**简化程式的架构—Singleton的使用**

Posted on 2013年05月05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 131 次

架构程式的过程中，要取得玩家血量、速度，还是场面上的敌人数量、资料，会需要将狠多物件关联起来。只是在关联参考的时候，管理上往往变的狠麻烦。所以要创造一个环境去让程式可以找到所有可能需要被参考的资料，但又為了这个环境，又需要去创造更多的物件出来。

要解决这个问题，在程式的架构技巧上，有一个狠常被使用的方法 – Singleton。

这个架构指的是这项物件在环境中是独一无二的，例如单机游戏的分数资料、杀敌数等等。 在这个情况下，我们便可以使用这个技巧来简化我们的程式的架构。

Singleton 的基本范例：

|  |  |  |
| --- | --- | --- |
|  |  |  |

|  |  |
| --- | --- |
| 01 | */\** |
| 02 | *\*Author YEN LIN WU (Idleman) 2013.04.20* |
| 03 | *\*/* |
| 04 |  |
| 05 | **using** UnityEngine; |
| 06 | **using** System.Collections; |
| 07 |  |
| 08 | **public** **class** MySingleton : MonoBehaviour { |
| 09 | #region Variables |
| 10 | **private** **static** MySingleton s\_singleton = **null**; |
| 11 | #endregion |
| 12 | #region Properpty |
| 13 | **public** **static** MySingleton Singleton { |
| 14 | **get** { |
| 15 | *// 没有的话就试图寻找* |
| 16 | **if**( s\_singleton == **null** ) { |
| 17 | *// 先从游戏中物件搜寻* |
| 18 | s\_singleton = GameObject.FindObjectOfType(typeof(MySingleton)) **as** MySingleton2 |
| 19 | *// 还是找不到就自己做一个* |
| 20 | **if**( s\_singleton == **null** ) { |
| 21 | GameObject go = new GameObject("MySingleton"); |
| 22 | s\_singleton = go.AddComponent(); |
| 23 | } |
| 24 | } |
| 25 | **return** s\_singleton; |
| 26 | } |
| 27 | } |
| 28 | #endregion |
| 29 | #region Behaviour |
| 30 | **private** **void** Awake() { |
| 31 | *//防错用，摧毁掉多余的物件* |
| 32 | **if**( MySingleton.Singleton != **this** ) { |
| 33 | Destroy(**this**); |
| 34 | **return** ; |
| 35 | } |
| 36 | } |
| 37 | #endregion |
| 38 | } |

延伸为游戏资料的范例：

|  |  |  |
| --- | --- | --- |
|  |  |  |

|  |  |
| --- | --- |
| 01 | */\** |
| 02 | *\*Author YEN LIN WU (Idleman) 2013.04.20* |
| 03 | *\*/* |
| 04 |  |
| 05 | **using** UnityEngine; |
| 06 | **using** System.Collections; |
| 07 |  |
| 08 | **public** **class** MyGameData : MonoBehaviour { |
| 09 | #region Variables |
| 10 | **private** **static** MyGameData s\_singleton = **null**; |
| 11 |  |
| 12 | **public** **string** m\_id = "Idleman"; |
| 13 | **public** **int** m\_score = 0; |
| 14 | **public** **int** m\_kills = 100000; |
| 15 | #endregion |
| 16 |  |
| 17 | #region Properpty |
| 18 | **public** **static** MyGameData Singleton { |
| 19 | **get** { |
| 20 | **if**( s\_singleton == **null** ) { |
| 21 | s\_singleton = GameObject.FindObjectOfType(typeof(MyGameData)) **as** MyGameData; |
| 22 |  |
| 23 | **if**( s\_singleton == **null** ) { |
| 24 | GameObject go = new GameObject("MyGameData"); |
| 25 |  |
| 26 | s\_singleton = go.AddComponent(); |
| 27 | } |
| 28 | } |
| 29 |  |
| 30 | **return** s\_singleton; |
| 31 | } |
| 32 | } |
| 33 | #endregion |
| 34 |  |
| 35 | #region Behaviour |
| 36 | **private** **void** Awake() { |
| 37 | **if**( MySingleton.Singleton != **this** ) { |
| 38 | Destroy(**this**); |
| 39 | **return** ; |
| 40 | } |
| 41 | } |
| 42 | #endregion |
| 43 | } |

这时候如果要设定玩家的资料，就只要做这样的动作

分数 + 20

MyGameData.Singleton.m\_score += 202

杀敌数 + 1

MyGameData.Singleton.m\_kills++2

很简单吧~