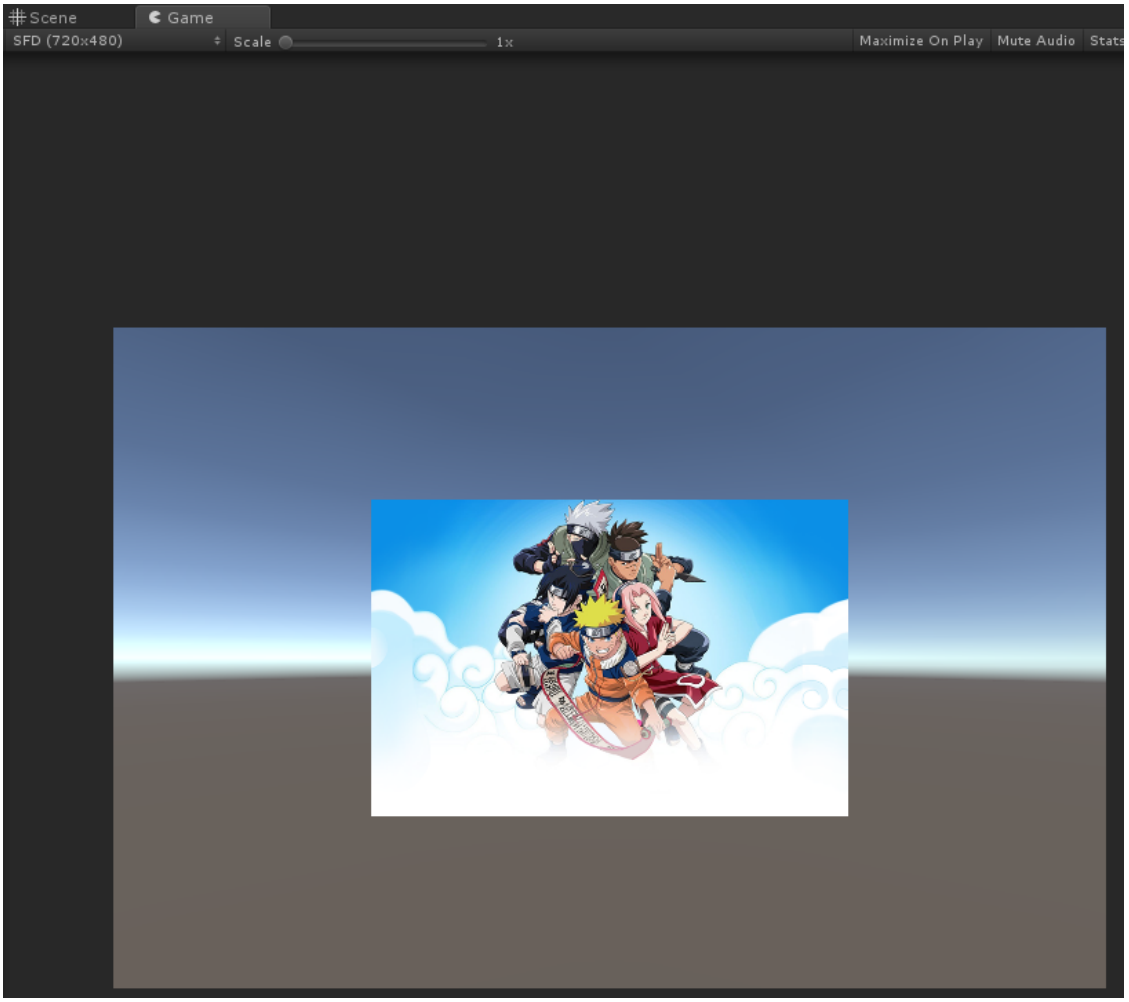
在2D游戏开发中

摄像机的Size=MaxHeight/2/PixelsPerUnit（100）

Unity的屏幕适配都是基于高度的

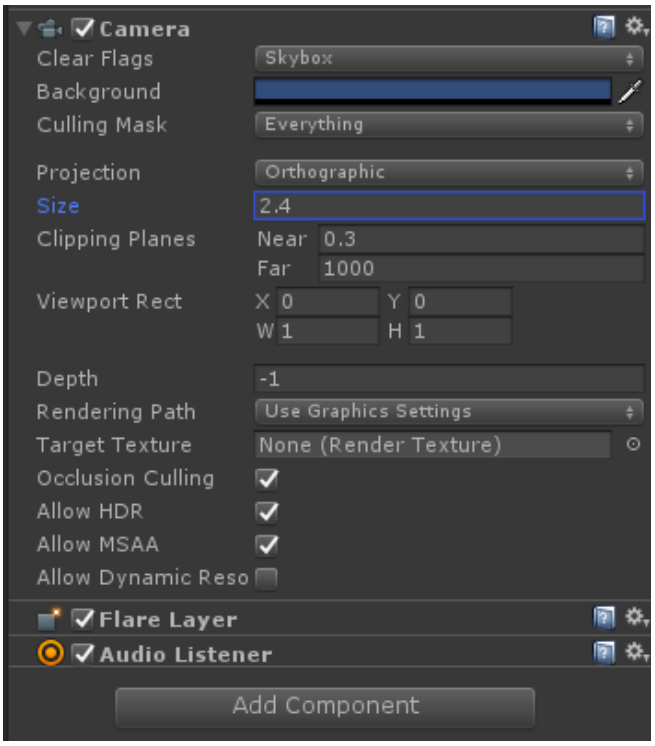
在不同分辨率下的设备 控制显示效果 也应该通过调整摄像机的Size来实现

当屏幕分辨率和图片分辨率完全一致时 显示效果不准确



当前设计分辨率为720*480

根据公式 摄像机的Size=480/2/100=2.4



开始场景

主相机绑脚本：

```
1  using System.Collections;
```

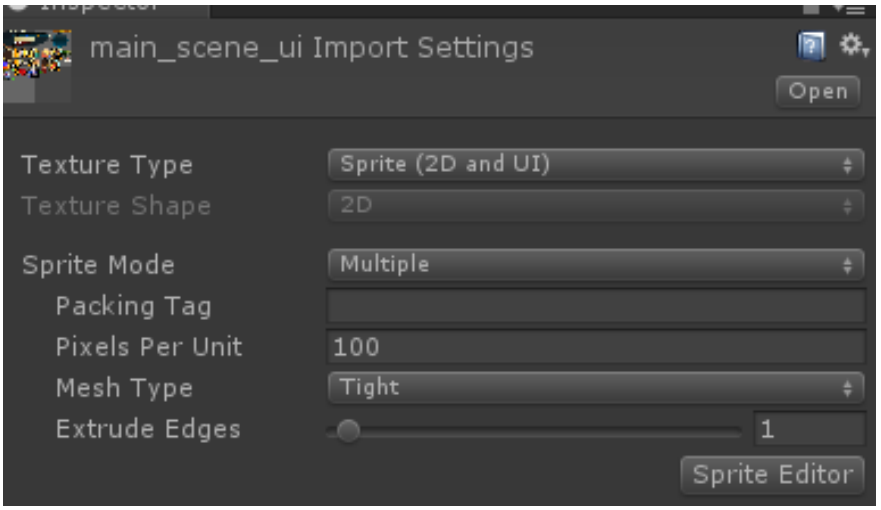
```
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.SceneManagement;

public class GameStartUI : MonoBehaviour {

    public GUISkin skin;
    void OnGUI()
    {
        //设置GUI皮肤
        GUI.skin = skin;
        if (GUI.Button(new Rect(50, 80, 100, 50), "进入游戏"))
        {
            //加载游戏场景
            SceneManager.LoadScene("Game");
        }
        if (GUI.Button(new Rect(50, 150, 100, 50), "继续游戏"))
        {
            //读取存档数据
        }
        if (GUI.Button(new Rect(50, 220, 100, 50), "退出游戏"))
        {
            //Unity编辑器模式下的退出方法
            UnityEditor.EditorApplication.isPlaying = false;
            //退出应用程序的函数
            //Application.Quit();
        }
    }
}

```

2D多合一图片的导入 SpriteMode选择Multple 使用SpriteEditor切割



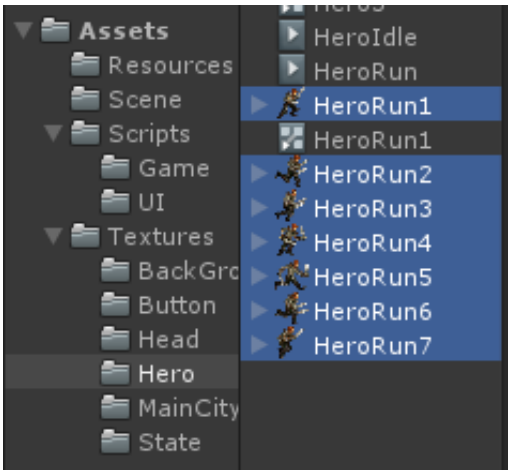
SpriteEditor 使用Slice进行多图切割 默认锚点设置在中心 可以手动修改 比如血条一般锚点设置在Left

主城场景

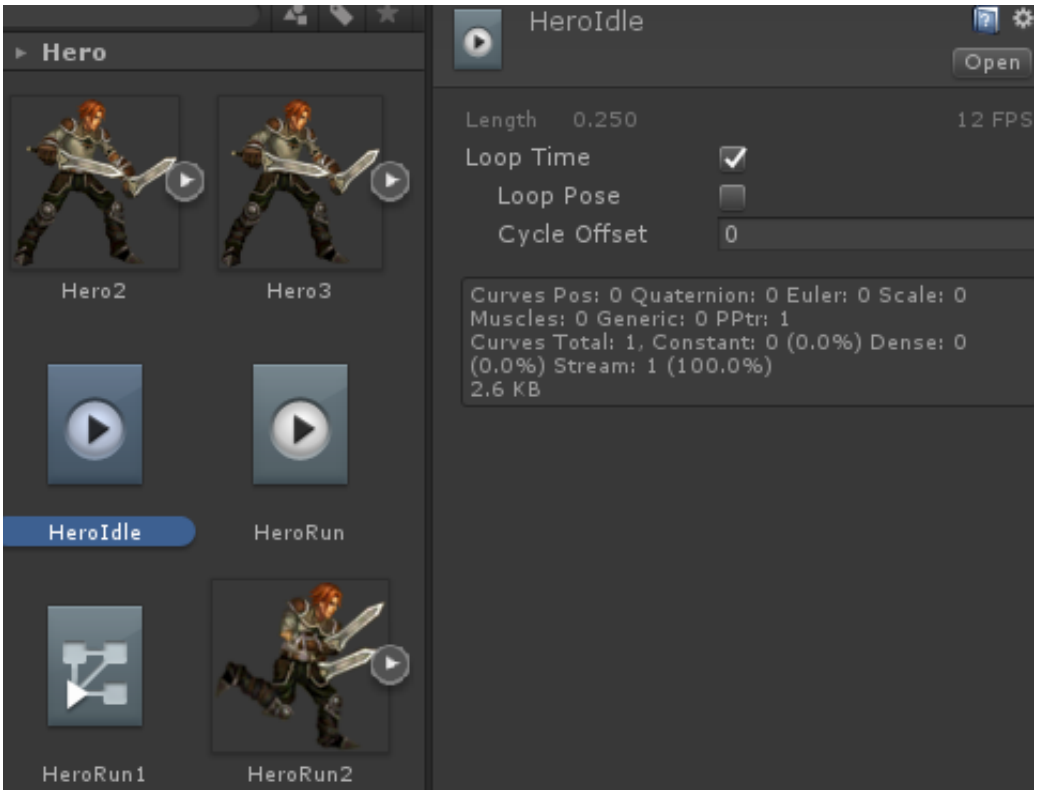
\

2D序列帧动画

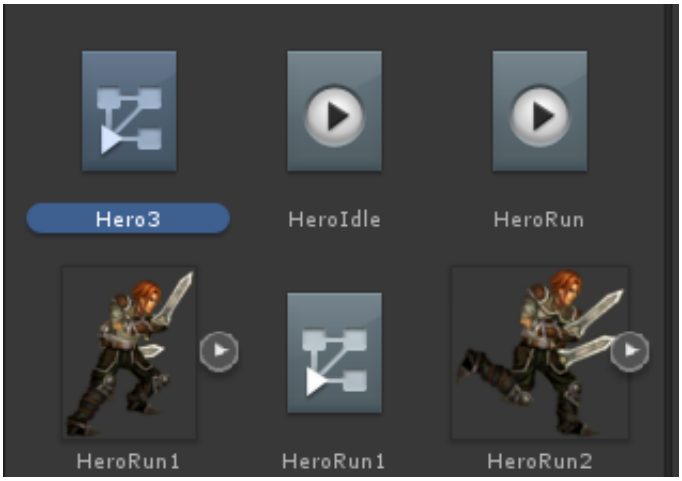
制作办法 把一个动画的多个序列帧图片拖入场景 Unity会自动生成动画文件



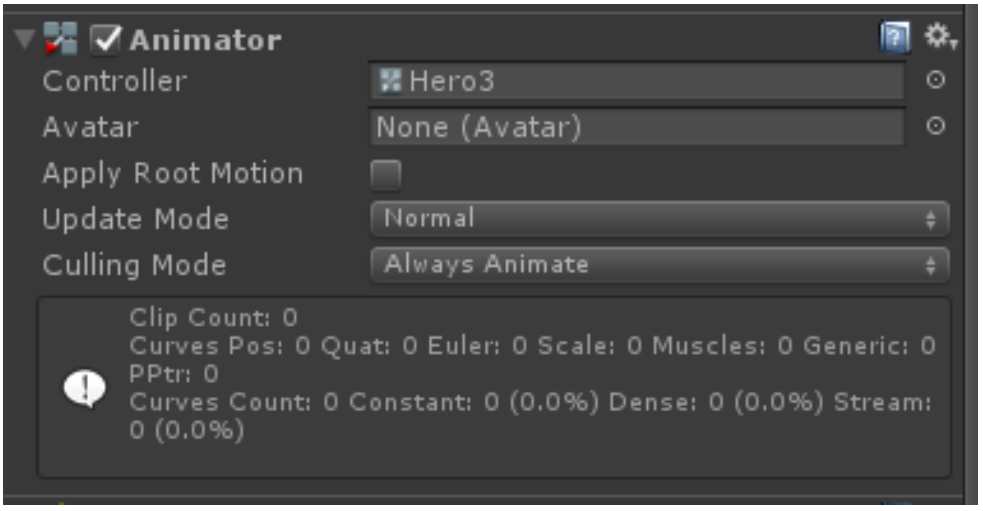
HeroIdle和HeroRun就是自动生成的动画 LoopTime勾选代表动画循环



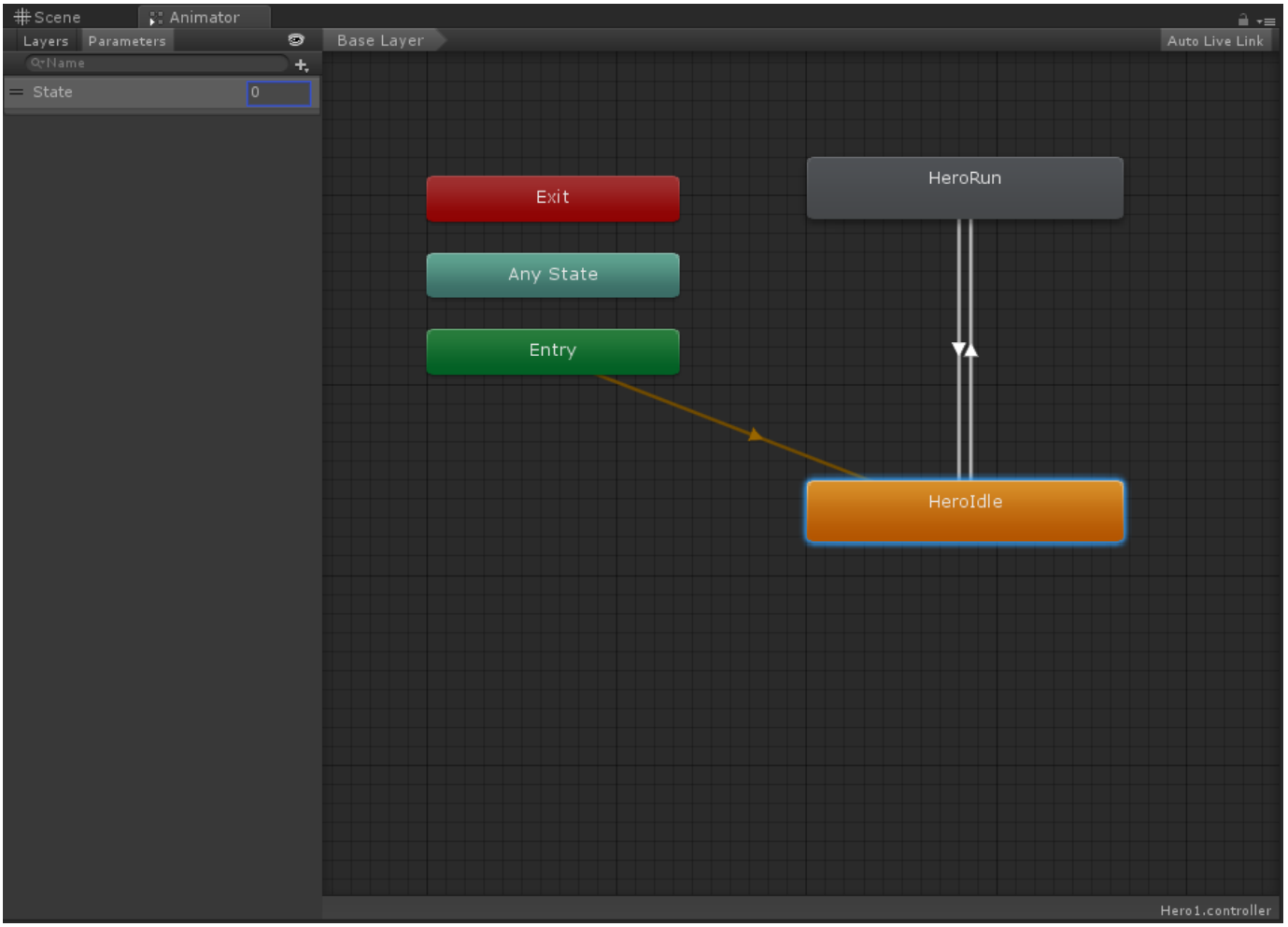
HeroRun1和Hero3是动画控制器



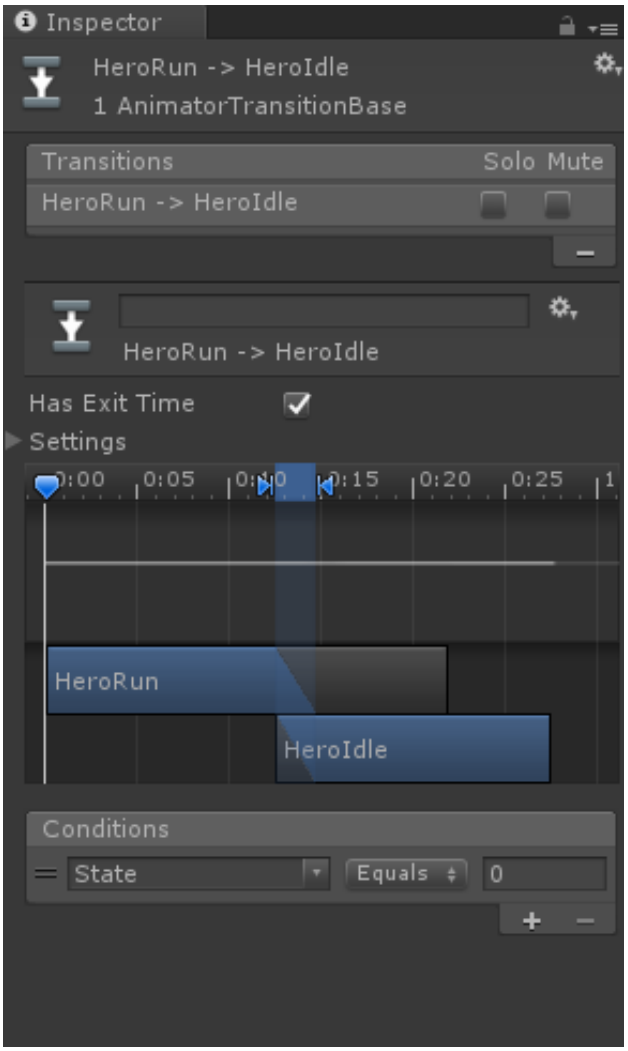
播放动画的物体 添加Animator组件 把生成的动画控制器拖拽给Controller赋值



在Animator控制器中 添加Int类型的State参数



在条件编辑窗口设置参数值 设置过渡时间


Inspector
HeroRun -> HeroIdle
1 AnimatorTransitionBase
Transitions
Solo
Mute
HeroRun -> HeroIdle
HeroRun -> HeroIdle
Has Exit Time
Settings
0:00
0:05
0:15
0:20
0:25
HeroRun
HeroIdle
Conditions
State
Equals
0

主场景：


VIP大礼
地下城
竞技场
任务
技能升级
装备改造
宝藏

人物控制脚本：

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

public class HeroControl : MonoBehaviour {

    public Transform checkPoint;

    private Animator animator;
    private Rigidbody2D body;
    private bool facingRight = true;
    private bool isGrounded;
    private bool jump;
    void Start () {
        animator = GetComponent<Animator>();
        body = GetComponent<Rigidbody2D>();
    }

    void Update()
    {
        isGrounded = Physics2D.Linecast(transform.position,
                checkPoint.position, 1 <<
LayerMask.NameToLayer("Ground"));

        if (Input.GetButtonDown("Jump") && isGrounded)
        {
            jump = true;
        }
    }

    void FixedUpdate() {

        //监听水平轴变化
        float h=Input.GetAxis("Horizontal");

        if (h != 0)
        {
            //播放奔跑动画
            animator.SetInteger("State", 1);
```

```
        }
        else
        {
            //播放站立动画
            animator.SetInteger("State", 0);
        }

        //设置刚体的速度 修改人物位置
         body.velocity = new Vector2(3.0f*h, body.velocity.y);

        //需要往右移动 并且面向左方 需要转身
        if (h > 0 && !facingRight)
        {
            Flip();
        }
        else if (h < 0 && facingRight) //需要往左移动 并且面向右方 需要转身
        {
            Flip();
        }

        if (jump)
        {
            jump = false;
            //刚体添加向上的作用力
            body.AddForce(new Vector2(0f, 250f));
        }

    }

    void Flip()
    {
        facingRight = !facingRight;
        //获取物体的缩放值 把x变符号
        Vector3 scale = transform.localScale;
        scale.x *= -1;
        transform.localScale = scale;
    }
}
```

Unity的音频管理系统

声音分为3D声音和2D声音

3D立体声 声音有近大远小的效果 随着距离衰减

一般用于3D游戏世界 展示真实的音效效果

2D声音 不受距离远近的影响 一般用于2D游戏或背景音乐

声音种类 背景音乐和游戏音效

声音文件格式：ogg mp3 wave等

ogg使用者最多 因为在安卓使用ogg可以绕开mp3的版权问题

wave格式 文件更大 是高保真无压缩的格式

Unity播放音频组件：

1 AudioClip 音频剪辑

不管是什么格式的音频文件 在Unity中都使用AudioClip类型来表示

在Unity运行过程中播放的都是AudioClip

2 AudioListener 音频监听器

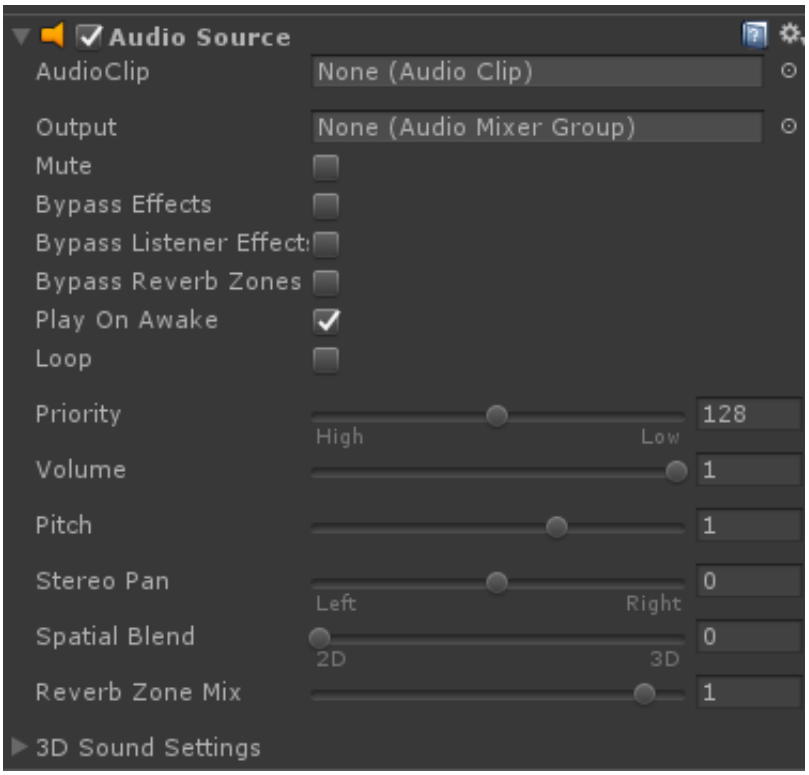
相当于人的耳朵 把耳朵移动到哪里 就可以听到哪里的声音

一个场景中必须存在一个音频监听器 有了它玩家才能听到声音

音频监听器类似于话筒 负责接收声音源 然后把音频数据通过设备的扬声器播放

一般来说场景中只有一个AudioListener即可 默认情况加上主摄像机上

3 AudioSource 声音源



在3D世界中 很多声音需要有衰减的效果 需要计算监听器和声音源的距离

如果需要实现特殊的声音混合 需要使用AudioSource组件 帮助我们播放AudioClip

AudioClip 需要播放的音频文件

OutPut 音频混合器输出

Mute 是否静音

ByPass Effects 声音滤波开关

ByPass Listener Effects 监听器滤波

ByPass Reverb Zones 回音混淆开关

Play On Awake 唤醒时播放

Loop 循环播放

Priority

播放优先级 0最高 256最低 默认128

优先级高的一般是重要的音效

比如 背景音乐设置成0 就会避免它被其他的音乐替换掉

一般设备可以最多允许32个声音源 同时存在 当超过32时 优先级决定了保留顺序

Volume

音量 范围0-1

Pitch

音调

Stereo Pan

立体声声道（-1左声道 1 右声道）

Spatial Blend

空间回合 0 为2D音效 1 为3Dy音效 2D音效无衰减

```
public class A : MonoBehaviour {

    public AudioClip clip;
    public AudioSource source;
    void Start () {

        if (!source.isPlaying)
        {
            //立即播放
            source.Play();
            //延迟3秒开始播放
            source.PlayDelayed(3.0f);
        }
        //修改当前声音源的音频剪辑
        // source.clip = clip;
        // source.Play();

        //播放音效 仅播放一次
        source.PlayOneShot(clip);
        //静态方法 在某个点播放个音效 生成空物体添加Source组件 播放后销毁
        AudioSource.PlayClipAtPoint(clip, transform.position);
    }

    // Update is called once per frame
    void Update() {
        if (Input.GetMouseButtonDown(0))
        {
            //暂停播放
            source.Pause();
        }
        if (Input.GetMouseButtonDown(1))
```

```
        {
            //停止播放
            source.Stop();
            source.mute = true;
            source.volume = 1.0f;
        }
    }
}
```