使用foreach迭代 自定义类型的整形数组中所有的偶数

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

class MyClass
{
    private int[] data;
    public MyClass(int[] data)
    {
        this.data = data;
    }

    public IEnumerator<int> GetEnumerator()
    {
        for (int i = 0; i < data.Length; i++)
        {
            if (data[i] % 2 != 0)
                yield return data[i];
        }
    }
}

public class Text : MonoBehaviour {

    void Start()
    {
        MyClass mc = new MyClass(new int[] { 100, 3, 24, 11, 15, 17,
19, 223, 334 });

        foreach (var it in mc)
        {
            Debug.Log(it);
        }
    }

}

```

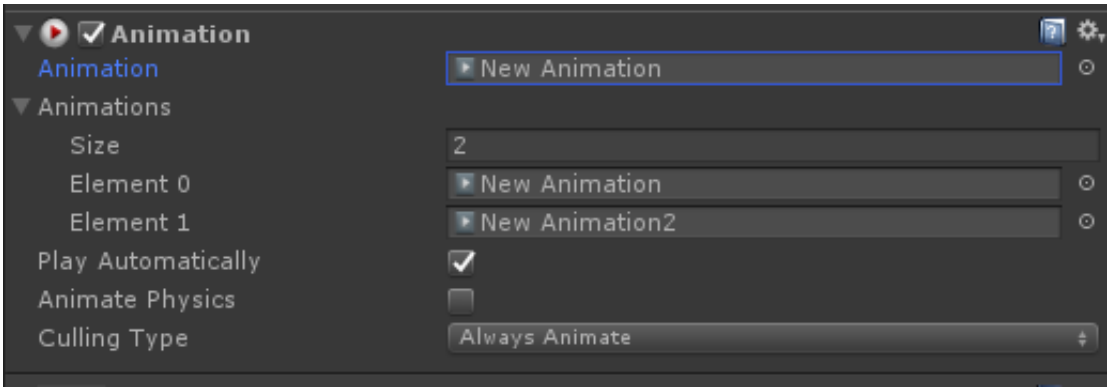
```
38
```

Animation动画系统

Unity最初的动画系统

新版动画 虽有蓝图开发 但对性能要求比较高 消耗较大

开发中一些简单的动画依然可以使用Animation来解决



Animation 默认动画 启动时会自动播放的动画剪辑

Animations 动画列表 可以用脚本访问的动画列表

Play Automatically 自动播放 启动游戏时是否需要自动播放

Animate Physics 动画物理

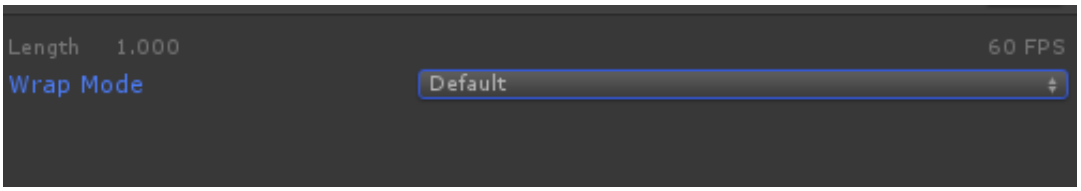
CullingType 动画消隐

1Always Animate

动画消隐被禁用 即使物体在屏幕外也正常播放动画

2 Based On Renderers

当渲染不可见时 动画被禁用



WrapMode 循环模式

Default 默认使用美术提供的循环方式

Once 当时间播放到末尾时停止动画的播放

Loop 当时间播放的末尾时 重新开始播放

PingPong 从开始和结束之间来回播放

ClampForever 当播放到末尾时 总是处于最后一帧的采样状态

动画剪辑 AnimationClip类

使用代码修改动画剪辑的wrapMode

```
public class Text : MonoBehaviour {

    public AnimationClip clip;//动画剪辑类型
    private Animation anim;
    void Start()
    {
        anim = GetComponent<Animation>();
        anim.GetClip("New Animation").wrapMode = WrapMode.Loop;
        anim.Play();
    }

}
```

Animation系统的主要类

1 Animation 动画类 控制动画的播放

2 AnimationClip 动画剪辑类 动画资源的数据类

3 AnimationState 动画状态类 用来记录某一个动画的播放状态的类

4 AnimationEvent 动画事件类 用来给对应的动画剪辑添加帧事件 回调方法

用代码添加动画剪辑

```
public class Text : MonoBehaviour {
```

```
2
3       public AnimationClip clip;//动画剪辑类型
4       private Animation anim;
5       void Start()
6       {
7           //通过代码添加需要的动画资源文件到Animation组件中
8           anim = GetComponent<Animation>();
9           anim.AddClip(clip, "Attack");
10      }
11
12  }
```

使用Animation控制动画的播放

```
1   using System.Collections;
2   using System.Collections.Generic;
3   using UnityEngine;
4   using System;
5
6   public class Text : MonoBehaviour
7   {
8       public AnimationClip clip;//动画剪辑类型
9       private Animation anim;
10      void Start()
11      {
12          //通过代码添加需要的动画资源文件到Animation组件中
13          anim = GetComponent<Animation>();
14          anim.AddClip(clip, "Attack");
15          anim.Play("Attack");//播放动画  参数:需要播放的动画名
16          Invoke("Fun", 0.5f);
17      }
18
19      void Fun()
20      {
21          //isPlaying判断动画是否正在播放
22          if (anim.isPlaying)
23              anim.Stop();//停止当前正在播放的动画
24      }
25      void Update()
26      {
```

```
        if (Input.GetMouseButtonDown(0))
        {
            //0.2f的时间淡入淡出的方式播放run动画
            anim.CrossFade("run",0.2f);
           //anim.player("run");
        }

    }

}


```

Unity3D中的动画帧事件

1 直接在编辑器窗口AddEvent 选择对应回调方法和传参 但是传参不方便 代码中的数据没法传递

2 在代码中添加事件

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

public class AnimText : MonoBehaviour {

    Animation anim;
    void Start () {
        anim = GetComponent<Animation>();
        AnimationClip clip= anim.GetClip("New Animation");

        //创建动画事件
        AnimationEvent animEvent = new AnimationEvent();
        animEvent.functionName = "Fun222";//回调方法名
        animEvent.intParameter = 100;//方法参数
        animEvent.time = clip.length/2;//方法调用的时间节点
        clip.AddEvent(animEvent);//添加动画事件到动画剪辑

    }
```

```
20
21      void Fun222(int a)
22      {
23          Debug.Log(a);
24      }
25  }
26
```

练习：

要求为游戏物体制作简单的跳跃且落下的动画

第一个物体跳跃到最高点 下一个物体开始跳跃 以此类推

当最后一个物体跳跃到最高点时触发最初第一个物体的跳跃

```
1   using System.Collections;
2   using System.Collections.Generic;
3   using UnityEngine;
4   using System;
5
6   public class Text : MonoBehaviour
7   {
8       public GameObject nextObj;
9       private Animation anim;
10      void Start()
11      {
12          anim = GetComponent<Animation>();
13          AnimationClip clip = anim.GetClip("Sphere1");
14          AnimationEvent aEvent = new AnimationEvent();
15          aEvent.time = clip.length / 2;
16          aEvent.functionName = "Function";
17          clip.AddEvent(aEvent);
18      }
19
20      void Function()
21      {
22          nextObj.GetComponent<Animation>().Play("Sphere1");
23      }
24  }
```

AnimationState 动画状态

```
public class Text : MonoBehaviour
{
    private Animation anim;
    void Start()
    {
        anim = GetComponent<Animation>();
        AnimationState state = anim["Sphere1"];
        state.speed = 4.0f;
        state.wrapMode = WrapMode.Loop;
        anim.Play("Sphere1");
    }
}
```

AnimationState主要用于控制动画的混合

多数情况下 Animation已经足够使用 如果需要完全控制混合就必须使用AnimationState

enable 是否启用/禁用动画

WrapMode 循环模式

time 当前动画时间

normalizedTime 动画当前规范化时间

speed 动画的播放速度 默认是1

normalizedSpeed 规范化速度

Length 动画时长 动画剪辑的时间 单位秒

clip 此动画状态播放的剪辑对象

*layer 动画层 计算混合动画时 layer越高则优先获取权值

*weight 动画的权重

name 动画名字

Animation实现两个layer的动画同时播放

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

public class Text : MonoBehaviour
{
    private Animation anim;
    void Start()
    {
        anim = GetComponent<Animation>();
        anim["Sphere1"].layer = 0;
        anim["Sphere2"].layer = 2;
        anim["Cub1"].layer = 2;
        anim.Play("Sphere1");
        anim.Play("Sphere2");
    }

    void Update()
    {
        if (Input.GetMouseButtonDown(0))
            anim.Play("Cub1",PlayMode.StopAll);
    }
}
```

实现动画的倒播

```
public class Text : MonoBehaviour
{
    private Animation anim;
    void Start()
    {
        anim = GetComponent<Animation>();
        AnimationState state= anim["Sphere2"];
        state.normalizedTime = 1;
        state.speed = -1;
```

```
        anim.Play("Sphere2");
    }
}
```

人物模型动画一般是美术在三维软件里设计好 再导入到UNITY中

默认支持的软件格式是:FBX

常用的模型格式：FBX,MAX,MB,3DS,OBJ

FBX文件优势

1 跨软件 可以作为中间格式互转 可以被大部分建模软件导入和导出

maya 3dsmax,softimage mudbox motionBuilder

2 大量的SDK 使用者众多

3 跨平台 便于Unity开发者使用

4 免费SDK

5 多记录支持：网格，材质，动画，骨骼都可以记录在FBX里

控制人物的移动和站立

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

public class Text : MonoBehaviour
{
    Animation anim;
    CharacterController cc;
    float h;
    float v;
    Vector3 direction;

```

```
    void Start()
    {
        anim = GetComponent<Animation>();
        cc = GetComponent<CharacterController>();
        anim["run"].speed = 2.0f;
    }

    void Update()
    {
        h = Input.GetAxis("Horizontal");
        v = Input.GetAxis("Vertical");
        direction.Set(h, 0, v);

        if (direction == Vector3.zero)
        {
            anim.CrossFade("attack_idle", 0.1f);
        }else
        {
            anim.CrossFade("run", 0.1f);
            //把基于摄像机的本地坐标 转化为 世界坐标

 direction=Camera.main.transform.TransformDirection(direction);
            direction.y = 0f;
            //让transform的forward向量 和 direction保持一致
            transform.rotation=Quaternion.LookRotation(direction);
        }
        cc.SimpleMove(direction * 3.0f);
    }

}
```