# Unity基础整理汇总

## C#基础知识

### ArrayList和List和Array

* 1. 数组Array
     1. 优点：连续储存 因此索引速度快 赋值和修改元素简单 时间复杂度为O(1),
     2. 缺点：因为是分配在连续内存所以要提前确定大小 空间的连续导致了存储效率低 插入和删除元素效率低。

定义数组 一定要申请空间 指明数组长度 只能存储同意中数据类型。

int[] tempArray = new int[4]

### Sealed关键字

Sealed密封，禁止的意思，在c#中修饰类和方法。不能与abstract对象一起使用。修饰方法时，表示该方法不能被重写，使用的是在方法继承自基类，然后不想这个方法被重写就可以使用。

### 抽象方法与虚方法（abstract，virtual）

抽象方法只能在抽象类中定义，存在。而虚方法可以在抽象类中存在也可以在普通类中存在。抽象方法不会再抽象类中实现，而虚方法必须在类中实现。抽象方法必须在派生类中重写，虚方法不必。

## Unity基础知识

### 协程

在主程序之外额外创建一段逻辑，来协助主程序的运行，并发执行同时只能存在一个协程。

1.1协程开启方式：

通过StartCoroutine（“方法名”），但最多只能传递一个参数。通过携程开启void方法特别消耗内存，一般不推荐

1.2通过StartCoroutine（IEnumerator routine）使用，可以传递多个参数。推荐此方法

1.3返回值等待说明

//yield return X; 数字(0除外的数字) 程序运行到这里，会等待1帧后执行

//yield return new WaitForSeconds(X); 等待X秒后执行

//yield return new WaitForFixedUpdate (); 等待FixedUpdate ()执行完后执行0.02f

//yield return new WaitForEndOfFrame (); 等待程序渲染和GUI中的内容完成后执行

//yield return StartCoroutine (); 本协程到这里停止 开启下一个协程，直至开启的协程关闭或运行结束后，则重新开始当前协程

//yield break; 直接结束当前协程

1.4关闭协程

//1.StopCoroutine(string menthod)只能关闭通过StartCoroutine(”方法名“)开启的协程

//2.StopAllCoroutines 关闭所有协程

### 碰撞器

1. 分类

1.盒子碰撞器2.球体碰撞器3.胶囊碰撞器4.网格碰撞器5.车轮碰撞器6.地形碰撞器（后面几个看看）

1. 碰撞方法

void OnCollisionEnter(Collision g) //开始碰撞

{

Debug.Log ("开始碰撞"+g.gameObject.name);

}

void OnCollisionStay(Collision g) //碰撞中

{

Debug.Log ("碰撞中"+g.gameObject.name);

}

void OnCollisionExit(Collision g) //碰撞结束

{

Debug.Log ("碰撞结束" +g.gameObject.name);

}

1. 触发器方法

首要条件先在collider中勾选中IS Trigger

void OnTriggerEnter(Collider g)

{

Debug.Log ("触发开始" + g.gameObject.name);

}

void OnTriggerStay(Collider g)

{

Debug.Log ("触发中" + g.gameObject.name);

}

void OnTriggerExit(Collider g)

{

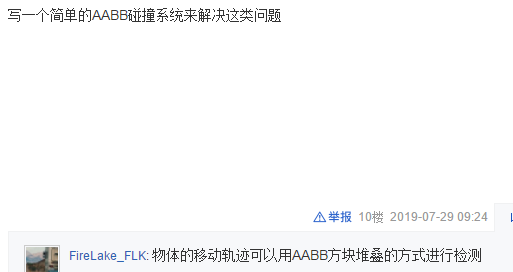
Debug.Log ("触发结束" + g.gameObject.name);

}

1. 问题

碰撞器有时候会检测不到（速度过快检测不到。碰撞检测是Unity自身每一帧对相碰撞的物体进行检测的过程，但要是相碰撞的物体速度过快那么这一帧就会检测不到）。

解决方案：4.1记录前一帧物体的位置，从前一帧的位置向当前帧的位置发一条射线检测是否碰撞。4.2帧率调高，Edit-projectsetting-Time 里面有帧率。4.3

4.4

### Rigibody属性

1. 属性

1.1Mass质量 默认为1，大部分物理的Mass为0.1~10float

1.2Drag 阻力 任意方向都会收到阻力 float

1.3Angular Drag 旋转阻力 float

1.4Use Gravity 使用重力

1.5Is Kinematic 是否遵循动力学

1.6 Interpolate 差值方式

1.7 Collision Delection 碰撞检测模式 防止速度过快而产生击穿现象

1.8Constrains 约束条件 冰冻位置和旋转

1. 使用方法

