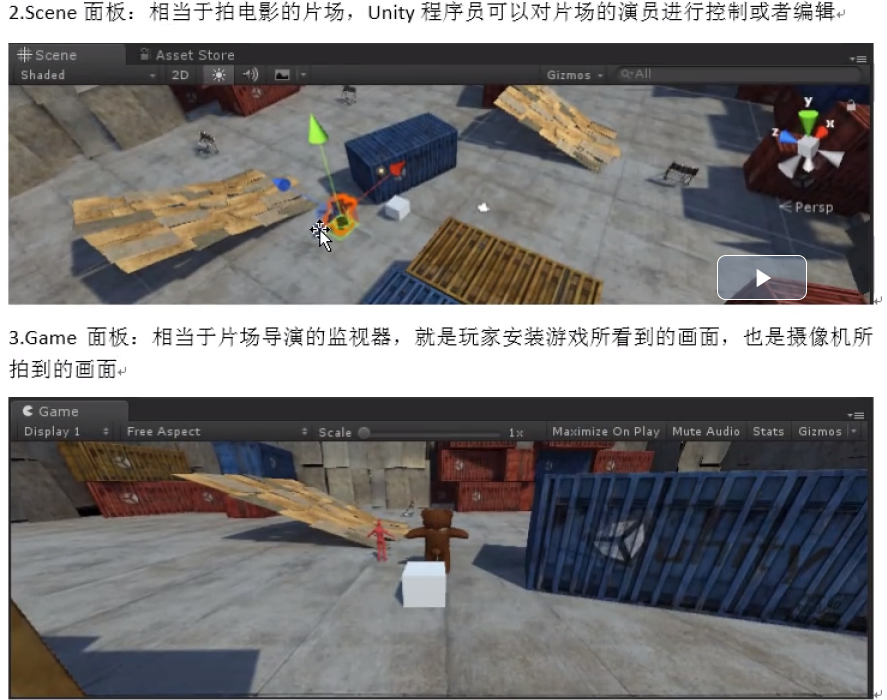
# 面板

## Scene



## Game

## Hierarchy



## Inspector

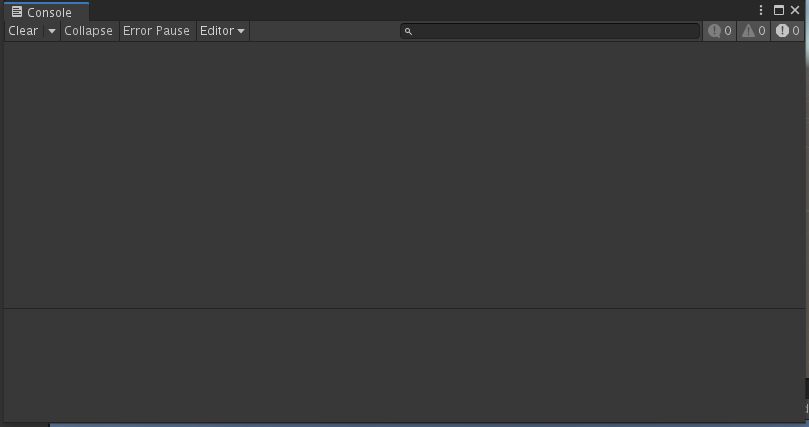


## Project

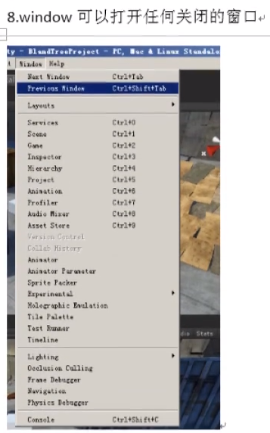


## Console

控制台面板



## Window

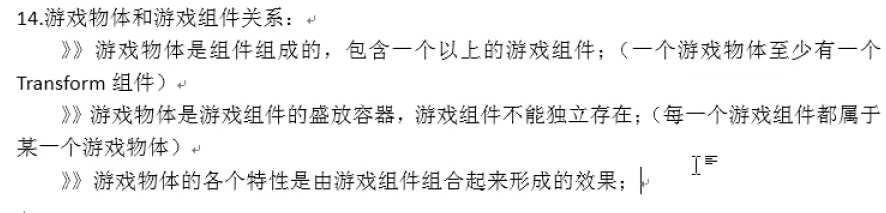


## Center：游戏物体之间的中心位置

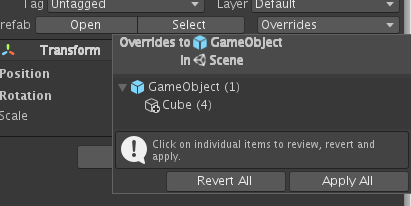
## Global/local：世界坐标系方向/自身坐标系方向

## 调试，暂停，下一帧

# 游戏物体与游戏组件关系



# 预制体⭐



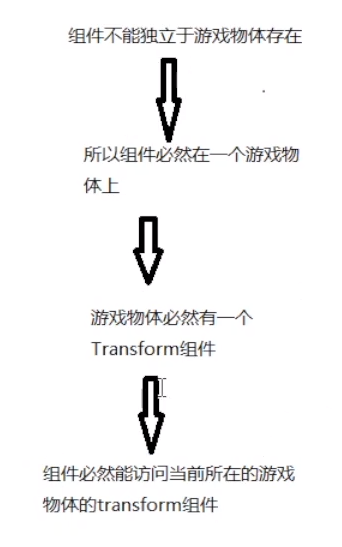
懂得都得

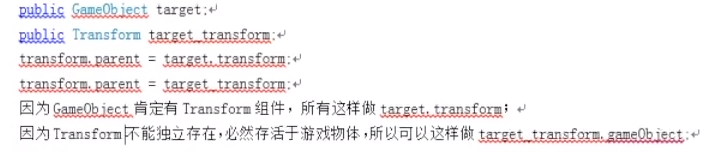
四个按钮

Open/Select/Revert ALL/Apply ALL

不懂找翻译

# 组件





# Transform（F12是个好东西）

## Rotate（转向）

## Translate（移动）

## LookAt（看向）

Debug.DrawLine(画线)

## Parent（变成儿子son）

Setparent

## Find（查找）

transform.Find()查找子物体

GameObject.Find();全局查找

👇

GameObject.FindGameObjectsWithTag("NPC"); 查找为“NPC”的全部 GameObject.FindGameObjectWithTag("NPC");查找最后一个为“NPC”的

---------------------------------------------------------------------------------



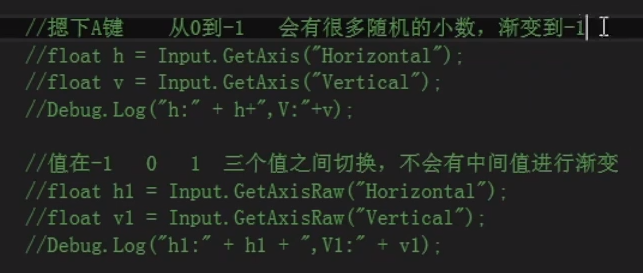
# Input

## GetAxis / GetAxisRaw

GetAxis 趋于0和1或者-1和0 有小数

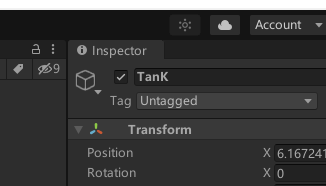
GetAxisRaw 趋于0和1或者-1和0

鼠标指向左右



# 生命周期

## 普通

//第一个执行

private void Awake()

{

Debug.Log("Awake");

}

//启用调用 启动时候出现 方法： gameObject.SetActive(true);

private void OnEnable()

{

Debug.Log("OnEnable");

}

//禁用调用 禁用时候出现 方法： gameObject.SetActive(false);

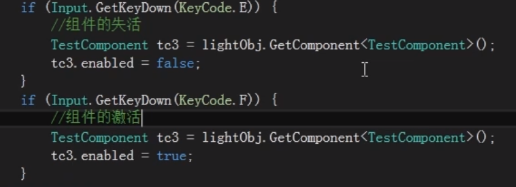
private void OnDisable()

{

Debug.Log("OnDisable");

}

### 组件失活和激活（游戏物体激活失活在↑）



//销毁 方法： Destroy(NAME)

private void OnDestroy()

{

Debug.Log("OnDestroy");

}

// Start is called before the first frame update

//这个是默认……

void Start()

{

}

## 实时

//每帧执行

void Update()

{

Debug.Log("普通Update");

}

//物理更新写在这

private void FixedUpdate()

{

Debug.Log("FixedUpdate");

}

//执行完Update后执行 慢点

private void LateUpdate()

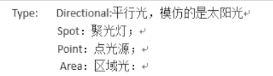
{

Debug.Log("晚一点的LateUpdate");

}

# 灯光组件

## Type



## Mode



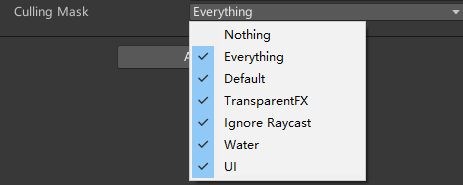
## 光线



## 影子

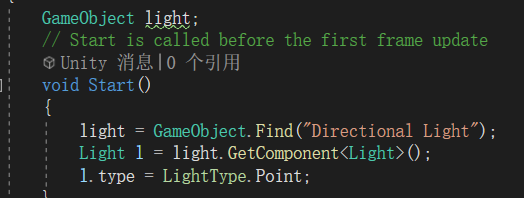


对哪一层有效

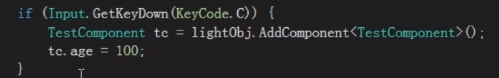




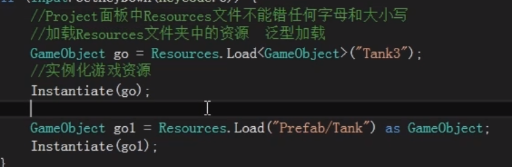
## 获取组件



## 添加组件



# Resources加载资源（重要）



# GUI

## OnGUI（原）

//显示

GUI.Label(new Rect(50, 50, 100, 100), content);

//按钮

if (GUI.Button(new Rect(250, 50, 100, 100), "cao"))

{

content = "伞兵";

}

//切换

Switch = GUI.Toggle(new Rect(500, 50, 100, 100), Switch, "gan");

if (Switch)

{

content = "你是伞兵";

}

else {

content = "你不是伞兵";

}

//拉杆

VerticalValue = GUI.VerticalSlider(new Rect(500, 50, 100, 100), VerticalValue, 100, 0);

HorizontalValue = GUI.HorizontalSlider(new Rect(500, 200, 100, 100), HorizontalValue, 100, 0);

content = VerticalValue + "|" + HorizontalValue;

//可以变宽的拉杆

VerticalValue = GUI.VerticalScrollbar(new Rect(500, 50, 100, 100), VerticalValue,0.5f, 100, 0);

content = VerticalValue.ToString();

//选择工具

select = GUI.Toolbar(new Rect(500, 200, 100, 100), select, new string[] { "aa", "bb", "cc" });

content = select.ToString();

#### ！

对象在镜头的坐标系



# Time

Time.time 运行时间

Time.deltaTime 每帧运行时间

Time.fixedTime fixed fixe时间

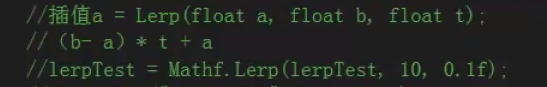
Time.timeScale 时间缩放

Time.unscaledTime 不受暂停和时间缩放影响

# Mathf类

Mathf.Clamp()取值范围

Mathf.Lerp() 插值运算 可以更平滑移动



Mathf.SmoothDamp () 比lerp更细腻



# Vector3

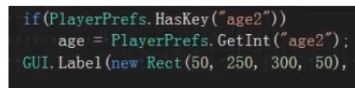
Normalized向量

# PlayerPrefs（存数据）

存



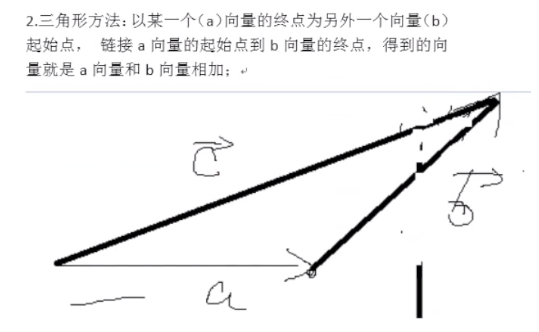
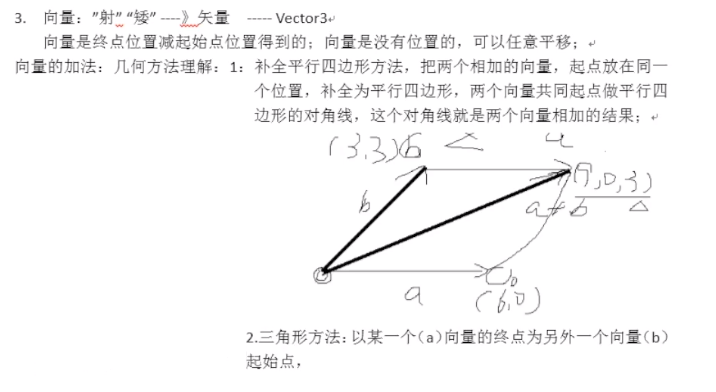
取



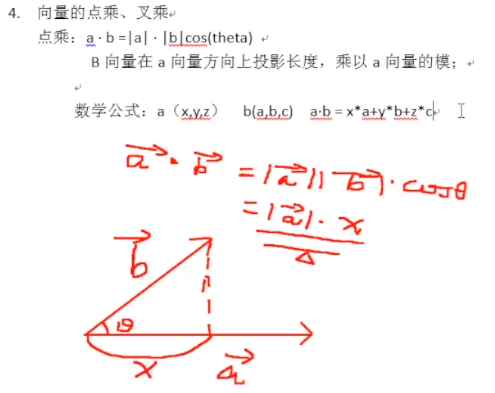
# 坐标系



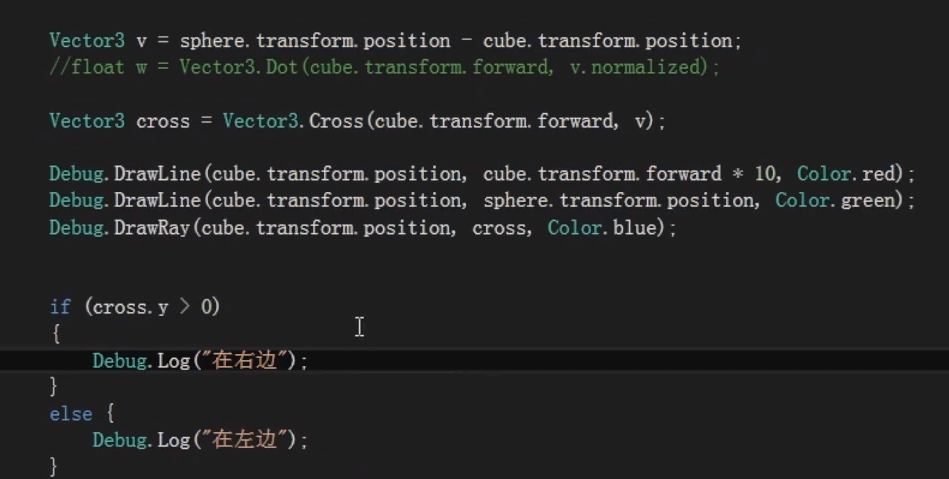
## 向量



### 点乘

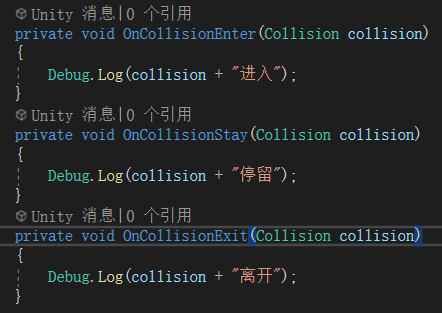


### 叉乘

****

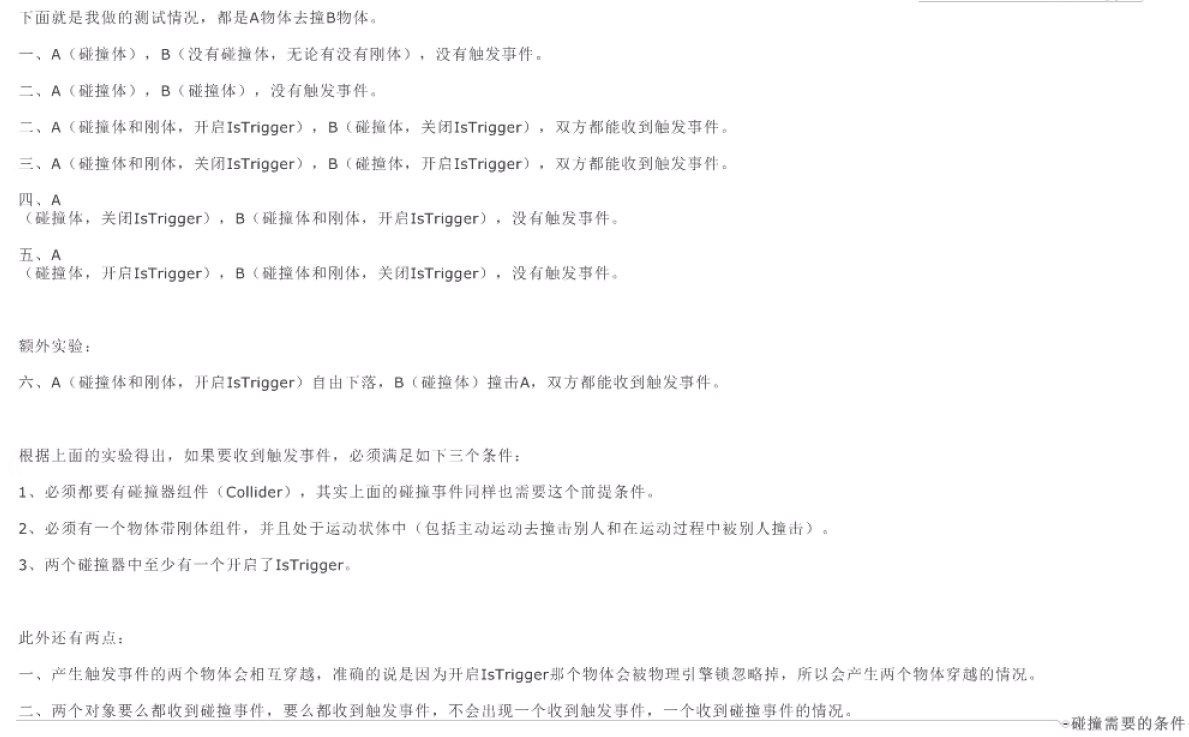
# 碰撞 与 触发器

## 碰撞器



## 触发器

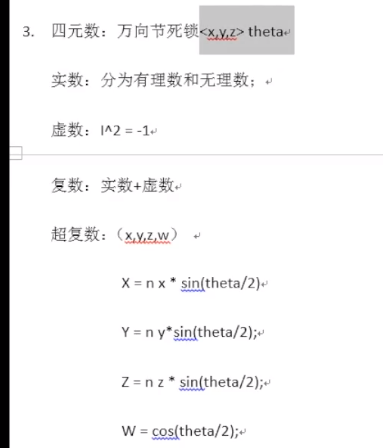
 触发器勾选 可产生触发器 自身不是刚体可以碰撞刚体触发 有穿透效果

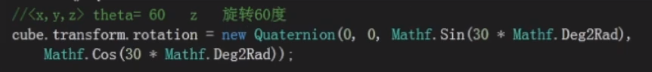


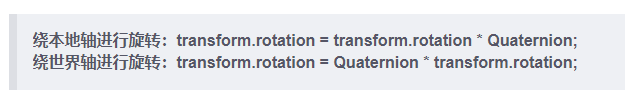


# 

# 四元数

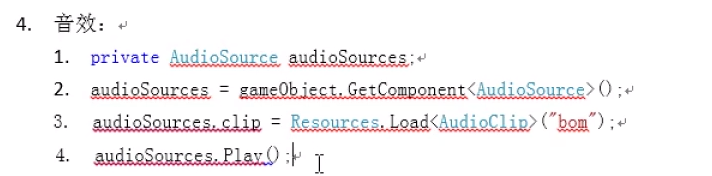




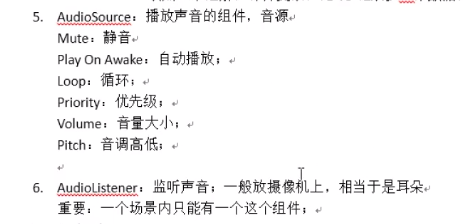


**只有乘法顺序决定了旋转轴属于世界坐标系还是物体坐标系。**

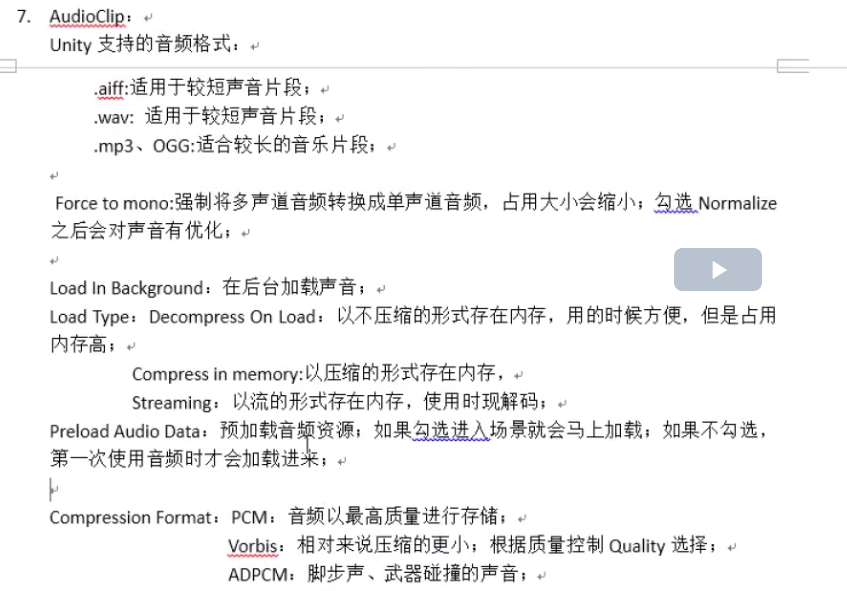
# 音效（Audio）



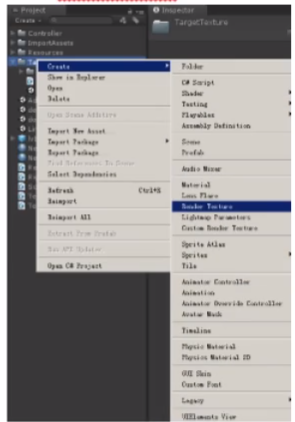
## AudioSource和AudioListener



## AudioClip

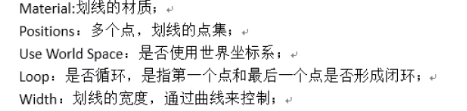


# 创建RenderTexture



创建一个新摄像机 给摄像机Tatget Texture 赋值RenderTexture 摄像机的Depth就会失效

# LineRenderer划线组件



# 力 AddForce

# 打包



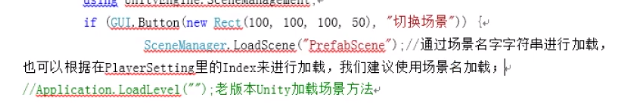
宏定义Scripting Define Symbols

用于不同型号的手机之类的

SDK JDK（环境变量）

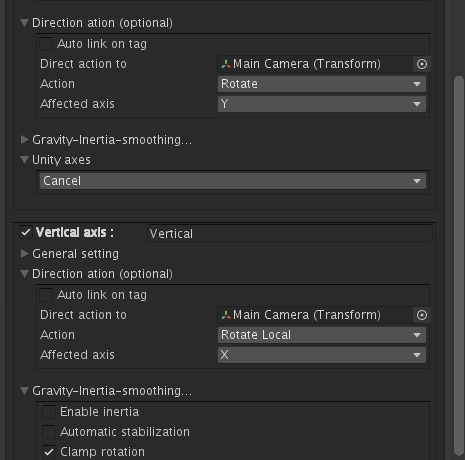
# 加载场景方法





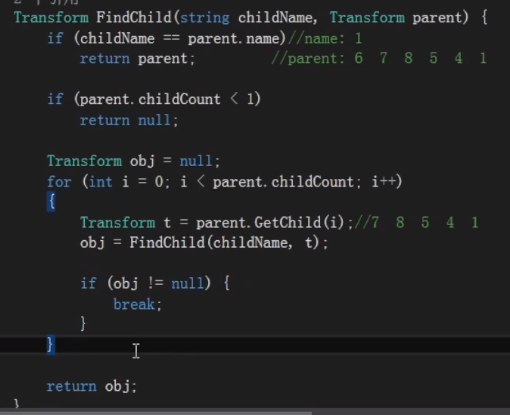
# 小tips

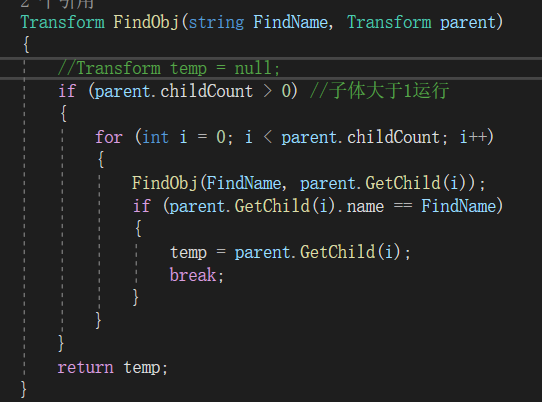
Camera.main代表 tag 设为“MainCamera”的摄像机的 Camera 组件的引用。



这个一个是自身转一个是世界转

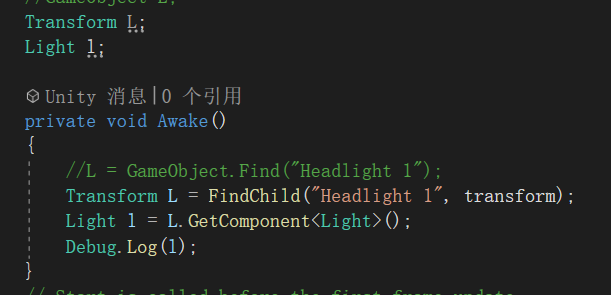
# 自己写的和别人写的





# 耻辱bug

因附了个类， 导致null 初步估计是在Awake定义了 外面没用着

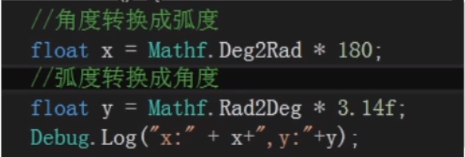
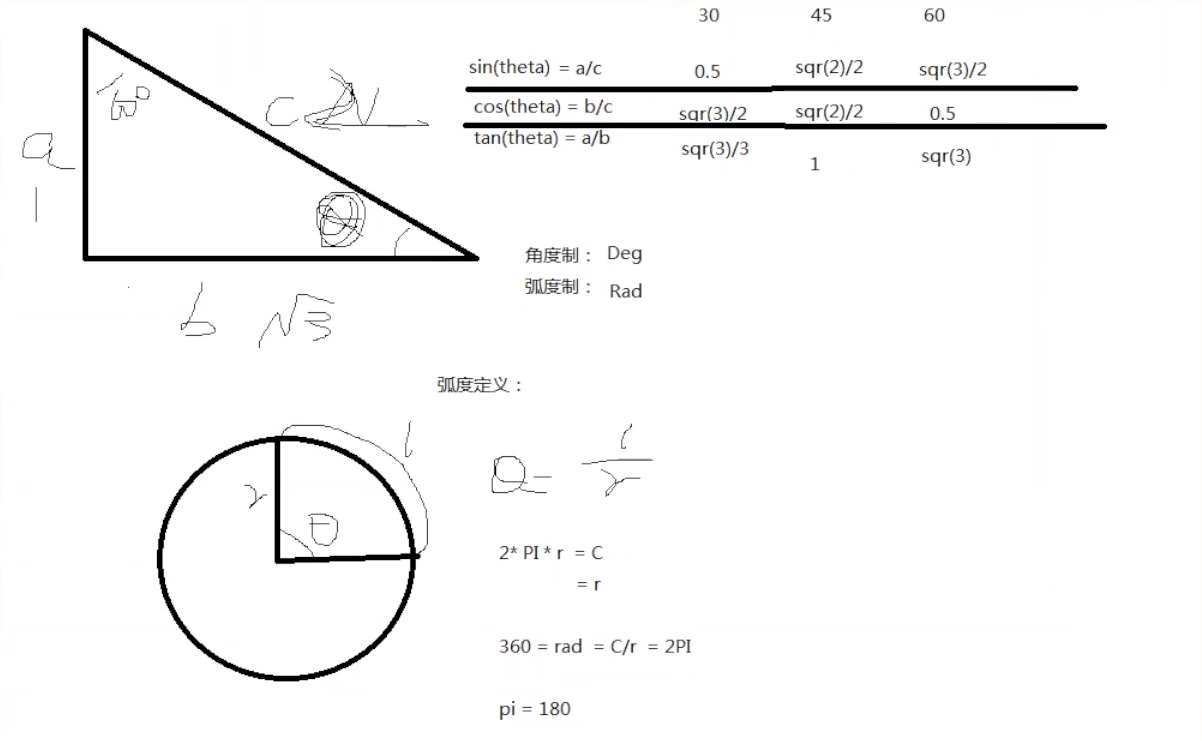




删除前面 赋值全局变量



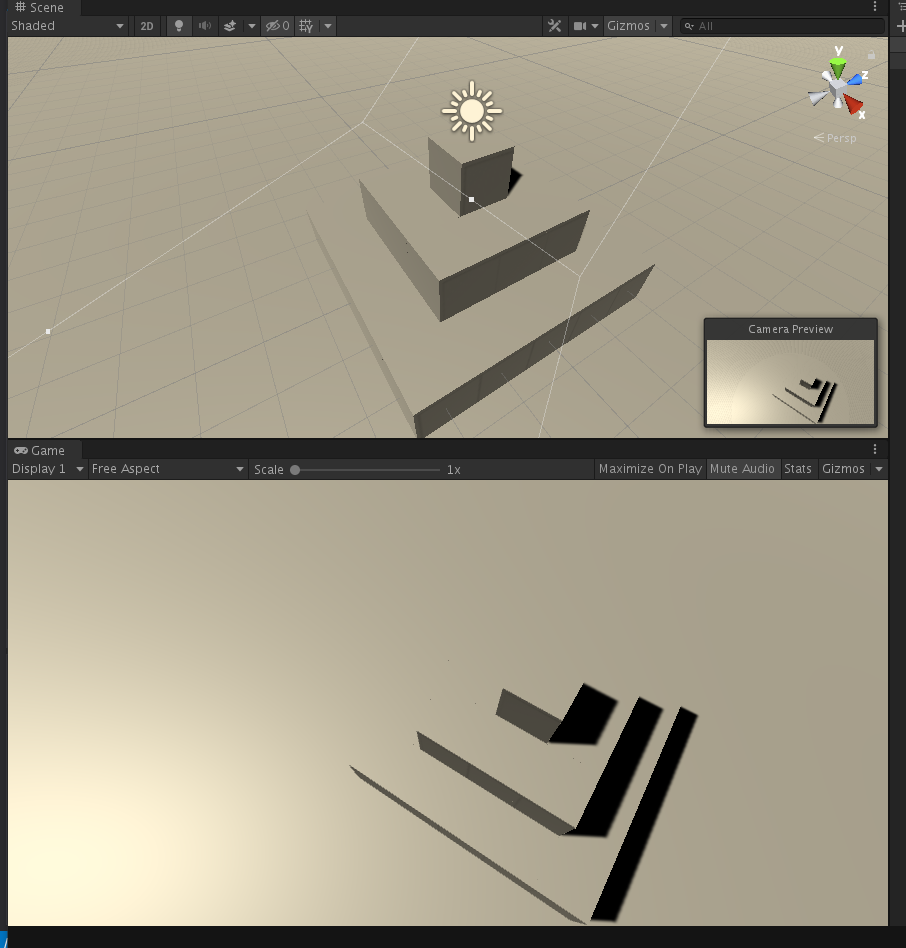
# 有待探讨

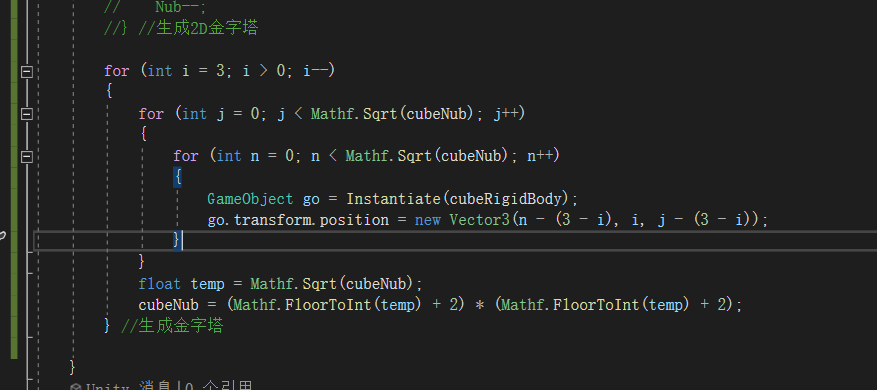


# 多多观看

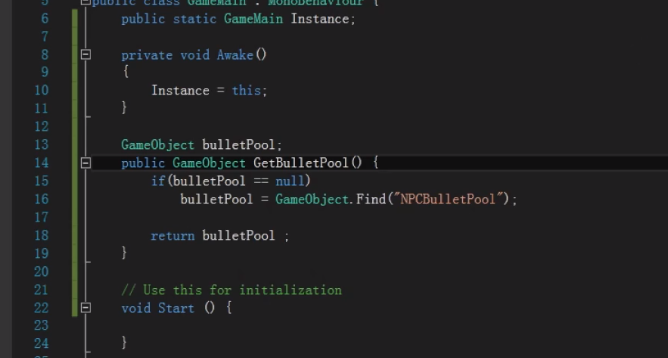


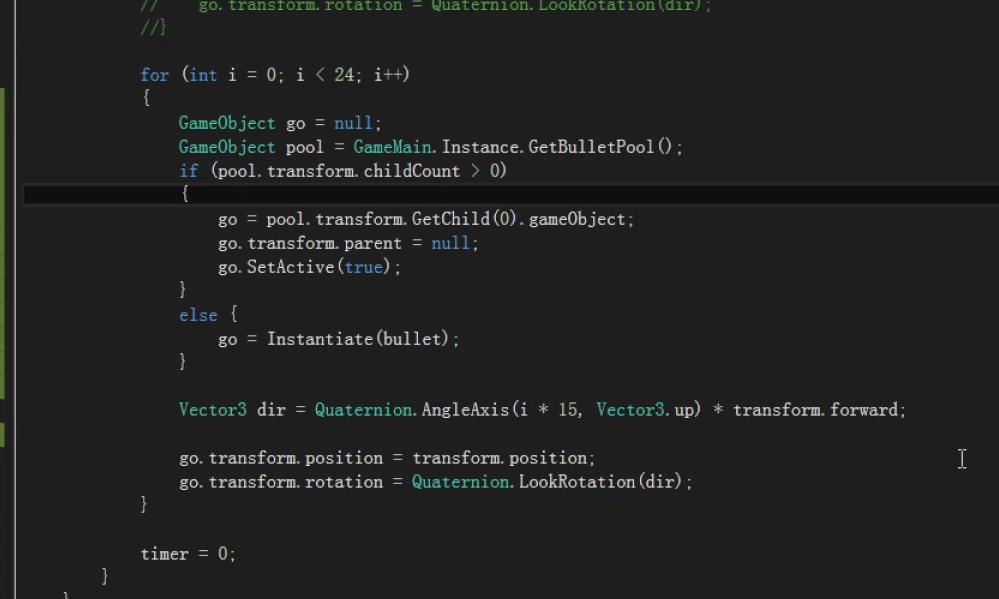
# 自己觉得不错的写法

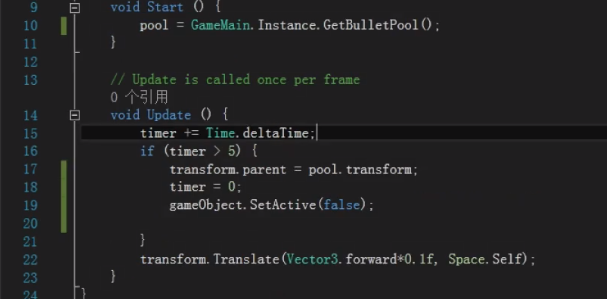


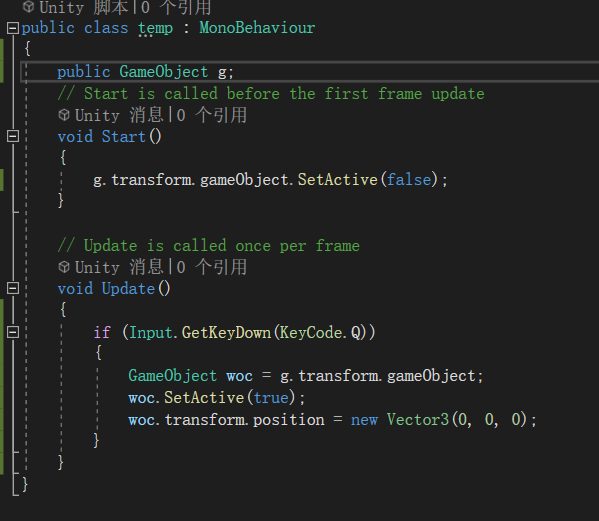


## 缓冲池









挺好玩 发现这可以让物体赋予到新物体

## 太阳日落 Slerp

