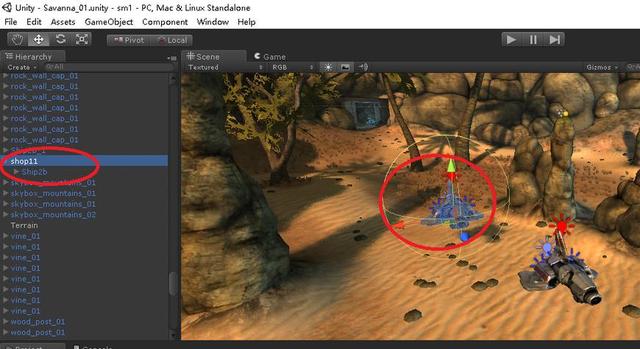
Unity 四元数 quaterion

<https://baijiahao.baidu.com/s?id=1594555750467206968&wfr=spider&for=pc>供参考

1.Unity中的坐标轴 上轴：Y轴 前轴：z轴 纵轴：x轴

摘录：

实际上更多的做法是先建立一个空物体，然后把模型拉成空物体的子级（见下图），然后在设计期间在里面旋转调整模型的角度就可以了，而外层空物体当做正常模型来使用就可以了。让它直接 lookat 目标就行，不用两次代码进行调整：

空物体做父级可轻松解决导出模型轴问题

大家肯定松了一口气。但这还不算完，如果是操作摄像机那没什么好说的，如果是操作模型，直接转向是很生硬的，通常会加上一个渐变的过程，所以具体的代码实际上要变成正面这样：

ship1.transform.rotation = Quaternion.Slerp(ship1.transform.rotation,Quaternion.LookRotation(target.transform.position - ship1.transform.position), //两个位置在前后是不一样的，原物体位置在前是最好的//还可以这样在最后的结果上再做调整，两个角度相乘就是角度叠加了//Quaternion.LookRotation(target.transform.position - ship1.transform.position) \* Quaternion.Euler(new Vector3(90, 0, 0)), //欧拉角度转 Quaternion, //两个位置在前后是不一样的，原物体位置在前是最好的//10 \* Time.deltaTime);2.5f \* Time.deltaTime);//速度//transform.Rotate(0, 0, 90);//ok//不行，很抖动

具体的大家可以查 Quaternion.Slerp() 函数的用法，总的来说它就是在两个角度状态之间根据不同的速度计算出二者的渐进值。要特别说明的是原物体的位置最好作为第一个参数，要不可能是没有渐变平滑效果出来的。

3.利用四元数实现fps 的鼠标控制镜头转向