1. Overview

A quick glance over the features of this asset.

1.1 FXLab

The FXLab emulates the Unity Pro feature to render into textures and reuse these for special effects.

This asset delivers everything you need to create and use these textures. Additionally this assets provides many ready to use shaders and post processing effects.
Some of these effects are covered by this documentation, so please have a look.

The drawbacks of this asset compared to the native Unity Pro ability are that the performance will be more decreased. But the FXLab will take care of you and will try to ensure a very low impact on it. By using advanced algorithms which improves the overall speed of the basics used to generate the render textures.

1.2 Water

The water effect provides multiple shaders for every situation in which water may plays the primary role.

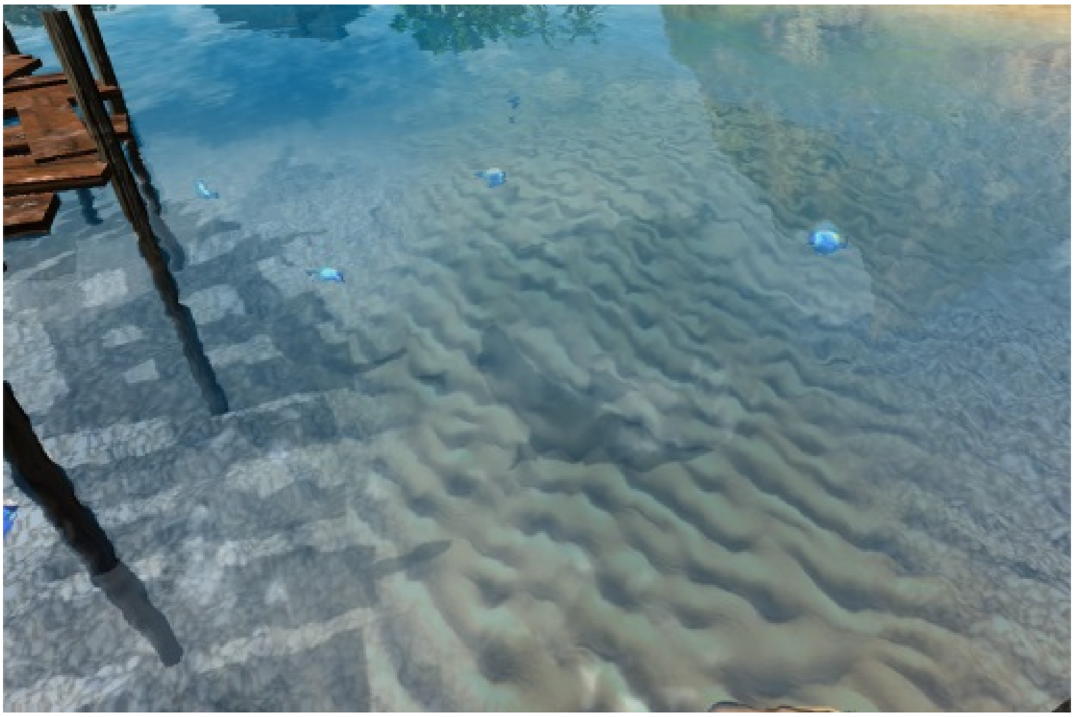
The following options are provided and may be combined at will.

Realtime Reflection:

Specify the reflection plane (water surface) and see your world get reflected on it. It will react to moving objects or changing environment variables and display these.

Realtime Refraction:

The core functionality of the FXLab Asset provides the ability to distort everything below the water



surface. The strength of the distortion is variable as everything else. Multiple fresnel settings allow a fine control between the visibility of the refraction and reflection color.

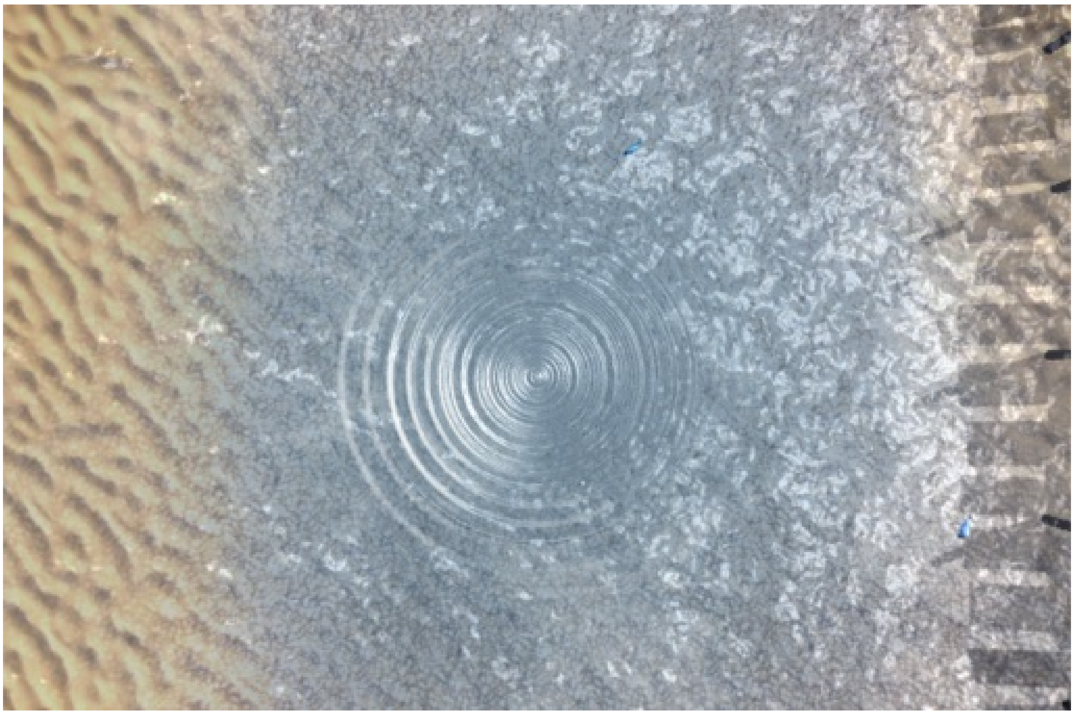
Color Extinction:

You can use the viewdepth and the height to simulate the color extinction of water. The further you look into the water the more blueish the water color will be. The colors are fully controllable by you.

Color Dispersion:

All translucent surfaces bend the light in different ways, this is how rainbows are born. You can have this for the water too! Specify how every color channel will be

altered, it may be subtle but the impact is huge.



Decals (Splash and Waves):

Drop a stone into the water and watch the ripples grow and decease. There is nothing more to say about it, except that it will not look like a plain gradient.

1.3 Glass / Crystal

The glass effect provides multiple shaders for many glass types with focus on the refraction of light.

The following options are provided and may be combined at will.

Realtime Refraction:

The core functionality of the FXLab Asset

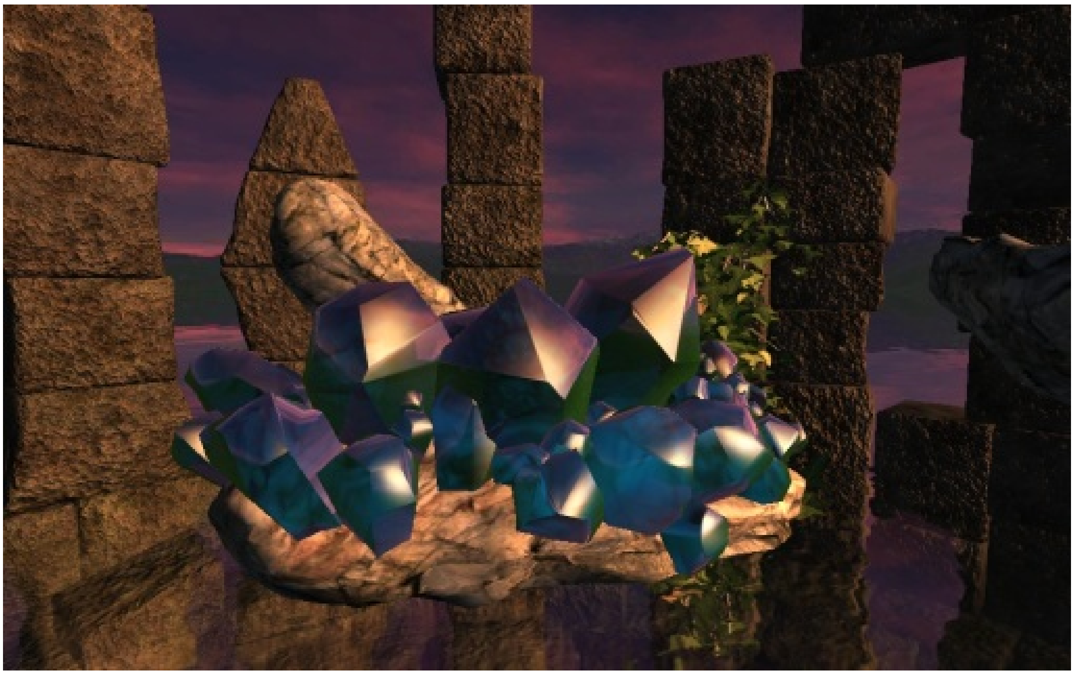
provides the ability to distort everything behind the glass surface. The strength of the distortion is variable as everything else. Multiple fresnel settings allows a fine control between the visibility of the refraction and reflection.

Color Dispersion:

All translucent surfaces bend the light in different ways, this is how rainbows are born. Specify how every color channel will be altered, it may be subtle but the impact is huge.

Blur:

You can simulate frosted and blurred glas. Just setup the blur strength and see the results. This effect is very performance intensive, so be carefull when to use it.



Multiple Colors and Choices:

You can specify a base texture (for example a bluish to purple crystal color). A color for the reflection, a color for the refraction and also a cubemap for the Reflection.

1.4 Shockwave

The shockwave effect is a great addition to every explosion. Its a very simple effect with a huge visual impact. Also it doesnt need a huge setup to use.

Shockwave:

A simple shockwave emitting effect.

1.5 Distortion

The distortion effect is basically the core of the FXLab as most described effects will use some kind of distortion. This category is meant to be for the most simple effect, a simple distortion without any color or reflection effects.

The following images show some example usages. More specific effects can easily be crafted.

HeatHaze:

Add the distortion shader onto a material designed for particles and create the effect of rising heat.

Sucktion:

Create the illusion of an eat all day and night black hole by

assigning the distortion shader onto a plane. Dont setting a FXObject layer will ensure a nice feedback effect.

Motion Trail:

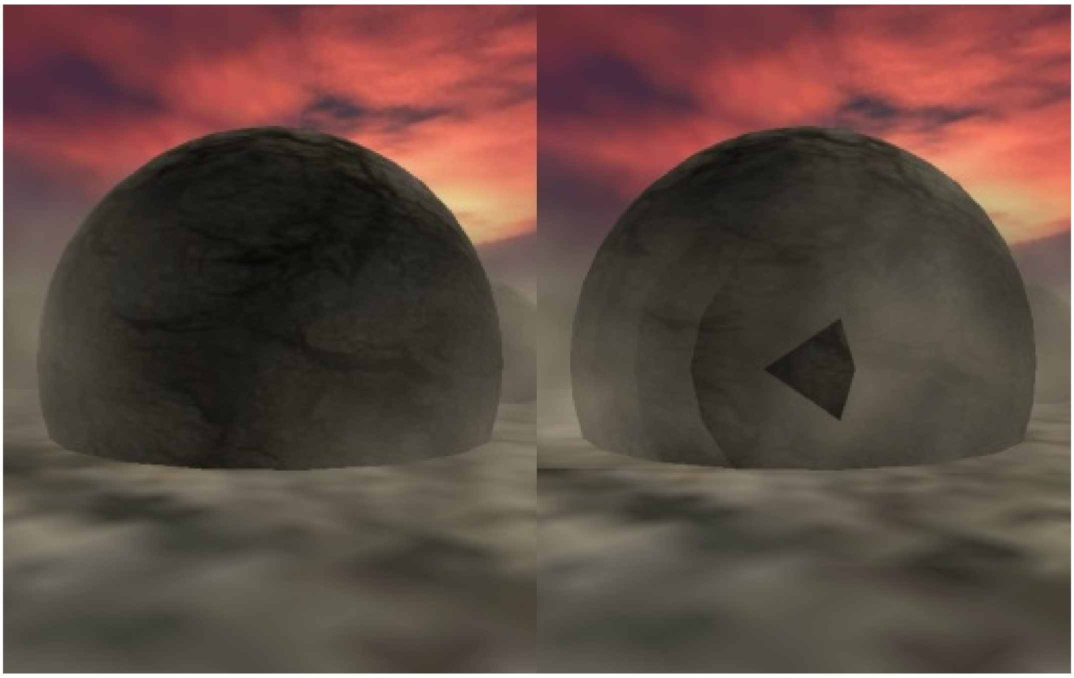
The distortion shader is usable with Trail and LineRenderers, which allows the usage as motion trails, for sword swings and more.

1.6 Soft Particles

Soft Particles eliminate the problem with visibly overlapping areas when particles appear to stuck inside models.

Soft Particles:

Left: Soft Particles
Right: Normal Particles



1.7 Post Processing

A component to render post processing effects is available inside this asset. Many will use the render textures generated by FXLab, but some will work without those.

Following effects are provided and covered by many sample scenes:

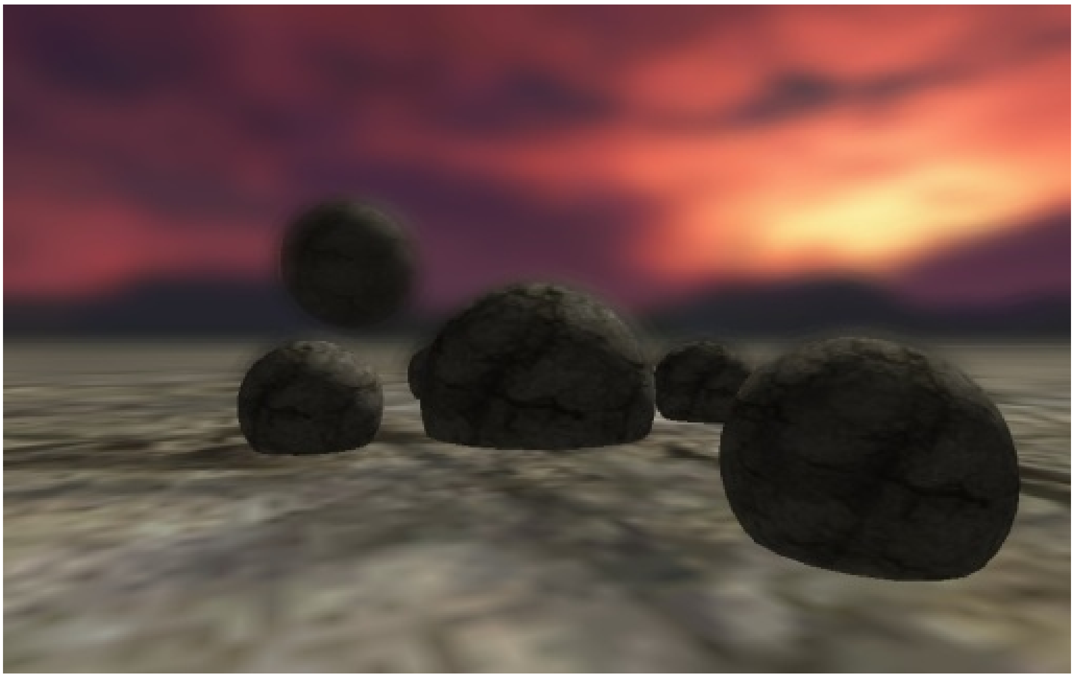
- Depth of Field
- Bloom
- Luminance (Grayscale)
- Pixelize
- Vignette
- TV Noise
- Interlace Lines
- Motion Blur
- Screen Space Antialiasing (SSAA)
- Screen Space Light Shafts

More specific effects can easily be crafted.

The following images show some example post processing effects.

Depth Of Field:

A depth of field effect with variable blur range, focal distance and falloff range.



Under Water:

A simple underwater effect, which distorts the view and blurs the image based upon the view distance.

2. Setup

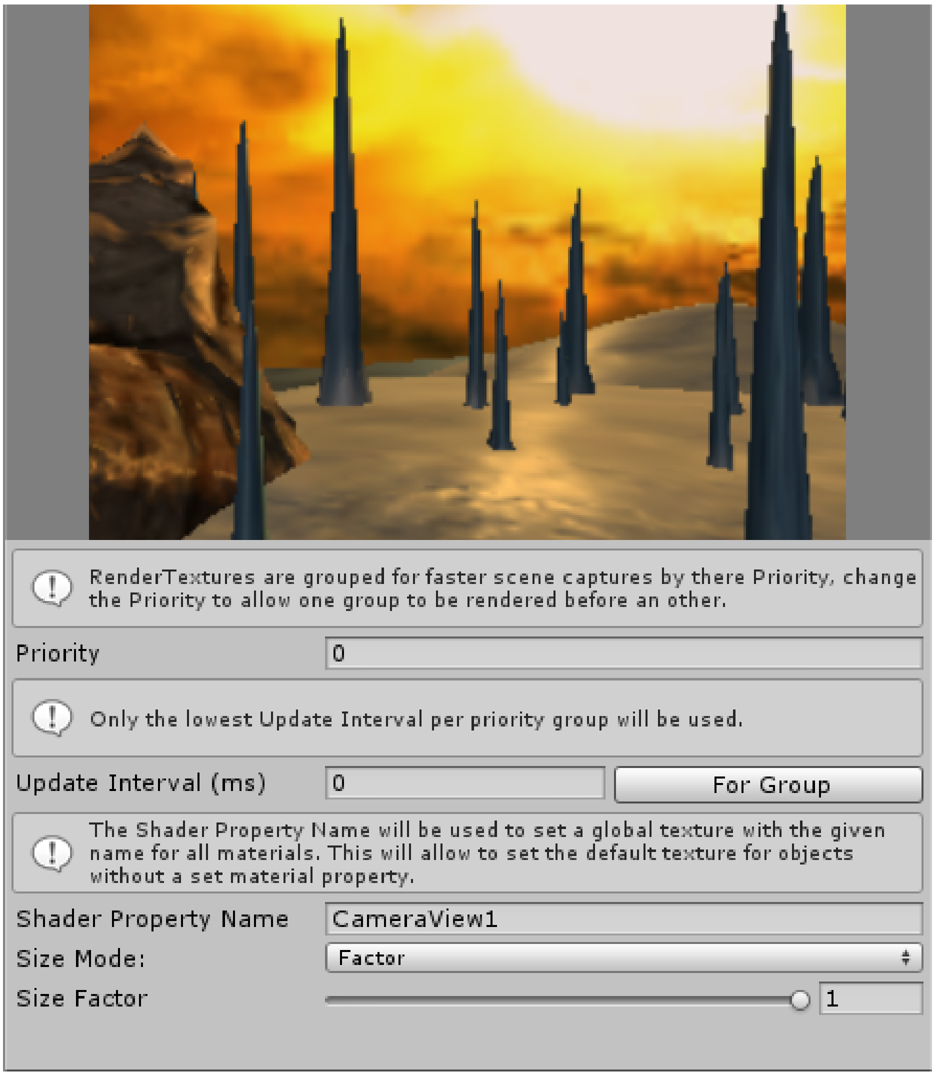
How to activate the FXLab effects.

2.1 FX Textures

The camera needs to know what it should render, for this we provide two possibilities for you, the first one is the FX Texture.

We provide 5 different FX Texture types, each with an own purpose. As different effects need different sets of textures you can selectively add those FX Texture Components to your Camera GameObject.
Klick on "Add Component" and navigate to FXLab -> Textures, then select those you need.

All FX Texture components provide a "Target Texture", this texture is the render texture. You can assign an own rendertexture to this field, when nothing is provided an automatically generated rendertexture will be used. Expand the details to look into the used texture and change the options by your needs. You can also see a preview of the generated texture:



RenderTextures are grouped for faster scene captures by there Priority, change the Priority to allow one group to be rendered before an other.

Priority: 0

Only the lowest Update Interval per priority group will be used.

Update Interval (ms): 0

The Shader Property Name will be used to set a global texture with the given name for all materials. This will allow to set the default texture for objects without a set material property.

Shader Property Name: CameraView1

Size Mode: Factor

Size Factor: 1

RenderTexture:

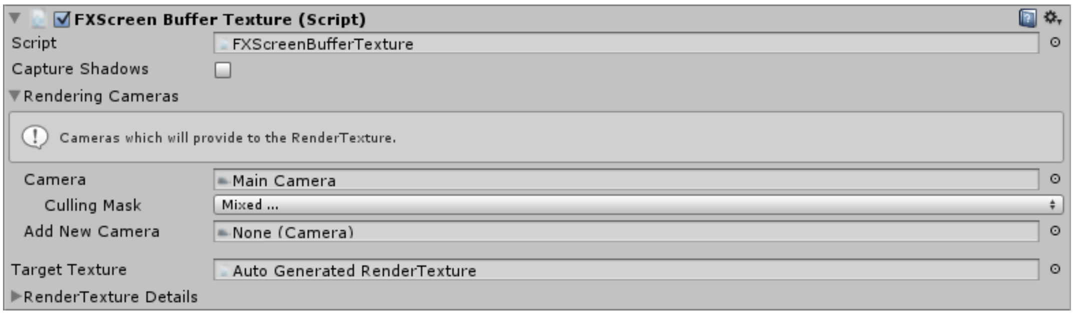
RenderTextures are assets which can be manually created by the user or automatically provided by the FXCaptureScreen or FXTexture components.

Automatically generated RenderTextures can not be used for manual assignments and will automatically assigned to all texture slots with the given Shader Property Name.

Automatically generated RenderTextures can be converted to manually generated by clicking the provided export button.

You can also add multiple cameras to render the target texture by adding them into the "Cameras" slot. Selecting a "CullingMask" for one camera will create an intersection between the cameras CullingMask and the selected CullingMask at the FXTexture.

These are the from FXLab provided texture generator types:



▼ ☑ FXScreen Buffer Texture (Script)

Script: FXScreenBufferTexture

Capture Shadows: □

▼Rendering Cameras

Cameras which will provide to the RenderTexture.

Camera: Main Camera

Culling Mask: Mixed ...

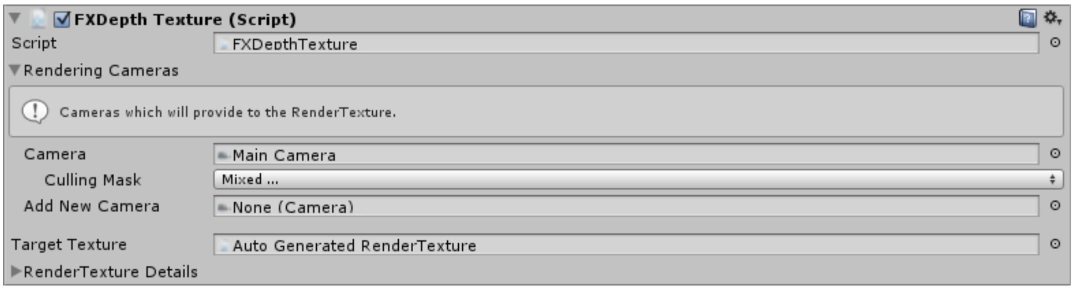
Add New Camera: None (Camera)

Target Texture: Auto Generated RenderTexture

▶RenderTexture Details

Screen Buffer:

This texture holds the whole visible content of the scene. This is a very basic texture type as most effects will make use of this texture.



☑ FXDepth Texture (Script)

Script: FXDepthTexture

▼ Rendering Cameras

Cameras which will provide to the RenderTexture.

Camera: Main Camera

Culling Mask: Mixed ...

Add New Camera: None (Camera)

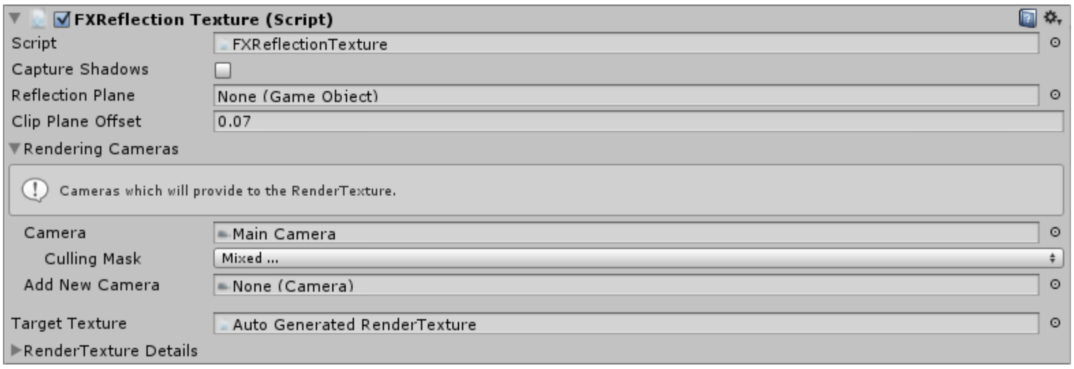
Target Texture: Auto Generated RenderTexture

▶ RenderTexture Details

Depth:

This texture stores the depth of the visible content of the scene to the camera. This is also a very basic effects.

The depth texture works with most objects (Opaque, TransparentCutout, Vegetation, eg.).



☑ FXReflection Texture (Script)

Script: FXReflectionTexture

Capture Shadows: ☐

Reflection Plane: None (Game Object)

Clip Plane Offset: 0.07

▼ Rendering Cameras

Cameras which will provide to the RenderTexture.

Camera: Main Camera

Culling Mask: Mixed ...

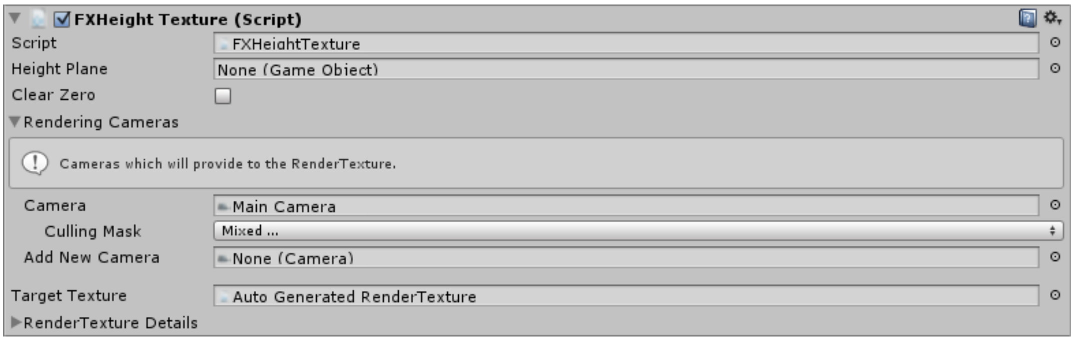
Add New Camera: None (Camera)

Target Texture: Auto Generated RenderTexture

▶ RenderTexture Details

Reflection:

This texture stores the reflected scene based upon your camera and a reflection plane.



☑ FXHeight Texture (Script)

Script: FXHeightTexture

Height Plane: None (Game Object)

Clear Zero: ☐

▼ Rendering Cameras

Cameras which will provide to the RenderTexture.

Camera: Main Camera

Culling Mask: Mixed ...

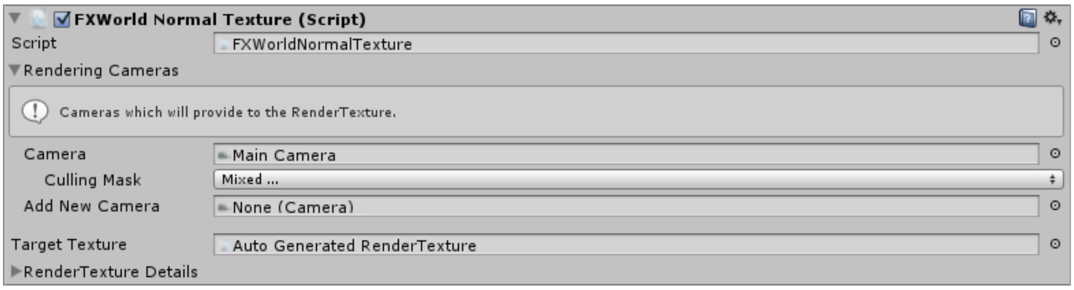
Add New Camera: None (Camera)

Target Texture: Auto Generated RenderTexture

▶ RenderTexture Details

Height:

This texture stores the height to the given height plane from everything below the height plane. You can use this texture for example to create height fog.



☑ FXWorld Normal Texture (Script)

Script: FXWorldNormalTexture

▼ Rendering Cameras

Cameras which will provide to the RenderTexture.

Camera: Main Camera

Culling Mask: Mixed ...

Add New Camera: None (Camera)

Target Texture: Auto Generated RenderTexture

▶ RenderTexture Details

WorldNormals:

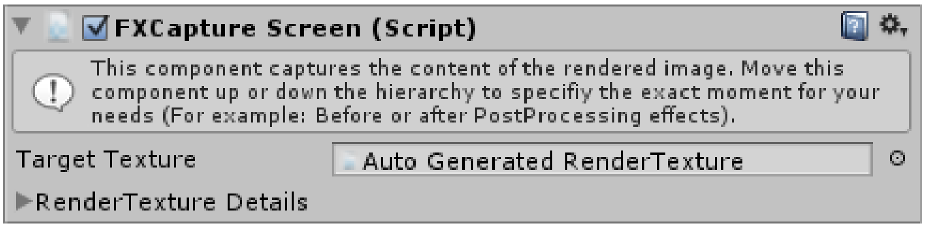
This texture stores the world space normals of the visible content.

2.2 Screen Capturing

The second way to get the content for a render texture is the usage of the FX Capture Screen component.

As the name implies, this component will capture the currently visible screen into a texture. This allows you to create a full resolution copy of your screen which then can be used as input for many post processing effects. Placing it before or after post processing effects may create different results.

This component will have a very high impact onto the performance, so you should use this with care and when possible use FX Texture components instead (FX Textures are better as they will internally share some data and are "batchable", whereas the render textures used by the FX Capture Screen component are not.)


☑ FXCapture Screen (Script)

This component captures the content of the rendered image. Move this component up or down the hierarchy to specifiy the exact moment for your needs (For example: Before or after PostProcessing effects).

Target Texture: Auto Generated RenderTexture

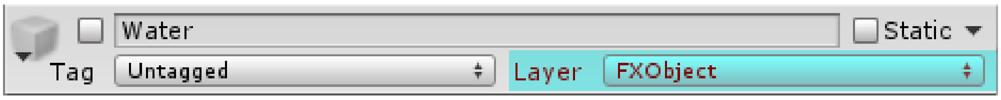
▶RenderTexture Details

Capture Screen:

This component will capture the content of the screen before any UI is applied.

2.3 Objects

The setup of your GameObjects is very simple.


□ Water

□ Static

Tag: Untagged

Layer: FXObject

Step 1:

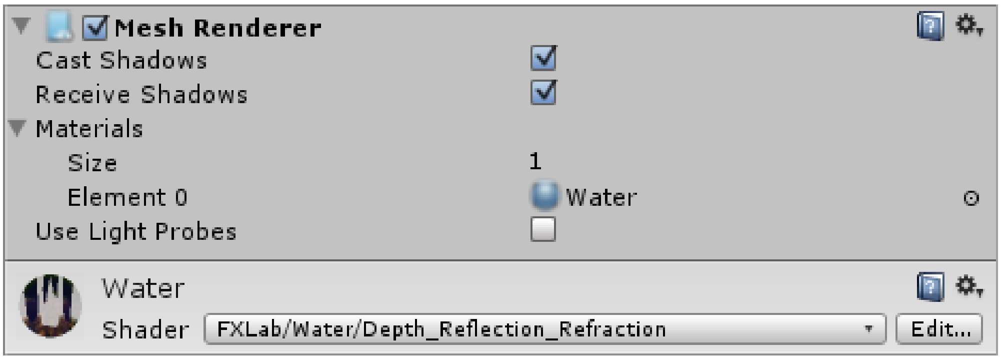
Change the layer of your GameObjects which should use the FXLab effects to a layer the FXCamera does not render. We used the "FXObject" layer for this, but you are free to choose your own.

This ensures that the FXCamera does not render an effect in the texture which uses this effect. Which may result in strange visual artifacts (you can use this to create feedback effects, this side effect will be used to create the black hole effect).

Step 2:

Add the material with an FXLab effect onto your GameObject (MeshRenderer). You can use the GameView (not the SceneView) while in edit mode to see the results.

FX Materials will use a custom material editor, which allows the assigning of Render Textures to special properties. You can leave this properties empty, in this case the last created RenderTexture in the scene with the same propertyname will be automatically used.



☑ Mesh Renderer	
Cast Shadows	☑
Receive Shadows	☑
Materials	
Size	1
Element 0	Water
Use Light Probes	□

Water

Shader: FXLab/Water/Depth_Reflection_Refraction

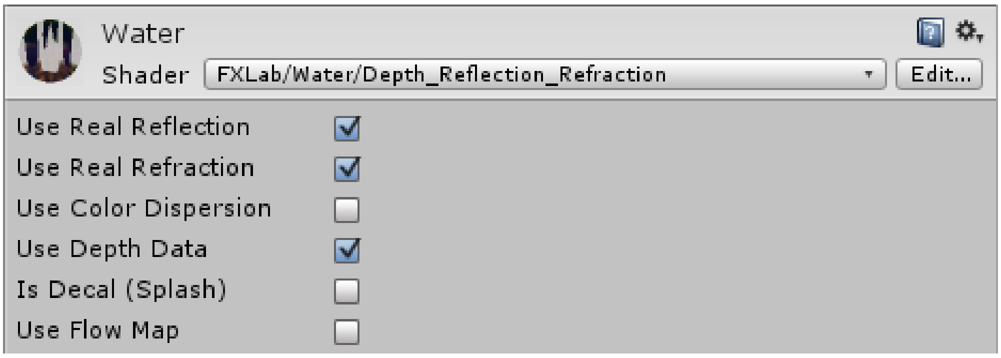
3. Water

The water effect in detail.

3.1 Water

Water is a very complex effect with an versatile amount of options, thats why there exists over 60 shaders only for water. But you need only one. The FXLab will ensure that you find the best suited shader without the need for you to traverse every single one.
For this the FXLab provides a custom material editor only for the water.
Simply create a material and add any water shader on it, you do not need to select the most fitting one, as this will be handled through the water material editor. Just ensure that you use a plane for the best results as target for your water material.

You can use the checkboxes to select the most fitting shader for your material.
Check or uncheck the options you think you will need. Certain options require different



Water

Shader FXLab/Water/Depth_Reflection_Refraction

Use Real Reflection ☑

Use Real Refraction ☑

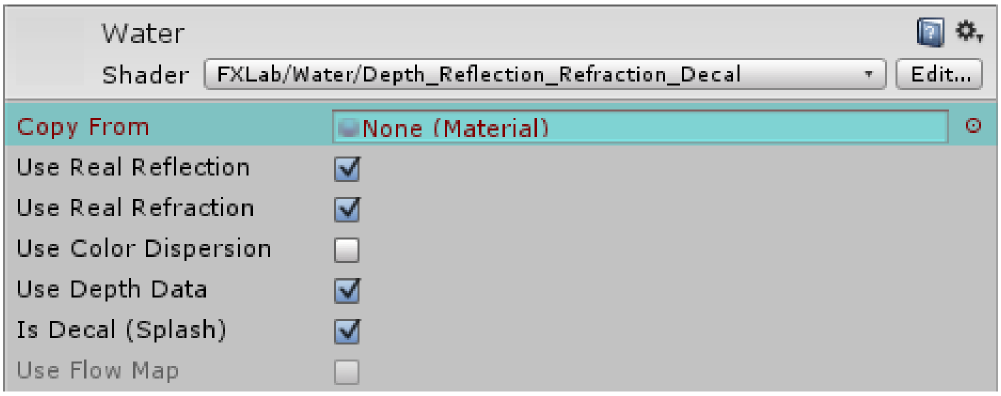
Use Color Dispersion □

Use Depth Data ☑

Is Decal (Splash) □

Use Flow Map □

FXCamera settings. The Reflection and Depth effect require that a water plane has been assigned to the FXCamera, as both effects rely on it.



Water

Shader FXLab/Water/Depth_Reflection_Refraction_Decal

Copy From None (Material)

Use Real Reflection ☑

Use Real Refraction ☑

Use Color Dispersion □

Use Depth Data ☑

Is Decal (Splash) ☑

Use Flow Map □

When using the water decal option a new field will appear in the editor, with this you can select any other water material you whish to copy the settings from. This makes it easier for you to define the best fitting decal material for you.

Water decals are used to generate good looking waves on your water, to display this you must create a plane exactly on your water and use this material for it. You can set the size of this plane as you need.

3.2 Settings

Following settings are available when using a water material:

Specular Color (_SpecColor)

The color of specular light reflections on the water surface.

Specular (_Specular)

The specular light reflection strength.

Shininess (_Shininess)

The specular light reflection exponent.

Bump Map (_BumpMap)

The Normalmap and Distortionmap used on the water.

Distortion Strength (_DistortionStrength)

How much should the Bump Map be used to change the reflection and refraction. (Ranges mostly from 0 to 100, where 100 means the full screen texture size will be used and 0 means no distortion.)

Fresnel Normal Strength (_FresnelNormalStrength)

How strong is the influence of the Bump Map on the Fresnel calculation.

Fresnel (_Fresnel)

How strong is the reflection based upon the view angle to the surface, compared to the refraction visibility.

Fresnel Factor (_FresnelFactor)

How much should the calculated fresnel term be multiplicated.

Fresnel Bias (_FresnelBias)

How much should the calculated fresnel term be offsetted.

Flow Map (_FlowMap)

The texture which should be used to define the flow of the water.

```
Only when Decal is deactivated
Only when FlowMap is activated
```

Flow X Speed (_FlowSpeedX)

The flow speed on the X axis.

```
Only when Decal is deactivated
Only when FlowMap is deactivated
```

Flow Y Speed (_FlowSpeedY)

The flow speed on the Y axis.

```
Only when Decal is deactivated
Only when FlowMap is deactivated
```

Wave Flow Factor (_WaveSpeed)

How fast should the flow speed of the water be.

```
Only when Decal is deactivated
```

Wave Scale (_WaveScale)

How much should every wave layer be scaled (There are 4 wave layers).

```
Only when Decal is deactivated
```

Wave Influence Factor (_WaveInfluenceFactor)

How much should every wave layer be visible (There are 4 wave layers).

```
Only when Decal is deactivated
```

MainTex (_MainTex)

The base water texture.

```
Only when Refraction is deactivated
```

Reflection Cube (_ReflectionCube)

The cubemap to use to simulate the reflection.

```
Only when Reflection is deactivated
```

View Depth Color (_ViewDepthColor)

The color the water should take when the maximum view depth into the water has been reached.

```
Only when Depth is activated
```

Depth Color (_DepthColor)

The color the water should take when the maximum depth has been reached.

```
Only when Depth is activated
```

Shore Hardness (_ShoreHardness)

How strong should the water be 100% visible as water depending on the depth.

```
Only when Depth is activated
```

Depth from Background to Reflection Height Factor (_HeightToReflectionPlaneFactor)

Which should the maximum depth be at which the water will take 100% of the Depth Color.

```
Only when Depth is activated
```

View Depth from Reflection to Background Plane Factor (_DepthToReflectionPlaneFactor)

Which should the maximum vide depth be at which the water will take 100% of the View Depth Color.

```
Only when Depth is activated
```

Red Dispersion Factor (_RedDispersionFactor)

How much should the light be bended for the red color channel.

```
Only when Dispersion is activated
```

Green Dispersion Factor (_GreenDispersionFactor)

How much should the light be bended for the green color channel.

```
Only when Dispersion is activated
```

Blue Dispersion Factor (_BlueDispersionFactor)

How much should the light be bended for the blue color channel.

Only when Dispersion is activated

Mask Map (_MaskMap)

The texture which Blue color channel will be used to change the transparency of the decal. (100% transparency will show the underlying water.)

Only when Decal is activated

Transparency (_Transparency)

Used to change the transparency of the decal. (100% transparency will show the underlying water.)

Only when Decal is activated

Bump Up influence (_BumpUpInfluence)

How much should the up vector of the Bump Map be used to determine the transparency of the decal. (100% transparency will show the underlying water.)

Only when Decal is activated

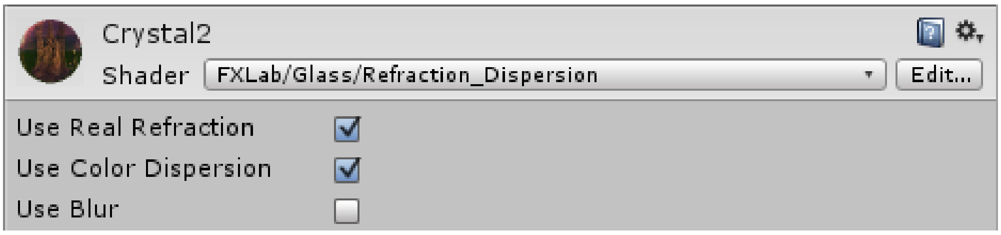
---

4. Glass

The glass effect in detail.

4.1 Glass

Glass is a very complex effect with an versatile amount of options.
For this the FXLab provides a custom material editor only for the glass.
Simply create a material and add any glass shader on it, you do not need to select the most fitting one, as this will be handled through the glass material editor.



Crystal2

Shader FXLab/Glass/Refraction_Dispersion

Use Real Refraction ☑

Use Color Dispersion ☑

Use Blur □

You can use the checkboxes to select the most fitting shader for your material.
Check or uncheck the options you think you will need.

4.2 Settings

Following settings are available when using a glass material:

Refraction Color (_RefractionColor)

The color in which the refraction will be tinted.

Reflection Color (_ReflectionColor)

The color in which the reflection will be tinted.

Specular Color (_SpecColor)

The color of specular light reflections on the glass surface.

Specular (_Specular)

The specular light reflection strength.

Shininess (_Shininess)

The specular light reflection exponent.

Bump Map (_BumpMap)

The Normalmap and Distortionmap used on the glass.

Bump Strength (_BumpStrength)

How much should the Bump Map be visible.

Distortion Strength (_DistortionStrength)

How much should the Bump Map be used to change the reflection and refraction. (Ranges mostly from 0 to 100, where 100 means the full screen texture size will be used and 0 means no distortion.)

Reflection Cube (_ReflectionCube)

The cubemap to use to simulate the reflection.

Fresnel Normal Strength (_FresnelNormalStrength)

How strong is the influence of the Bump Map on the Fresnel calculation.

Fresnel (_Fresnel)

How strong is the reflection based upon the view angle to the surface, compared to the refraction visibility.

Fresnel Factor (_FresnelFactor)

How much should the calculated fresnel term be multiplicated.

Fresnel Bias (_FresnelBias)

How much should the calculated fresnel term be offsetted.

Red Dispersion Factor (_RedDispersionFactor)

How much should the light be bended for the red color channel.

`Only when Dispersion is activated`

Green Dispersion Factor (_GreenDispersionFactor)

How much should the light be bended for the green color channel.

`Only when Dispersion is activated`

Blue Dispersion Factor (_BlueDispersionFactor)

How much should the light be bended for the blue color channel.

`Only when Dispersion is activated`

Blur Factor (_BlurFactor)

How much should the refraction be blurred.

`Only when Blur is activated`

Light Influence (_LightInfluence)

How much should the light be used on the reflection.

5. Shockwave

The shockwave effect in detail.

5.1 Shockwave

The shockwave is a very simple effect, which does not require any knowledge to setup.

Create a material and add the shockwave shader on it. Most properties dont require any setup to look great.

5.2 Settings

Following settings are available when using a shockwave material:

Transparency (_Transparency)

How translucent should the material be.

Distortion Map (_DistortionMap)

A texture which will be used to describe the strength and direction of the distortion.

Distortion Strength (_DistortionStrength)

A factor which describes the strength of the distortion (Ranges mostly from 0 to 100, where 100 means the full screen texture size will be used and 0 means no distortion.).

Distortion Strength (_DistortionStrength)

A factor which describes the strength of the distortion (Ranges mostly from 0 to 100, where 100 means the full screen texture size will be used and 0 means no distortion.).

Blur Factor (_BlurFactor)

How much should the shockwave blur the background on the shockwave edges.

Edge Hardness (_EdgeHardness)

How hard should the transition from the edges of the shockwaves appear, the higher this value is the stronger the distortion appears on the edges of a shockwave.

_EdgePower (_EdgePower)

Increases the speed of the edge distortion strength by this value.

6. Write Shaders

How to write custom effects.

6.1 Normal Shaders

Normal shaders are shaders which doesnt use the surface shader technology provided by Unity. Surface Shaders may use the same techniques described for the normal shaders.

We start with an sample of a normal shader which only outputs the given MainTex Texture.

```
Shader "FXLab/Distortion" {
    Properties {
                _MainTex ("MainTex", 2D) = "white" {}
        }
        SubShader {
                Blend SrcAlpha OneMinusSrcAlpha
                Tags { "Queue"="Transparent" "IgnoreProjector"="True" "RenderTyp
                Lighting Off
                Cull Off
                Fog { Mode Off }
                ZWrite Off

                Pass {
                        CGPROGRAM
                        #pragma vertex vert
                        #pragma fragment frag
                        #pragma target 3.0
                        #include "UnityCG.cginc"

                        sampler2D _MainTex;

                        struct appdata {
                                float4 vertex : POSITION;
                                float4 texcoord : TEXCOORD0;
                                float4 color : COLOR0;
                        };

                        struct v2f {
                                float4 pos : SV_POSITION;
                                float2 uv : TEXCOORD0;
                                float4 color : COLOR0;
                        };

                        v2f vert (appdata v) {
                                v2f o;
                                o.pos = mul( UNITY_MATRIX_MVP, v.vertex );
                                o.uv = float4( v.texcoord.xy, 0, 0 );
                                o.color = v.color;
                                return o;
                        }

                        fixed4 frag( v2f o ) : COLOR
                        {
                                return tex2D(_MainTex, o.uv);
                        }
                        ENDCG
                }
        }
    Fallback off
}
```

Now we need to include the FXLab.cginc which can be found inside the FXLab/Shaders directory, you need to specify the path relative to the location of your shader.

```
...
#include "UnityCG.cginc"
#include "FXLab.cginc" // make this path relative to your shader path
...
```

With this done, you now can write the FXLab shader, the FXLab.cginc provides all neccessary types and methods to make it easy for you.

As most of the effects delivered by the FXLab work by using the screen space you need to define screen space coordinates. This is very easy, just extend the code by the following additions:

```
struct v2f {
        ...
        float4 screenPos : TEXCOORD1; // Add this line
        ...
};

...

v2f vert (appdata v) {
        ...
        o.screenPos = o.pos; // Add this line
        ...
}

...

float4 frag( v2f o ) : COLOR
{
        float screenUv = calcScreenUv(o.screenPos); // Add this line
        ...
}
```

You can now access the generated FXLab data with the help of the methods provided by the FXLab.cginc. You can for example read the content from a distortion texture to distort the background by a given factor:

```
Properties {
        ...
        _DistortionTex ("Distortion Texture", 2D) = "bump" {}
        _DistortionStrength ("Distortion Strength", Float) = 0.01
}

...

sampler2D _DistortionTex;
float _DistortionStrength;

...

fixed4 frag( v2f o ) : COLOR
{
        float screenUv = calcScreenUv(o.screenPos);

        float2 distortion = UnpackNormal(tex2D(_DistortionTex, o.uv)).xy * _Dist
        fixed3 refraction = sampleScreen(distortion + screenUv);

        return fixed4(tex2D(_MainTex, o.uv).xyz * refraction, 1);
}
```

When you want to use a selectable render texture you need to add a property to your shader with a FXTexture type in the description (even when you want to use the result of a captured screen):

```
Properties {
        ...
        _CustomTexture ("Custom Texture (FXScreenTexture)", 2D) = "" {}
}

...

sampler2D _CustomTexture; // required
float4 _CustomTexture_Area; // required
float2 _CustomTexture_TexelSize; // only required when you want to use the texel

...

fixed4 frag( v2f o ) : COLOR
{
        float screenUv = calcScreenUv(o.screenPos);

        fixed3 color = sampleColor(_CustomTexture, screenUv);
        return fixed4(color, 1);
}
```

6.2 Surface Shaders

Writing surface shaders is similar to writing normal shaders, you only need to specify a float4 into your Input structure:

```
struct Input
{
        ...
        float4 screenPos; // Add this line
        ...
};

...

void surf (Input IN, inout SurfaceOutput o)
{
        float screenUv = calcScreenUv(o.screenPos); // Add this line
        ...
}
```

After this, you can use the sample functions like in any other shader.

6.3 Built in Variables

The FXLab.cginc defines many helper and required methods for the FXLab. You can use them to write custom shaders, but for this you need to know what you can do.

float2 calcScreenUv(float4 screenPos)

A function for the usage inside of the fragment shader. Add this into your shader to define a set of screenspace texture coordinates.

Returns a float2 texture coordinate variable.

float fresnelTerm(float3 normal, float3 eyeVec, float refractionStrength, float fresnelFactor, float fresnelBias)

Calculates the fresnel term based upon the input variables for a water material.

- float3 normal - the surface normal
- float3 eyeVec - the view direction
- float refractionStrength - the strength of the resulting refraction
- float fresnelFactor - a factor which will be used when calculating the fresnel term
- float fresnelBias - a offset which will be used when calculating the fresnel term

float4 _ReflectionPlaneEquation

The plane equation when using a reflection plane

float4 _HeightPlaneEquation

The plane equation when using a height plane

float sampleDepth(float2 uv)

A function which will sample depth of the scene, ranging from 0 to camera far plane.

- float2 uv - the screenspace texture coordinate

Returns a float variable.

float sampleHeight(float2 uv)

A function which will samples the height from the ground at the fragment position to the height plane surface above.

- float2 uv - the screenspace texture coordinate

Returns a float variable.

float3 sampleScreen(float2 uv)

A function which will sample the color of the screen buffer texture.

- float2 uv - the screenspace texture coordinate

Returns a float3 variable.

float3 sampleReflection(float2 uv)

A function which will sample the color of the reflection texture.

- float2 uv - the screenspace texture coordinate

Returns a float3.

float3 sampleColor(sampler2D texture, float2 uv)

Samples the content of the render texture.

- sampler2D texture - the render texture to sample from
- float2 uv - the screenspace texture coordinate

Returns a float3.

float3 sampleColorBlurred(sampler2D texture, float2 uv, float radius)

Samples the blurrend content of the render texture.

- sampler2D texture - the render texture to sample from
- float2 uv - the screenspace texture coordinate
- float radius - the radius of the blur

Returns a float3.

float sampleFloat(sampler2D texture, float2 uv)

Samples the content of the float render texture.

- sampler2D texture - the float render texture to sample from
- float2 uv - the screenspace texture coordinate

Returns a float.
Note that a float textures ranges from 0 to 1. The user must scale the values to a meaningful value.

sampler2D _FXScreenTexture;

The default FXScreenBufferTexture render texture.

sampler2D _FXDepthTexture;

The default FXDepthTexture render texture.

sampler2D _FXHeightTexture;

The default FXHeightTexture render texture.

```
sampler2D _FXReflectionTexture;
```

The default FXReflectionTexture render texture.

```
sampler2D _FXWorldNormalTexture;
```

The default FXWorldNormalTexture render texture.

7. FAQ

Frequently asked questions.

7.1 FAQ

Q: Why am I unable to see the effects?

1. Make sure that your main camera has an assigned FX Camera.
2. Make sure that your main camera has all needed FX Textures assigned.

Q: There are strange moving colors on FXLab effects, why?

You must exclude the GameObjects with FXLab effects from beeing rendered through the FX Camera, look at the Setup chapter to see how to do this.

Q: Wrong colors when using the water effect with enabled refractions, why?

1. You must check that your reflection plane is correctly placed and that you has set the correct GameObject as Reflection Plane inside the FX Reflection Texture.
2. Check the View and HeightDepth values inside the material you use for the water effect. A too low value might result in an too early full opaque effect.

7.2 Contact

You can contact us by following services:

EMail:
Mark@Dras.biz

Facebook:
FXLab (https://www.facebook.com/UnityFXLab)

Twitter:
MarkLeutloff (https://twitter.com/MarkLeutloff)

Unity Forum:
FXLab (http://forum.unity3d.com/threads/204481)