```

Class TranslucentImageBlurSource

Inheritance

object

Object

ScriptableObject

ScriptableRendererFeature

TranslucentImageBlurSource

Inherited Members

ScriptableRendererFeature.SetActive(bool)

ScriptableRendererFeature.isActive

ScriptableObject.SetDirty()

ScriptableObject.CreateInstance(string)

ScriptableObject.CreateInstance(Type)

ScriptableObject.CreateInstance<T>()

Object.GetInstanceID()

Object.GetHashCode()

Object.Equals(object)

Object.Instantiate(Object, Vector3, Quaternion)

Object.Instantiate(Object, Vector3, Quaternion, Transform)

Object.Instantiate(Object)

Object.Instantiate(Object, Transform)

Object.Instantiate(Object, Transform, bool)

Object.Instantiate<T>(T)

Object.Instantiate<T>(T, Vector3, Quaternion)

Object.Instantiate<T>(T, Vector3, Quaternion, Transform)

Object.Instantiate<T>(T, Transform)

Object.Instantiate<T>(T, Transform, bool)

Object.Destroy(Object, float)

Object.Destroy(Object)

Object.DestroyImmediate(Object, bool)

Object.DestroyImmediate(Object)

Object.FindObjectsOfType(Type)

Object.DontDestroyOnLoad(Object)

Object.DestroyObject(Object, float)

Object.DestroyObject(Object)

Object.FindSceneObjectsOfType(Type)

Object.FindObjectsOfTypeIncludingAssets(Type)

Object.FindObjectsOfType<T>()

Object.FindObjectOfType<T>()

Object.FindObjectsOfTypeAll(Type)

Object.FindObjectOfType(Type)

Object.ToString()

Object.name

Object.hideFlags

object.Equals(object, object)

object.GetType()

object.MemberwiseClone()

object.ReferenceEquals(object, object)

Namespace: LeTai.Asset.TranslucentImage.UniversalRP

Assembly: LeTai.TranslucentImage.UniversalRP.dll

Syntax

```
[MovedFrom("LeTai.Asset.TranslucentImage.LWRP")]
public class TranslucentImageBlurSource : ScriptableRendererFeature
```

Fields

canvasDisappearWorkaround

Declaration

```
public bool canvasDisappearWorkaround
```

Field Value

TYPE	DESCRIPTION
bool	

Methods

AddRenderPasses(ScriptableRenderer, ref RenderingData)

Declaration

```
public override void AddRenderPasses(ScriptableRenderer renderer, ref RenderingData renderingData)
```

Parameters

TYPE	NAME	DESCRIPTION
ScriptableRenderer	renderer	
RenderingData	renderingData	

Overrides

UnityEngine.Rendering.Universal.ScriptableRendererFeature.AddRenderPasses(UnityEngine.Rendering.Universal.ScriptableRenderer, ref UnityEngine.Rendering.Universal.RenderingData)

Create()

Declaration

```
public override void Create()
```

Overrides

UnityEngine.Rendering.Universal.ScriptableRendererFeature.Create()

GetPixelSize(CameraData)

Declaration

```
public Rect GetPixelSize(CameraData cameraData)
```

Parameters

TYPE	NAME	DESCRIPTION
CameraData	cameraData	

Returns

TYPE	DESCRIPTION
Rect	

RegisterSource(TranslucentImageSource)

When adding new Translucent Image Source to existing Camera at run time, the new Source must be registered here

Declaration

```
public void RegisterSource(TranslucentImageSource source)
```

Parameters

TYPE	NAME	DESCRIPTION
TranslucentImageSource	source	