**用C#写脚本不可不知道的秘密**

Posted on 2013年04月01日 by U3d / [Unity3D 基础教程](http://www.unitymanual.com/category/manual/unity3d-%e5%9f%ba%e7%a1%80%e6%95%99%e7%a8%8b)/被围观 479 次

C#写[**Unity3D脚本**](http://www.unitymanual.com/category/script)需要注意，除了语法外，使用C#或Boo会有一些差别，最明显的是：

1. 继承之MonoBehaviour类

所有的行为脚本代码必须继承之MonoBehaviour类（直接或间接）。如果使用的是javascript的话会自动隐性的继承，如果使用的是 C#或Boo就必须明确地指定其继承于MonoBehaviour。如果你是在u3d中通过“Asset->Create->C Sharp Script/Boo Script”来创建了脚本代码文件的话，Unity3D的脚本创建模板将会提前将相关继承语句定义在脚本代码文件中。

2.使用Awake或Start方法进行初始化。

你需要在C#或Boo在使用Awake或Start方法。

Awake和Start之间的区别在于：Awake是当一个场景调入过程完成后会自动运行，而Start则是会在Update或FixedUpdate方法被第一次调用之前被运行。所有的Awake方法运行的优先级会高于任意的Start方法。

3. 文件中主类名必须与文件名相同。

在javascript脚本文件中，[**Unity3D**](http://www.unitymanual.com)虽然没有明确地定义主类，但事实上，u3d已经隐性地自动定义了主类，并将类名设置为等于脚本文件名（不包括扩展名）。

如果使用的是C#a或Boo脚本，那就必须得手动的将主类名设置为与文件同名。

4. 使用C#实现协同，在语法上会有一处不同。

[**Unity3D**](http://www.unitymanual.com/3296.html)中的协同会同时用一个属于IEnumerator接口类型枚举的返回值和你使用的yield 返回值来替代yield

代码如下：

using System.Collections;

using UnityEngine;

public class NewBehaviourScript : MonoBehaviour {

// C# coroutine

IEnumerator SomeCoroutine () {

// Wait for one frame

yield return 0;

// Wait for two seconds

yield return new WaitForSeconds (2);

}

}

5.不要使用命名空间。

Unity3D目前不支持你在脚本中使用命名空间，这个需求会在未来的版本中实现。

6. 只有成员变量是可以在U3D程序的Inspector栏中会被以序列形式显示出来。

私有类型（private）和成员类型（protected）变量只能在专家模式（Expert Mode）下可见。

7. 避免使用构造函数

不要通过构造函数来初始化变量。这些工作可以使用第2条中的Awake方法和Start方法来替代（换句话来说就是在u3d中，Awake方法和 Start方法是每个脚本文件类中默认的构造函数）。Unity3D甚至可以在标准编辑模式下就调用它们。它们通常是直接汇编在脚本中,因为构造函数需要检索默认 脚本变量用于引用。Unity3D在任意的时候不光可以调用构造函数，还可能会调用预设物体或未被唤醒的游戏物体。

实例化C#脚本文件时，单脚本文件状态下使用自定义的构造函数可能会导致严重的后果，并且会产生引用为空的异常。

所以，如果你实例化C#脚本文件，单脚文件不要使用自定义的的构造函数，直接使用Awake方法替代即可，实在没有理由为一个继承之MonoBehaviour的文件类写任何的构造函数代码