**Unity3D教程：判断物体是否选中**

Posted on 2013年07月03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 64 次

判断物体是否选中，首先需要一个[**屏幕动态框**](http://www.unitymanual.com/7060.html)，有了方框，就只需判断物体是否在方框内，按照经典的数学算法可以根据直线方程和点的坐标计算判断点在线的左边右边 还是线上。

空间平面方程可表示为：

Ax+By+Cz=0

对于点（x1, y1, z1），有

若 Ax1+By1+Cz1 = 0,则点在平面上；

若 Ax1+By1+Cz1 < 0,则点在平面的一侧；

若 Ax1+By1+Cz1 > 0,则点在平面的另一侧；

但是在这里，因为都是水平、垂直的线，我们并不需要考虑画斜线、曲线框选物体。所以只需要简单的比大小就行了，土了那么一点但其实效率更高，毕竟作游戏不是算法比赛，怎么简单高效怎么用。

如果：物体的屏幕position.x >方框左下角.x ;; 物体的屏幕position.y >方框左下角.y ;; 物体的屏幕position.x <方框右上角.x ;; 物体的屏幕position.y <方框右上角.y, 那么这个物体就是在框选范围里了。至于z的判定，直接选择摄像机的near和far距离就行了，小于near大于far都无视。因为屏幕空间是左下角为0,0，判断是否在方框内要注意这个前提。

有Unity内置的世界坐标->屏幕坐标转换，其实根本不需要什么复杂的裁剪算法，也不需要搞一大堆参考物体，往外发射一堆射线什么的，一次坐标转换加6个判断条件的一条if语句，就完成全部选择逻辑了，简单高效。

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

|  |  |
| --- | --- |
| 01 | **void** checkSelection(Vector3 start,Vector3 end){ |
| 02 | Vector3 p1 = Vector3.zero; |
| 03 | Vector3 p2 = Vector3.zero; |
| 04 | **if**(start.x>end.x){*//这些判断是用来确保p1的xy坐标小于p2的xy坐标，因为画的框不见得就是左下到右上这个方向的* |
| 05 | p1.x = end.x; |
| 06 | p2.x = start.x; |
| 07 | } |
| 08 | **else**{ |
| 09 | p1.x = start.x; |
| 10 | p2.x = end.x; |
| 11 | } |
| 12 | **if**(start.y>end.y){ |
| 13 | p1.y = end.y; |
| 14 | p2.y = start.y; |
| 15 | } |
| 16 | **else**{ |
| 17 | p1.y = start.y; |
| 18 | p2.y = end.y; |
| 19 | } |
| 20 | **foreach**(GameObject obj **in** characters){*//把可选择的对象保存在characters数组里* |
| 21 | Vector3 location = camera.WorldToScreenPoint(obj.transform.position);*//把对象的position转换成屏幕坐标* |
| 22 | **if**(location.x<p1.x || location.x>p2.x ||location.y<p1.y ||location.y>p2.y |
| 23 | ||location.z<camera.nearClipPlane || location.z > camera.farClipPlane)*//z方向就用摄像机的设定值，看不见的也不需要选择了* |
| 24 | { *//Unity3D教程手册：www.unitymanual.com* |
| 25 | disselecting(obj);*//上面的条件是筛选 不在选择范围内的对象，然后进行取消选择操作，比如把物体放到default层，就不显示轮廓线了* |
| 26 | } |
| 27 | **else** |
| 28 | { |
| 29 | selecting(obj);*//否则就进行选中操作，比如把物体放到画轮廓线的层去* |
| 30 | } |
| 31 | } |
| 32 | } |

注意这种判定是以物体的position点为基准，如果物体中心不在几何中心可能会有奇怪的效果，但对作游戏来说模型中心在几何中心是基本要求，所以应该不是问题。当然也可以按照部分顶点相交来选中，但是那样程序就会复杂很多，要先解构物体的mesh顶点，然后把顶点变换到屏幕坐标，然后判定是否在方框范围内。实际上对大多数游戏来说划过中心点算作选中条件完全可以满足。