屏幕后处理效果

屏幕后处理效果（Screen Post-Processing Effects）是游戏开发过程中实现屏幕特效的一种常用方法，通常指的是在渲染完整个场景得到屏幕图像后，再对这个图像进行一系列地操作，来实现各种屏幕特效。它可以为游戏画面添加更多的艺术效果，例如景深、运动模糊、Bloom等等，最终渲染到屏幕中的画面往往叠加了多种屏幕特效，就像栈（Stack）一样。Unity官方提供了一套Post-Processing Stack后处理框架和各种常用的屏幕特效供开发者使用，本节我们将自己搭建一套后处理框架，并实现最常用的Bloom屏幕特效。

创建资产

1. 一个项目中可能需要多个Post-FX栈配置，因此我们在Runtime子文件夹下创建一个PostFXSettings脚本，用它来创建PostFXSettings资产作为Post-FX栈的配置。

3. 新建一个Render方法用于渲染后处理特效，调用buffer.Blit方法完成对图像的处理并显示到屏幕上。

5. 目前我们始终是直接将图像渲染到帧缓冲区中，但我们没有对该缓冲区的控制权，只能写入它们。如果要为后处理特效栈提供源纹理，我们必须创建一个渲染纹理来作为摄像机的中间帧缓冲区，并把它设置为渲染目标，这要在Setup方法清除渲染目标之前进行该操作。

7. 当渲染到中间帧缓冲区时，我们会给渲染纹理填充任意的数据，这在启用Frame Debugger时可以看到此情况。Unity会确保帧调试器在每帧开始时获得清理后的帧缓冲区，但当渲染到我们自己的纹理中时，我们会回避这个问题。它通常会导致我们在上一帧绘制的结果中继续绘制，但不能保证一定会这样做。如果摄像机的ClearFlags属性设置为Skybox或者Solid Color还好，因为保证可以完全覆盖旧的数据，颜色缓冲和深度缓冲都会被清除，但使用Depth only和Don't Clear是做不到的。因此当后处理特效栈被启用时，应当始终清除颜色和深度缓冲，我们在Setup方法中对相机的ClearFlags进行强制设置。

Gizmos

1. 我们目前在同一时刻绘制了所有的Gizmos，但在后处理特效渲染前后Gizmos应该有一些显示区别，因此在CameraRenderer.Editor脚本中我们将DrawGizmos方法一分为二。

2. 然后在Render方法中根据后处理特效渲染的前后分别渲染它们。

自定义绘制

我们当前使用的Blit方法会绘制一个覆盖了整个屏幕空间的由两个三角形组成的四边形网格（这是原来的后处理的做法），但是我们可以通过创建一个将整个屏幕包括在内的大三角形获得相同的结果，超出屏幕部分会由GPU自行裁剪（下面有图），这样工作量会少一些，甚至不需要向GPU发送单个三角形网格，而且可以使用程序来生成它。

这样做的好处有两个：

第一个好处是顶点数量由6个减到了3个，最重要的是消除了四边形的两个三角形相交的对角线，会在渲染流水线的三角形遍历阶段检查每个像素是否被一个三角网格所覆盖。如果被覆盖就生成一个片元，由于四边形有两个三角形，因此沿着对角线重叠的片元将被渲染两次（如下图所示，红色区域片元会被渲染两次），导致渲染效率低下。

第二个好处是可以提高硬件缓存的命中率，两个三角形在光栅化的时候缺乏连贯性，导致缓存预测中断，而单个三角形可以更好地保持本地缓存的连贯性。