

【翻译】第七章节：顺序无关的透明度（关于顺序无关的混合）

2014-11-28 08:47:00 1979 人阅读 [Unity3D](#) [cg](#) [顺序无关透明度](#)

A⁻ A⁺

文章内容	例子源码	网友评论	最后编辑：2014-12-21 18:03:26
本文永久地址： http://www.omuying.com/article/95.aspx ，【 文章转载请注明出处！ 】			

本章节介绍 order-independent blending。

在《[透明度](#)》章节中，我们讨论并解决了在着色器上呈现透明度一些标准的问题，如果你还没有阅读这个章节，那么你应该先去阅读它。

顺序无关的混合

在《[透明度](#)》章节中，我们知道，混合的结果取决于三角形的呈现顺序（尤其是 alpha 混合），如果三角形不是按从背面到前面的呈现方式，这样可能会导致渲染问题，术语 “order-independent transparency” 一词描述各种技术来避免这个问题，其中的一个技术便是 “order-independent” 混合，即使用一个不依赖于三角形栅格化顺序的混合公式，他们是：additive blending 和乘法（multiplicative）blending。

Additive Blending

照片的二次曝光是标准使用 additive blending 的例子，如下图：



我们不可能（至少很难）说是照片以何种顺序拍摄。additive blending 的特点可以依据《[透明度](#)》章节的混合公式：

```
1 float4 result = SrcFactor * fragment_output + DstFactor * pixel_color;
```

其中 SrcFactor 和 DstFactor 由 Unity 的语法决定：

```
1 Blend {code for SrcFactor} {code for DstFactor}
```

对于 additive blending，DstFactor 必须是 One，SrcFactor 的值不能取决于帧缓冲区里面的像素颜色，它可以是 One、SrcColor、SrcAlpha、OneMinusSrcColor 或者 OneMinusSrcAlpha。

着色器的代码是：

```
01 Shader "Cg shader using additive blending"
02 {
03     SubShader
04     {
05         Tags { "Queue" = "Transparent" }
06         // draw after all opaque geometry has been drawn
```

比基月除千

最新文章

暂无图片

【原创】C# 基础之 Lambda表达式 - 907 次阅读

暂无图片

【原创】C#基础之 IEnumerable和 IEnumerator - 792 次阅读

暂无图片

【原创】C#基础之事件 - 886 次阅读

暂无图片

【原创】C#基础之委托 - 912 次阅读

暂无图片

【原创】C#基础之委托的使用 - 856 次阅读

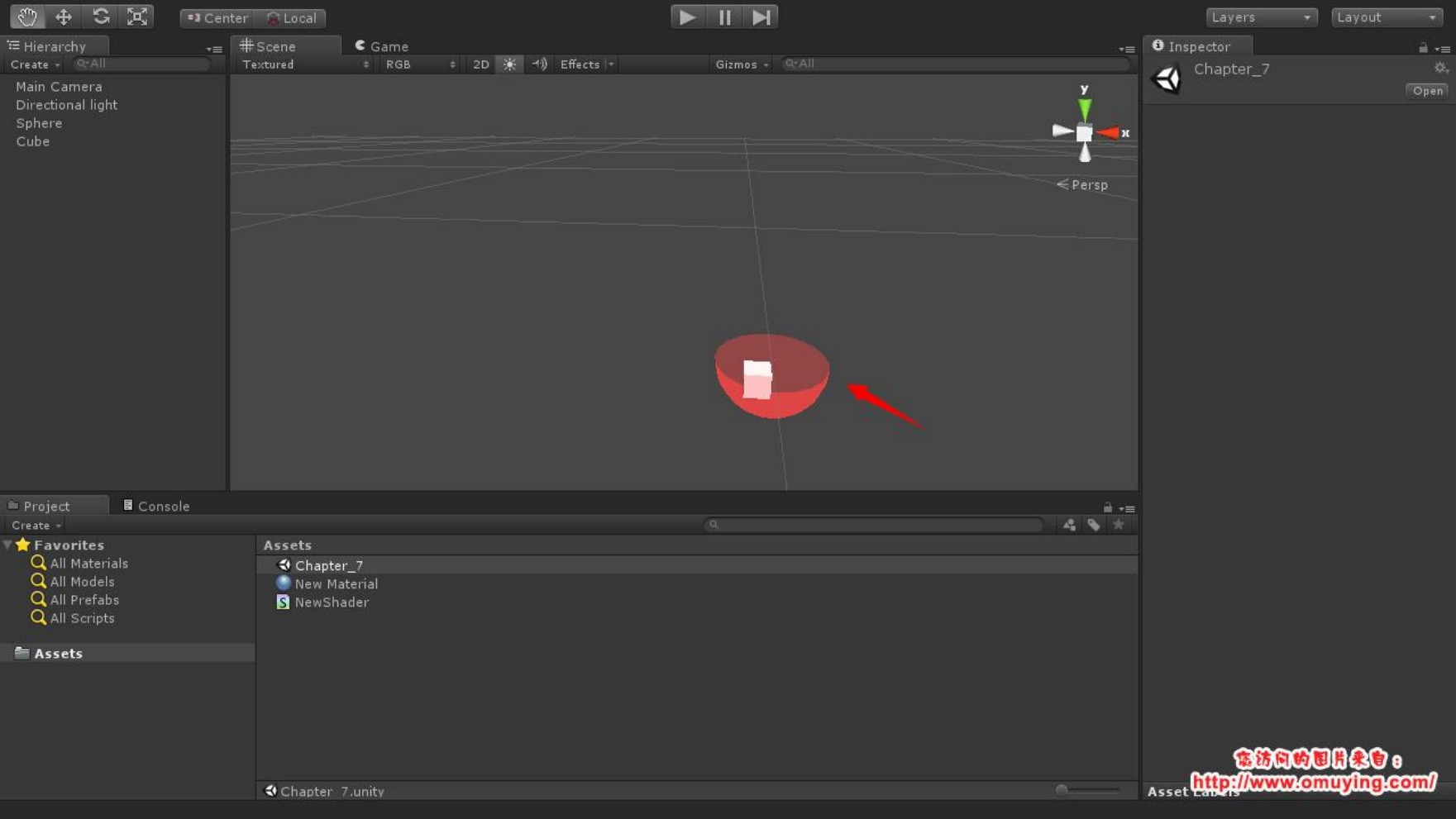


广告

随机阅读

```
07 Pass
08 {
09     Cull Off // draw front and back faces
10     ZWrite Off // don't write to depth buffer
11     // in order not to occlude other objects
12     Blend SrcAlpha One // additive blending
13
14     CGPROGRAM
15
16     #pragma vertex vert
17     #pragma fragment frag
18
19     float4 vert(float4 vertexPos : POSITION) : SV_POSITION
20     {
21         return mul(UNITY_MATRIX_MVP, vertexPos);
22     }
23
24     float4 frag(void) : COLOR
25     {
26         return float4(1.0, 0.0, 0.0, 0.3);
27     }
28     ENDCG
29 }
30 }
31 }
```

把着色器应用到一个球体上查看效果，如图：



Multiplicative Blending

multiplicative blending 使用的例子是在摄影中使用多个均匀（uniform）的灰色滤波器：摄像机滤波器的顺序对于图像的衰减是无关紧要的，在三角形的栅格化方面，图像对应于在三角形栅格化之前帧缓冲区中的内容，而滤波器对应于三角形。

可以像下面的格式来指定 multiplicative blending：

```
1 Blend {code for SrcFactor} {code for DstFactor}
```

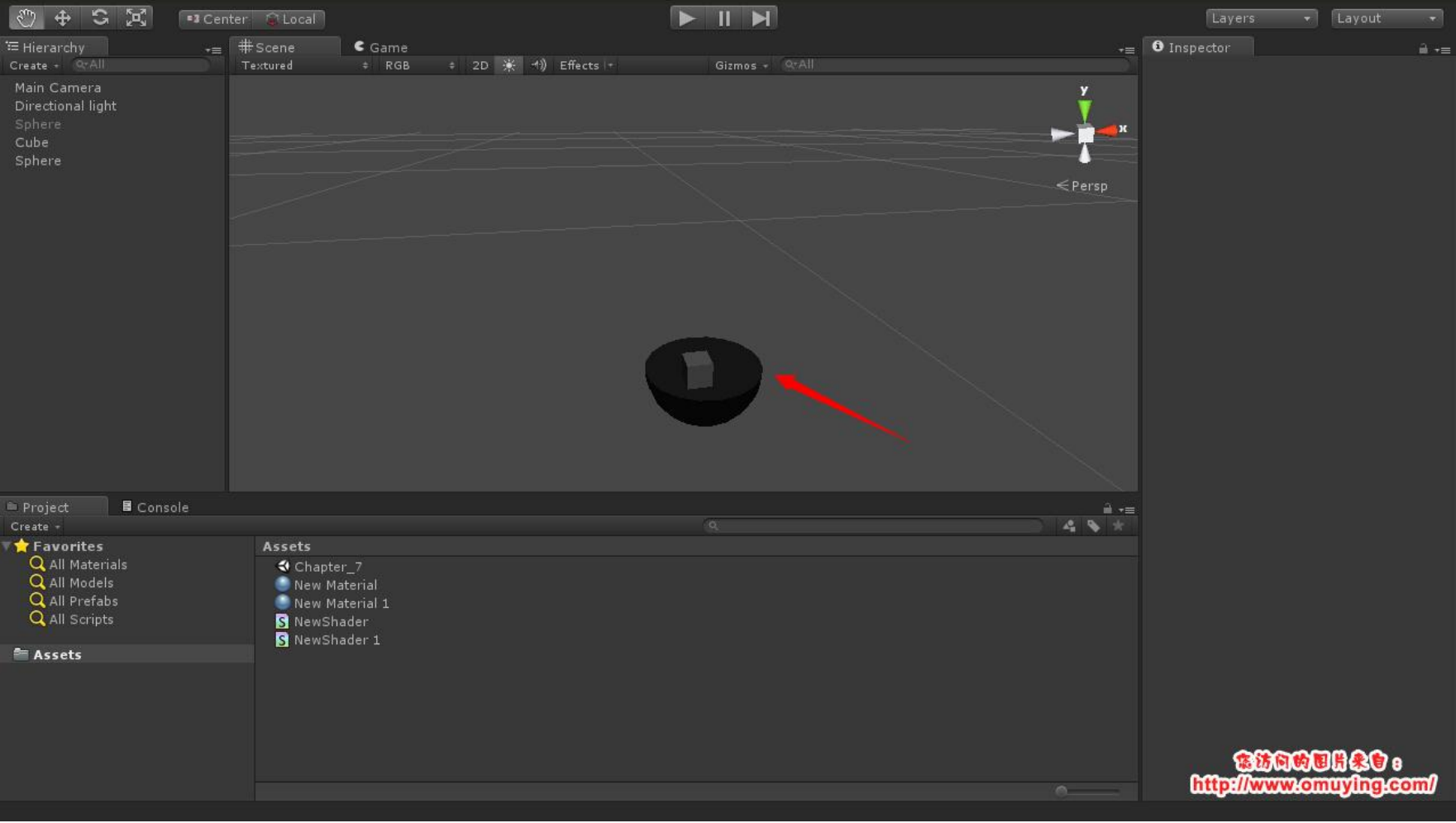
对于 SrcFactor 必须是 Zero，并且 DstFactor 必须取决于片段颜色，所以可能的值是 SrcColor、SrcAlpha、OneMinusSrcColor 或者 OneMinusSrcAlpha。典型的例子是 DstFactor 使用 OneMinusSrcAlpha 来表示用片段的 alpha 分量来指定不透明背景的衰减：

```
01 Shader "Cg shader using multiplicative blending"
02 {
03     SubShader
04     {
05         Tags { "Queue" = "Transparent" }
06         // draw after all opaque geometry has been drawn
07         Pass
08         {
09             Cull Off // draw front and back faces
10             ZWrite Off // don't write to depth buffer
11             // in order not to occlude other objects
12             Blend Zero SrcAlpha // multiplicative blending
13             // for attenuation by the fragment's alpha
14
15             CGPROGRAM
```

 暂无图片	【翻译】第十二章：光滑的镜面高光（关于每像素光照） - 1181 次阅读
 暂无图片	【原创】Shader 内置 Shader 之 Bumped Specular 学习 - 1785 次阅读
 暂无图片	【翻译】第九章：漫反射（关于每顶点漫反射和多光源漫反射） - 1946 次阅读
 暂无图片	【原创】Shader 内置 Shader 之 Specular 学习 - 2024 次阅读
 暂无图片	【翻译】第十五章：纹理球（关于纹理球面） - 1907 次阅读

```
16         #pragma vertex vert
17         #pragma fragment frag
18
19
20         float4 vert(float4 vertexPos : POSITION) : SV_POSITION
21         {
22             return mul(UNITY_MATRIX_MVP, vertexPos);
23         }
24
25         float4 frag(void) : COLOR
26         {
27             return float4(1.0, 0.0, 0.0, 0.3);
28         }
29     ENDCG
30 }
31 }
32 }
```

着色器的效果如图：



完成着色器

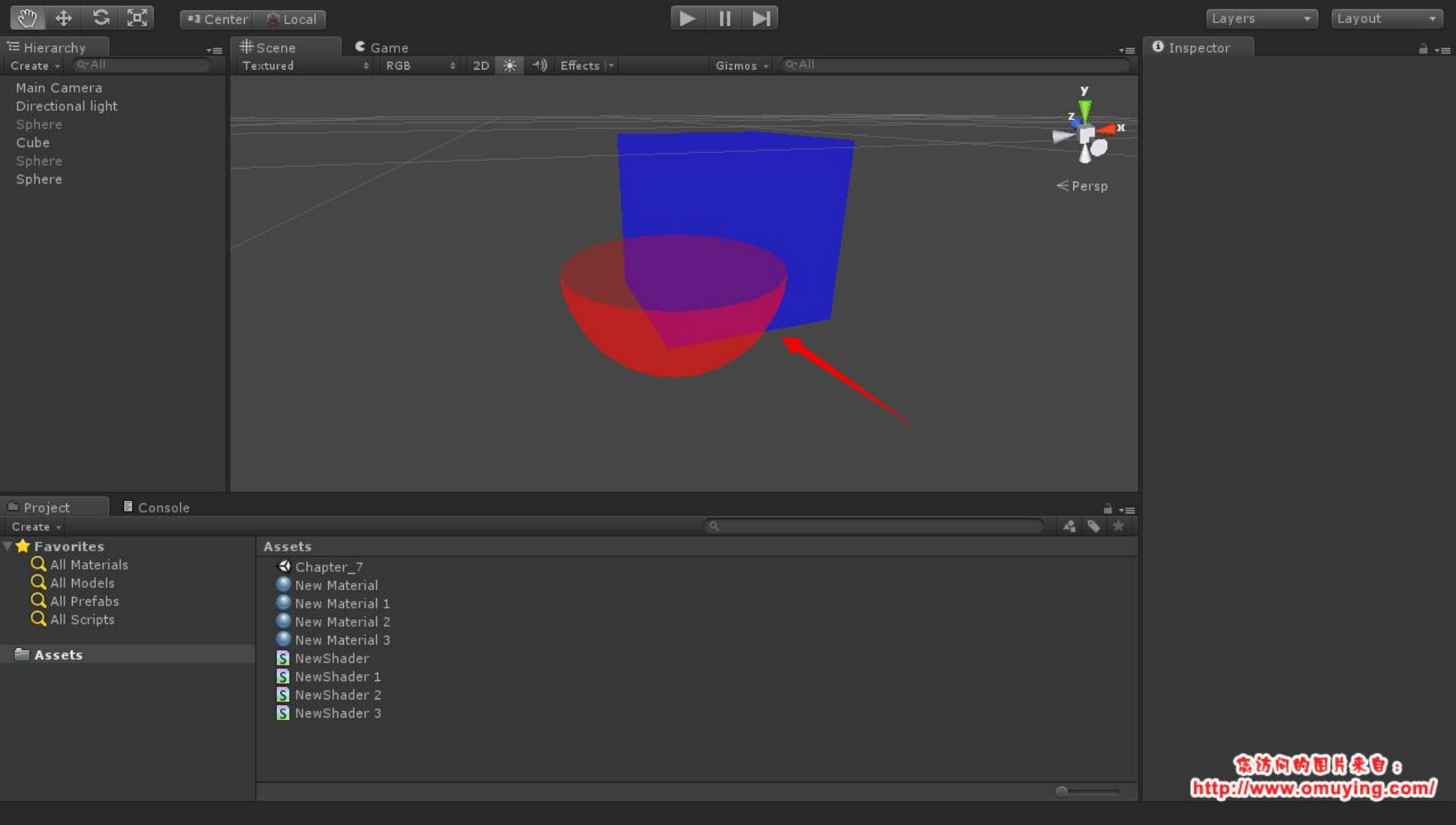
最后，我们可以在一个着色器中使用两个 Pass 来合并 multiplicative blending（指定背景的衰减）和 additive blending（给三角形添加颜色），如果忽略三角形网格本身的颜色衰减，这可以被认为是一种近似小不透明体（small opacities）的 alpha 混合，即小的 alpha 值。

```
01 Shader "Cg shader using order-independent blending"
02 {
03     SubShader
04     {
05         Tags { "Queue" = "Transparent" }
06         // draw after all opaque geometry has been drawn
07         Pass
08         {
09             Cull Off // draw front and back faces
10             ZWrite Off // don't write to depth buffer
11             // in order not to occlude other objects
12             Blend Zero OneMinusSrcAlpha // multiplicative blending
13             // for attenuation by the fragment's alpha
14
15             CGPROGRAM
16
17             #pragma vertex vert
18             #pragma fragment frag
19
20             float4 vert(float4 vertexPos : POSITION) : SV_POSITION
21             {
22                 return mul(UNITY_MATRIX_MVP, vertexPos);
23             }
24
25             float4 frag(void) : COLOR
26             {
27                 return float4(1.0, 0.0, 0.0, 0.3);
28             }
29             ENDCG
30         }
31
32         Pass
33         {
34             Cull Off // draw front and back faces
35             ZWrite Off // don't write to depth buffer
```



```
36 // in order not to occlude other objects
37 Blend SrcAlpha One // additive blending to add colors
38
39 CGPROGRAM
40
41 #pragma vertex vert
42 #pragma fragment frag
43
44 float4 vert(float4 vertexPos : POSITION) : SV_POSITION
45 {
46     return mul(UNITY_MATRIX_MVP, vertexPos);
47 }
48
49 float4 frag(void) : COLOR
50 {
51     return float4(1.0, 0.0, 0.0, 0.3);
52 }
53 ENDCG
54 }
55 }
56 }
```

注意这两 pass 的顺序很重要，第一个表示背景的衰减接着是颜色被添加。着色器的效果如图：



恭喜你，在本篇你应该了解：

- 1、什么是顺序无关的透明度（ order-independent transparency ）和顺序无关的混合（ order-independent blending ）。
- 2、实现 order-independent blending 两种重要的方式（ Additive 和 Multiplicative ）。
- 3、怎样实现 additive and multiplicative blending。
- 4、如何结合两个 pass 并合并（ additive 和 multiplicative blending ）来实现无顺序的 alpha 混合。

资源下载地址：[点击下载](#)，共下载 32 次。

前一篇：[第六章节：透明度（关于混合）](#)
后一篇：[第八章节：轮廓加强（关于转换法线向量）](#)



赞
4 人



打酱油
0 人



呵呵
0 人



鄙视
0 人








正能量
4 人




0

还没有评论，沙发等你来抢

社交帐号登录:  微信  微博  QQ  人人 [更多»](#)



说点什么吧...







发布


最终幻想正在使用多说

0条评论


[最新](#) [最早](#) [最热](#)

还没有评论，沙发等你来抢

社交帐号登录:  微信  微博  QQ  人人 [更多»](#)



说点什么吧...



发布

最终幻想正在使用多说