**GUIFly脚本**

Posted on 2013年05月17日 by U3d / [Unity3D脚本/插件](http://www.unitymanual.com/category/script)/被围观 52 次

用途：一个GUI对象在屏幕中飞行，用GameObject.SendMessage告诉GUI对象何时飞行。

使用：把该脚本拖到一个GameObject上，用另一个脚本GameObject.SendMessage发送飞行消息。

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

|  |  |
| --- | --- |
| 001 | [javascript] view plaincopy |
| 002 |  |
| 003 | *//Attach this to the same as the GUIFly is attached to.* |
| 004 |  |
| 005 | *// Fly in* |
| 006 |  |
| 007 | gameObject.SendMessage(“Fly”, **true**); |
| 008 |  |
| 009 | *// Wait 5 seconds* |
| 010 |  |
| 011 | **yield** new WaitForSeconds(5); |
| 012 |  |
| 013 | *// Fly out* |
| 014 |  |
| 015 | gameObject.SendMessage(“Fly”, **false**); |
| 016 |  |
| 017 | [c-sharp] view plaincopy |
| 018 |  |
| 019 | **using** UnityEngine; |
| 020 |  |
| 021 | **using** System.Collections; |
| 022 |  |
| 023 | **public** **class** GUIFly : MonoBehaviour |
| 024 |  |
| 025 | { |
| 026 |  |
| 027 | **public** **enum** InterpolationType |
| 028 |  |
| 029 | { |
| 030 |  |
| 031 | Linear, |
| 032 |  |
| 033 | Sinusoidal, |
| 034 |  |
| 035 | Hermite |
| 036 |  |
| 037 | } |
| 038 |  |
| 039 | **public** Vector3 m\_InPosition; |
| 040 |  |
| 041 | **public** Vector3 m\_OutPosition; |
| 042 |  |
| 043 | **public** **float** m\_TravelTime = 0.5f; |
| 044 |  |
| 045 | **public** **float** m\_DelayToStartTravelingAfterMessageReceived = 0.1f; |
| 046 |  |
| 047 | **public** **bool** m\_StartWithInPosition = **false**; |
| 048 |  |
| 049 | **public** InterpolationType m\_InterpolationType = InterpolationType.Sinusoidal; |
| 050 |  |
| 051 | **void** Start () |
| 052 |  |
| 053 | { |
| 054 |  |
| 055 | transform.position = (m\_StartWithInPosition) ? m\_InPosition : m\_OutPosition; |
| 056 |  |
| 057 | } |
| 058 |  |
| 059 | IEnumerator Fly(**bool** flyIn) |
| 060 |  |
| 061 | { |
| 062 |  |
| 063 | **yield** **return** new WaitForSeconds(m\_DelayToStartTravelingAfterMessageReceived); |
| 064 |  |
| 065 | Vector3 targetPosition = (flyIn) ? m\_InPosition : m\_OutPosition; |
| 066 |  |
| 067 | **float** startTime = Time.time; |
| 068 |  |
| 069 | Vector3 startPosition = transform.position; |
| 070 |  |
| 071 | **while** (Time.time < startTime + m\_TravelTime) |
| 072 |  |
| 073 | { |
| 074 |  |
| 075 | **switch** (m\_InterpolationType) |
| 076 |  |
| 077 | { |
| 078 |  |
| 079 | **case** InterpolationType.Linear: |
| 080 |  |
| 081 | transform.position = Vector3.Lerp(startPosition, targetPosition, (Time.time - startTime) / m\_TravelTime); |
| 082 |  |
| 083 | **break**; |
| 084 |  |
| 085 | **case** InterpolationType.Sinusoidal: |
| 086 |  |
| 087 | transform.position = Sinerp(startPosition, targetPosition, (Time.time - startTime) / m\_TravelTime); |
| 088 |  |
| 089 | **break**; |
| 090 |  |
| 091 | **case** InterpolationType.Hermite: |
| 092 |  |
| 093 | transform.position = Hermite(startPosition, targetPosition, (Time.time - startTime) / m\_TravelTime); |
| 094 |  |
| 095 | **break**; |
| 096 |  |
| 097 | } |
| 098 |  |
| 099 | **yield** **return** 0; |
| 100 |  |
| 101 | } |
| 102 |  |
| 103 | transform.position = targetPosition; |
| 104 |  |
| 105 | } |
| 106 |  |
| 107 | **void** Reset() |
| 108 |  |
| 109 | { |
| 110 |  |
| 111 | m\_InPosition = transform.position; |
| 112 |  |
| 113 | } |
| 114 |  |
| 115 | **private** **static** Vector3 Sinerp(Vector3 start, Vector3 end, **float** **value**) |
| 116 |  |
| 117 | { |
| 118 |  |
| 119 | **return** new Vector3(Sinerp(start.x, end.x, **value**), Sinerp(start.y, end.y, **value**), Sinerp(start.z, end.z, **value**)); |
| 120 |  |
| 121 | } |
| 122 |  |
| 123 | **private** **static** Vector3 Hermite(Vector3 start, Vector3 end, **float** **value**) |
| 124 |  |
| 125 | { |
| 126 |  |
| 127 | **return** new Vector3(Hermite(start.x, end.x, **value**), Hermite(start.y, end.y, **value**), Hermite(start.z, end.z, **value**)); |
| 128 |  |
| 129 | } |
| 130 |  |
| 131 | */\* The following functions are also in the Mathfx script on the UnifyWiki, but are included here so the script is self sufficient. \*/* |
| 132 |  |
| 133 | **private** **static** **float** Sinerp(**float** start, **float** end, **float** **value**) |
| 134 |  |
| 135 | { |
| 136 |  |
| 137 | **return** Mathf.Lerp(start, end, Mathf.Sin(**value** \* Mathf.PI \* 0.5f)); |
| 138 |  |
| 139 | } |
| 140 |  |
| 141 | **private** **static** **float** Hermite(**float** start, **float** end, **float** **value**) |
| 142 |  |
| 143 | { |
| 144 |  |
| 145 | **return** Mathf.Lerp(start, end, **value** \* **value** \* (3.0f - 2.0f \* **value**)); |
| 146 |  |
| 147 | } |
| 148 |  |
| 149 | } |