**Unity3D基础教程：简单AI编写**

Posted on 2013年01月14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 363 次

开发环境：Window7、Unity3D 3.4.1、MB525defy Android 2.2.1。

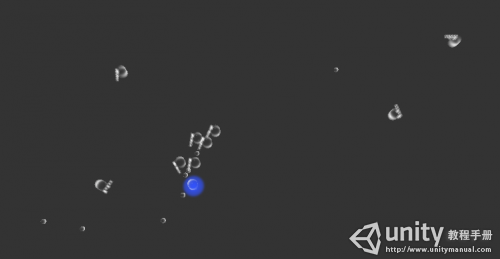
**1. Unity简单AI编写**

由于这次介绍的AI很简单，代码直接贴上，AI分成四个状态：思考，转身，移动，攻击，这里只是初步实现，所以想实现简单点的操作，就像自动范围内随机移动，锁敌攻击，超出距离复位，近距离察觉等。

Enemy\_AI.js

private var Regression : Vector3;  
public var Player\_Prefab : Transform;  
public var Enemy\_State : String;  
public var Doing : boolean = true;  
public var Range : float = 4.0;  
public var Bullet : Transform;  
public var Bullet\_Prefab : Transform;  
//初始化敌人方向和位置  
function Start()  
{  
transform.localEulerAngles.y = Random.value \* 360;  
Regression = transform.position;  
}  
//敌人行动模式  
public var Thinking : boolean = true;  
public var Thinking\_Time : float = 1.0;  
private var relativePos : Vector3;  
private var rotation : Quaternion;  
public var Facing : boolean = false;  
public var Facing\_Time : float = 2.0;  
public var Facing\_Speed : float = 2.0;  
public var Moving : boolean = false;  
public var Moving\_Speed : float = 0.5;  
public var Moving\_Time : float = 4.0;  
public var Moving\_Back : boolean = false;  
public var Attacking : boolean = false;  
private var Bullet\_DO : boolean = true;  
public var Bullet\_CD : float = 0.2;  
//随机移动方位  
private var R\_Position : Vector3;  
function Update ()  
{  
if(Attacking)  
{  
Enemy\_State = "Attacking";  
Facing = true;  
Moving = true;  
//Doing = true;  
Thinking = false;  
var dist2 = Vector3.Distance(Regression, transform.position);  
if(dist2 > 20)  
{  
relativePos = Regression - transform.position;  
rotation = Quaternion.LookRotation(relativePos);  
Attacking = false;  
Moving\_Back = true;  
}  
}  
if(!Moving\_Back)  
{  
var dist = Vector3.Distance(Player\_Prefab.position, transform.position);  
if(dist > 100)  
{  
Attacking = false;  
return;  
}  
else if(dist < 5)  
{  
Attacking = true;  
}  
RayJudge();  
}  
transform.localEulerAngles.x = 0;  
transform.localEulerAngles.z = 0;  
if(Thinking && !Attacking && !Moving\_Back)  
{  
Enemy\_State = "Thinking";  
if(Doing)  
{  
StartCoroutine(Think(Thinking\_Time));  
Doing = false;  
}  
}  
if(Facing)  
{  
Enemy\_State = "Facing";  
if(Attacking)  
{  
relativePos = Player\_Prefab.position - transform.position;  
rotation = Quaternion.LookRotation(relativePos);  
transform.rotation = Quaternion.Slerp(transform.rotation, rotation, Time.deltaTime \* Facing\_Speed \* 4);  
}  
else if(Moving\_Back)  
{  
transform.rotation = Quaternion.Slerp(transform.rotation, rotation, Time.deltaTime \* Facing\_Speed \* 4);  
}  
else  
{  
transform.rotation = Quaternion.Slerp(transform.rotation, rotation, Time.deltaTime \* Facing\_Speed);  
if(Doing)  
{  
StartCoroutine(Face(Facing\_Time));  
Doing = false;  
}  
}  
}  
if(Moving)  
{  
Enemy\_State = "Moving";  
if(Moving\_Back)  
{  
transform.Translate(Vector3.forward \* Time.deltaTime \* Moving\_Speed \* 6);  
}  
else if(dist > 2)  
{  
if(Attacking)  
{  
transform.Translate(Vector3.forward \* Time.deltaTime \* Moving\_Speed \* 4);  
}  
else  
{  
transform.Translate(Vector3.forward \* Time.deltaTime \* Moving\_Speed);  
}  
}  
if(Doing && !Attacking)  
{  
StartCoroutine(Move(Moving\_Time));  
Doing = false;  
}  
}  
}  
//前方锁敌  
function RayJudge()  
{  
var layerMask = 1 << 2;  
layerMask = ~layerMask;  
var hit : RaycastHit;  
if(Physics.Raycast (transform.position, transform.TransformDirection(Vector3.forward), hit, 20,layerMask))  
{  
var distanceToForward = hit.distance;  
if(hit.transform.tag == "Player")  
{  
Attacking = true;  
if(Bullet\_DO)  
{  
var Create = Instantiate (Bullet\_Prefab, Bullet.position, Quaternion.identity);  
Create.rigidbody.AddForce (Bullet.forward \* 1000);  
StartCoroutine(Wait(Bullet\_CD));  
Bullet\_DO = false;  
}  
}  
}  
}  
function Wait(waitTime : float)  
{  
yield WaitForSeconds (waitTime);  
Bullet\_DO = true;  
}  
function Move(waitTime : float)  
{  
print("Move");  
if(Moving\_Back)  
{  
yield WaitForSeconds (waitTime \* 0.4);  
}  
else  
{  
yield WaitForSeconds (waitTime + Random.value \* 2);  
}  
Thinking = true;  
Moving\_Back = false;  
Moving = false;  
Facing = false;  
Doing = true;  
}  
function Face(waitTime : float)  
{  
print("Face");  
yield WaitForSeconds (waitTime + Random.value);  
Facing = false;  
Thinking = false;  
Moving = true;  
Doing = true;  
}  
function Think(waitTime : float)  
{  
print("Thinking");  
yield WaitForSeconds (waitTime + Random.value);  
R\_Position = Regression + Random.insideUnitSphere \* Range;  
R\_Position.y = Regression.y;  
relativePos = R\_Position - transform.position;  
rotation = Quaternion.LookRotation(relativePos);  
Thinking = false;  
Moving = false;  
Facing = true;  
Doing = true;  
}

工程截图（这里是9个拿枪的敌人- - 蓝色为控制角色，WASD控制行动）



Unity3D基础教程：简单AI编写

**2.Unity学习过程中的一些细节分析**

1.获取位置坐标：当你translate.position获取的不是物体在世界的坐标时可以试试translate. localRotation

2.改变旋转角度：这里多半是用translate.localRotation= Quaternion.Euler(x,y,z)；

3.如何更改鼠标指针图片，这也是羽化以后可能遇到的问题，这里只能简单分析下，首先把鼠标默认指针隐藏掉Screen.showCursor=flase；再用个粒子或者图片代替，具体位置可以用Camera.main.ScreenToWorldPoint()和Input.mousePosition获得。但有个问题就产生了，UI会遮挡鼠标，鼠标图片用UI代替总感觉不妥。。。所以羽化还没想出解决方法- -

4.有关过场Loading的制作，一张图片还好说，换个Scene或者写个UI都能解决，动态Loading的是用Application.LoadLevelAsync可以达到效果，或者是预加载，具体可以看看羽化无缝地图研究博文里面的一个别墅例子。

5.也许有一天你也会遇到脚本用C#编写时遇到一些莫名其妙的错误，所以这里羽化建议动态脚本命令最好用js写。。。