Cha4-Unity3d 和 C#的双剑合璧 07

在上一课的内容中，我们学会了如何使用键盘来控制游戏角色。当然，在本部分内容最后的部分，我还会带着大家一起学习如何使用触摸屏来控制游戏角色。

不过遗憾的是，当角色离我们渐行渐远时，玩家就只能遗憾的看着角色美丽的背景消失了虚拟世界之中~既然是游戏，既然是虚拟世界，那玩家理应是上帝，理应时时刻刻都可以清清楚楚明明白白真真切切看清楚角色的一举一动~

那么，该怎么办呢？在这一课的内容中，我们会深入了解摄像机，从而让游戏角色可以永远闪耀在最闪亮的舞台中央。

还等什么呢？让我们开始吧~

个人微信号：iseedo

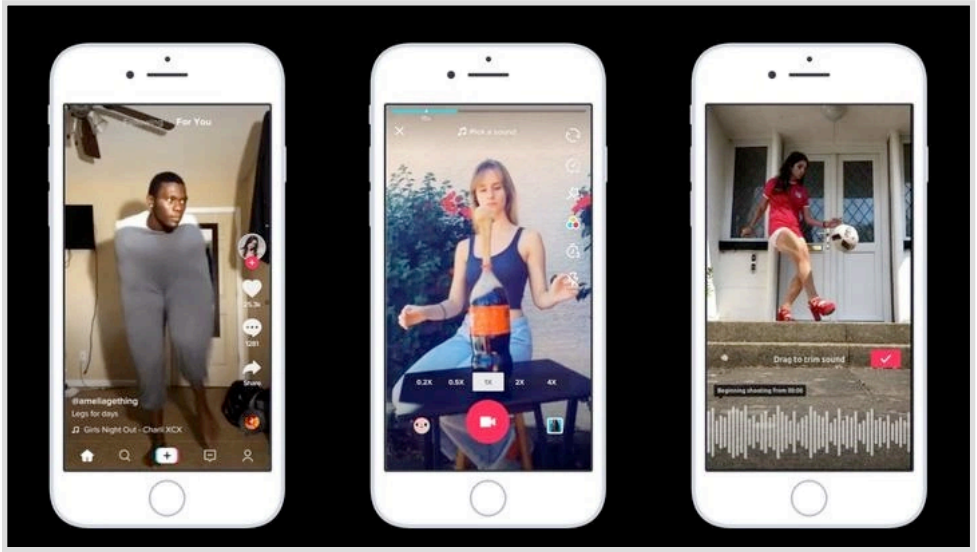
微信公众号：vrlife

07 Unity 和 C#的双剑合璧-让角色永远在最闪亮的舞台中央

打开 Unity，打开 FairyLand 项目。

默认情况下，我们已经看到打开的 SampleScene。

点击 Main Camera，这就是我们这一课内容中需要重点了解的游戏对象，也就是摄像机。



什么是摄像机？我们先来谈谈日常生活中的摄像机。

先拿如今人手一部的手机来说，每个手机上都有一个或多个摄像头，可以用来拍照或者录制视频。当我们录制亲朋好友的视频或者自拍、抖音直播时，或者是拍摄天性活泼好动的小孩和小动物时，摄像机的视角是始终固定的。当然，我们也可以用调整焦距的方式来拉伸或缩短人物在场景中的位置。

在拍摄电影、演唱会、大型表演、体育赛事的时候，会使用多个摄像机（俗称多机位）对角色和场景进行捕捉，并通过切换机位和后期制作的方式将精彩的画面无缝拼接在一起。

在 Unity 中，摄像机（Camera)的作用与之类似，只不过在 Unity 中，Camera 用来捕捉游戏场景，然后将画面实时展示给玩家。

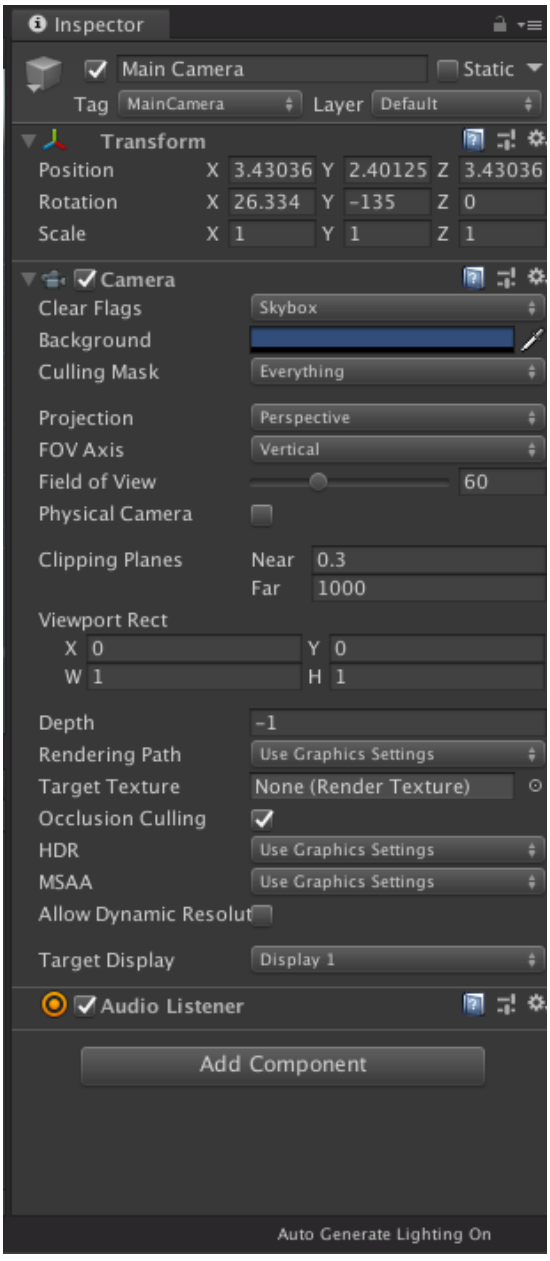
通过设置和操控 camera，我们可以让玩家所看到的游戏画面独一无二。比如，在解谜类型的游戏中，玩家可能希望时刻都能对迷宫的全局了然于心。在第一人称射击游戏中，摄像机往往跟玩家角色在一起，并且处在角色的眼睛位置。对于赛车游戏，我们就希望摄像机可以始终跟随者玩家所驾驶的汽车。

而且，就跟拍电影一样，我们可以在一个场景中放置多个摄像机。当然，这些摄像机可以使用任何顺序进行渲染，可以放置在场景中的任一位置，亦或是场景中的部分位置。

点击 SampleScene 场景中的 Main Camera 对象，在 Inspector 视图中可以看到对应的属性：

在 Unity 的官方文档中，对 Camera 组件各种属性的作用做了详细的说明：

https://docs.unity3d.com/Manual/class-Camera.html



这里我们简单说明其中一些重要属性的作用。

1.Clear Flags

1. Clear Flags:清除标记。决定哪些屏幕部分将会被清除，当使用多个摄影机来显示不同游戏元素时非常有用。有 3 种模式选择：

Skybox：默认模式，单个摄影机时选择，表示背景使用天空盒。在屏幕中的空白部分将显示当前摄像机的天空盒。如果当前摄像机没有设置天空盒，会默认用 Background 色。

Solid Color：纯色，单个摄影机时选择，表示背景使用单一颜色。选择该模式屏幕上的空白部分将显示当前摄像机的 background 色。

Depth only：多个摄影机时选择，根据 Depth 决定哪个摄影机所渲染的画面在上方，越高的 Depth 值越处于上方

Dont Clear：该摄影机在切换帧的时候不清除前一帧的颜色及深度信息，也就是说，会显示为叠加重影的效果。其结果是，每一帧渲染的结果叠加在下一帧之上。一般与自定义的 shader 配合使用。

2. Background：摄影机背景颜色（当使用 Solid Color 时，通过 Background 参数设置具体颜色）

3. Culling Mask：通过 Layer（层）的设置来屏蔽一部分游戏物体在此摄影机中不被渲染

4. Projection：摄影机投影方式

Perspective：透视摄影机

Orthographic：正交摄影机

5. FOV Axis：视野的轴线，有 Vertical（垂直）和 Horizontal（水平）两种选择。

6.Field of view：透视摄影机的视野，越大越透视

7.Physical Camera：物理摄像机

是否勾选代表是否摄像机可以产生物理碰撞

8. Clipping Planes： 剪裁平面。设置摄影机的最远和最近渲染距离，也就是摄像机开始渲染与停止渲染之间的距离。

Near：近点。摄像机开始渲染的最近的点。

Far：远点。摄像机开始渲染的最远的点。

9. Viewport Rect：标准视图矩形，设置摄影机渲染其视野画幅的哪一部分。用四个数值来控制摄像机的视图将绘制在屏幕的位置和大小，使用的是屏幕坐标系，数值在 0~1 之间。坐标系原点在左下角。

10. Depth：深度。用于控制摄像机的渲染顺序，较大值的摄像机将被渲染在较小值的摄像机之上，在有多个摄像机时非常有用。

11. Rendering Path：渲染路径。用于指定摄像机的渲染方法。

Use Player Settings：使用 Project Settings-->Player 中的设置。

Vertex Lit：顶点光照。摄像机将对所有的游戏对象座位顶点光照对象来渲染。

Forward：快速渲染。摄像机将所有游戏对象将按每种材质一个通道的方式来渲染。

Deferred Lighting：延迟光照。摄像机先对所有游戏对象进行一次无光照渲染，用屏幕空间大小的 Buffer 保存几何体的深度、法线已经高光强度，生成的 Buffer 将用于计算光照，同时生成一张新的光照信息 Buffer。最后所有的游戏对象会被再次渲染，渲染时叠加光照信息 Buffer 的内容。

12. Target Texture: 目标纹理。用于将摄像机视图输出并渲染到屏幕。一般用于制作导航图或者画中画等效果。举个简单的例子，在 FPS 对战游戏中可以用小地图来显示角色所在的位置。

13.Occlusion Culling：是否开启遮挡剔除

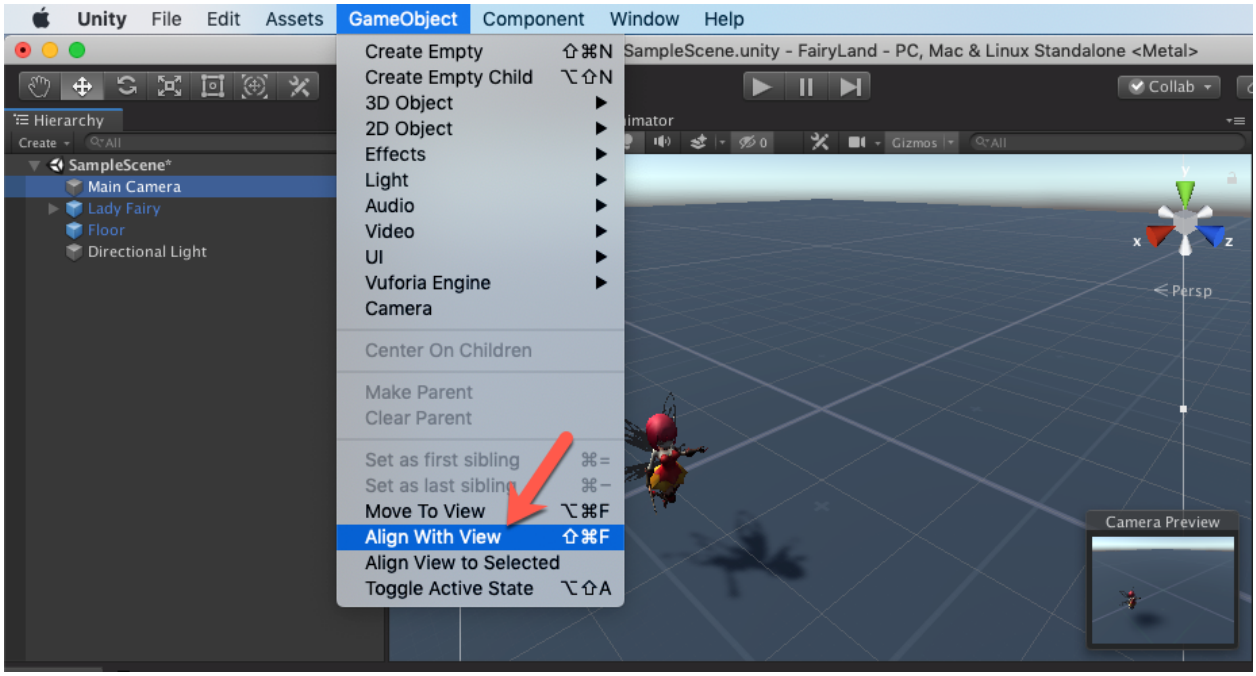
14. HDR：高动态光照渲染。用于启用摄像机的高动态范围渲染功能。

15.MSAA:允许进行硬件抗锯齿

16.Target Display:目标显示器，可以设置 1-8

在上一课结束的时候，其实我们已经对摄像机的视角做了一个小小的设置。

之所以我们要进行这样的设置，是因为在 Unity 中，主摄影机和场景视图（Scene View）的摄影机角度是不一样的，新人常常在场景视图中对好了正确的摄影机角度，然后点击“运行”，结果发现游戏画面完全不是自己想要的样子。



如果希望将主摄影机对齐我们的场景视图，只需要选择主摄影机物体，然后通过菜单：GameObject > Align With View 就可以了。这个命令实际上会将所选择的任何物体都与场景摄影机对齐，并不仅限于对齐主摄影机。

当然，在上一课中，我们使用的是快捷键，起到的是完全相同的作用。

其它两个命令的作用也简单说一下：

Move To View：

将游戏对象移动到当前视角的正中心位置（并不会改变旋转方向），如果我们希望看到场景中某个角落里的物体，就可以使用这个命令；

Align View to Selected：

可以将摄影机移动到选择的游戏对象上去，可以用来将场景摄影机与某个场景游戏对象对齐；

Toggle Active State：

激活/取消激活游戏对象，类似于一些 3D 软件中的“隐藏”。

好了，现在我们已经对 Unity 中的摄像机有了一个基本的认识。

那么，如何让主摄像机始终跟随游戏角色呢？

聪明的童鞋可能已经很快想到了一个简单的办法，那就是在 Hierarchy 视图中，让摄像机这个游戏对象成为 Lady Fairy 游戏对象的子对象。

那么，当角色移动时，主摄像机就会跟着一起移动了。

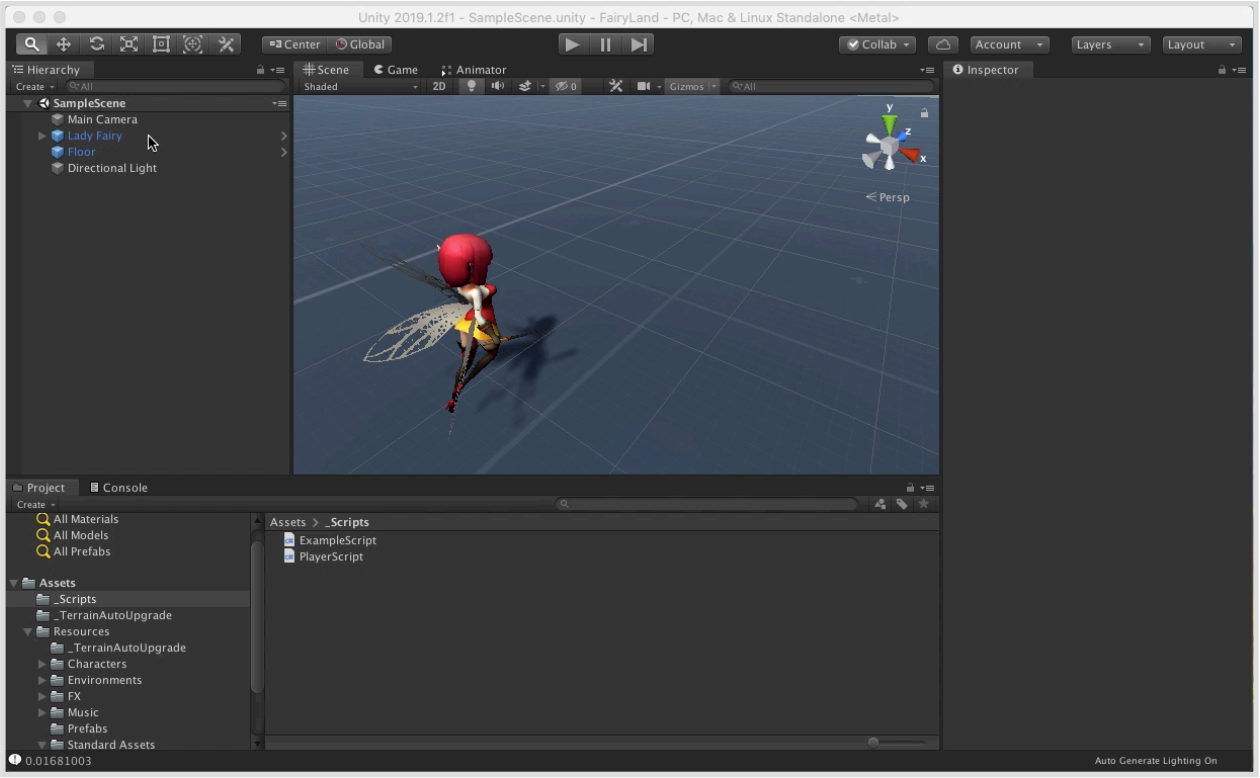
好了，让我们动手试试看会是怎样的情况~

可以看到，这是实现主摄像机对角色跟随拍摄，最简单直接粗暴的实现方式~

然鹅，这样做的坏处也很多。因为主摄像机现在是游戏角色的子对象，所以当游戏角色的 Transform 信息突然发生大的变化时，绝 b 会让充当“上帝”的玩家大吃一惊。特别是当角色因为种种原因出现了高速旋转，那么主摄像机也会跟着一起欢乐转一转~而且，别忘了万一角色突然变形了呢？

相信最终的结果就是玩家直接晕菜。

那么，除了这种最简单直接粗暴的方法，还有哪些方法呢？



但是大家都知道，在 Inspector 视图中，我们只能对游戏对象的属性提前进行调整，而无法在游戏运行的过程中进行修改。

那么，如果我们希望在游戏进行的过程中动态更改某个游戏对象的属性，显然就只有靠游戏脚本了~

所以，假如我们希望主摄像机在游戏进行的过程中如同“上帝视角”一样始终把焦点锁定到角色身上，就需要借助代码的原力了~

那么，应该怎么办呢？

在学习 Unity3d 的过程中，当遇到自己没有遇到过的问题时，首先应该怎么办？

有些童鞋会第一时间求助度娘或者谷歌，不过我个人的习惯是第一时间求助官方文档~

对于这类相对简单的问题（比如不知道该用什么 API，或者不知道类的方法有哪些，以及具体怎么用），我的查询顺序是：

Step1.查询官方文档

如果解决，ok，继续前进。如果不行，转 Step2

Step2.查询 Stackoverflow/Github/Google/Youtube/度娘

如果解决，ok，继续前进。如果不行，转 step3

Step3.去一些讨论论坛（比如 unity3d 官方讨论社区，或是 csdn 等等）

如果解决，ok，继续前进。如果不行，转 step5

Step4.去悬赏问答或是找牛人帮忙

当然，如果需要到 step4，可以说是 99.9%的人都没有遇到过的问题了~

所以首先采用 Step1,查询官方文档。

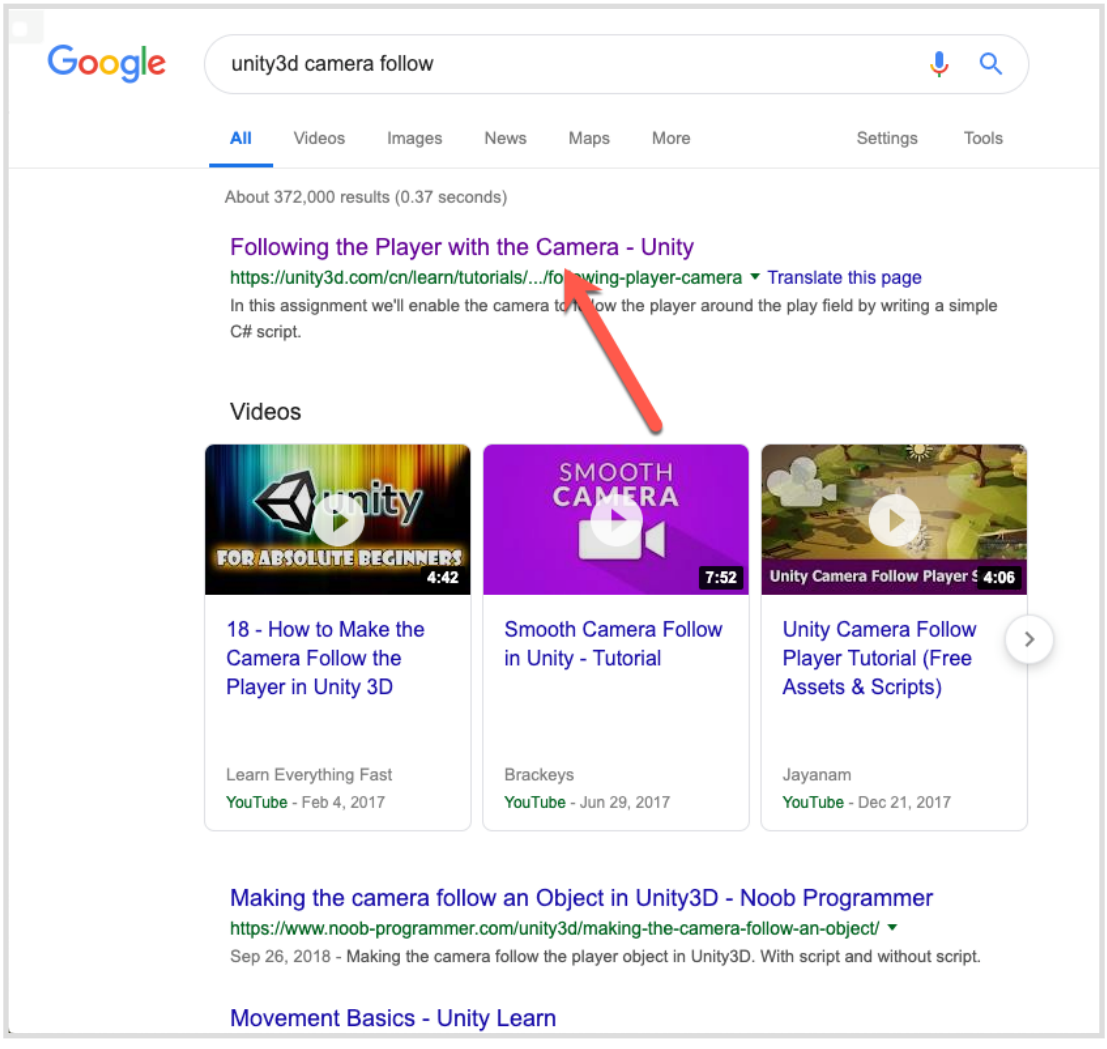
在 Unity 的官方文档的 Scripting API 中查询 Camera,找到如下页面：

https://docs.unity3d.com/ScriptReference/Camera.html

似乎并没有直接的解决方法。

那么跳转 Step2,搜索 unity3d camera follow，直接就发现一个官方的示例教程~

打开一看，一个详细的视频教程，还有示例代码，还要什么🚲~



照着试试看？

在 Project 视图中，在_Scripts 目录下新建一个脚本文件，将其命名为 CameraController，然后双击在 Visual Studio 中打开。

更改其中的代码如下：

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;


public class CameraController : MonoBehaviour
{
    //1.定义一个指向玩家角色的 GameObject 对象
    public GameObject player;


    //2.定义一个 Vector3 类型的变量，用来保存玩家角色和摄像机之间的坐标差
    private Vector3 offset;
    // Start is called before the first frame update
    void Start()
    {
        //3.计算玩家和摄像机之间的坐标差
        offset = transform.position - player.transform.position;
    }


    // Update is called once per frame
    void Update()
    {
        //4.根据玩家的最新空间坐标计算摄像机的坐标
        transform.position = player.transform.position + offset;
    }
}
```

以上代码超级简单，这里按照注释的数字顺序简单解释一下：

1.定义一个指向玩家角色的 GameObject 对象

2.定义一个 Vector3 类型的变量，用来保存玩家角色和摄像机之间的坐标差

3.计算玩家和摄像机之间的坐标差

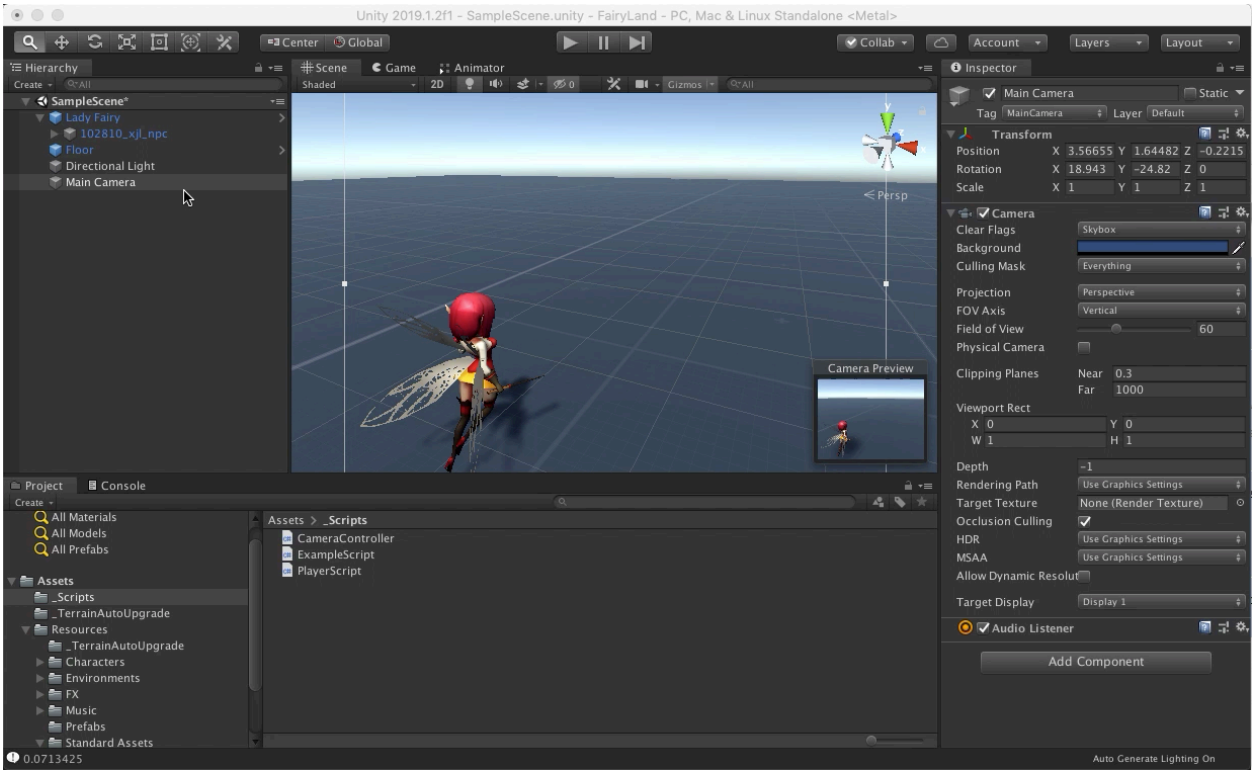
注意是在 Start()方法中计算的，也就是游戏开始的时候

4.根据玩家的最新空间坐标计算摄像机的坐标

注意是在 Update()方法中计算的，也就是游戏的每一帧都会重新计算~

回到 Unity 编辑器，在 Hierarchy 视图中选中 Main Camera，然后把 CameraController 添加为 Main Camera 的组件。

接下来还需要把 Camera Controller(Script)组件中的 Player 属性设置为 Lady Fairy。



完成之后点击工具栏上的预览按钮，就可以看到摄像机跟随游戏角色的效果了。

当然，需要提醒大家的是，这是最简单的相机跟随效果。

感兴趣的童鞋还可以参考以下文章：

https://blog.csdn.net/u011484013/article/details/51554745
https://blog.csdn.net/wuyt2008/article/details/52494884

好了，本课的内容就先到这里了。

到目前为止，我们已经成功实现了玩家和游戏角色之间的互动，或者说操控，实现了玩家当“上帝”的感觉~

在接下来的内容中，我们将迈出关键的下一步，也就是让游戏角色之间实现互动~

让我们下一课再见。