# **Unity完全自制游戏纸箱战争项目记录（20180703）**

作者：[Pluto](http://gad.qq.com/user/index?id=2267380)

链接：<http://gad.qq.com/article/detail/106499>

今天因为公司来了一只拍摄团队来拍摄宣传片的缘故，项目中断了几次，还客串了一把演员出镜了一段，对项目进度产生了些许影响。

同老师讨论了一下昨天构想的两种AI设计思路；

第一种是从RTS游戏入手，用控制AI包裹住个体AI，呈向内辐射的控制模式，个体AI返回下达指令的完成情况，控制AI结合战场局势修改指令的AI设计模式。

第二种则是摒弃控制AI，把所有的个体AI孤立成为类似动画状态机的条件实现模式，在满足一系列条件之后根据与非逻辑进行判断操作。

因为之前完全没有接触到AI领域，以片面的权衡考虑来说，第二种的AI似乎更容易实现一些，但整体效果似乎与第一种相差很多。

如果按照第二种AI逻辑实现的话，项目中的NPC将不会给到玩家一种战争的感觉，更像是过家家的打闹一般。

从老师那里得到的印证也比较倾向于用第一种设计模式来实现NPC的AI效果。

在预先构想的同时觉得是很容易实现的事情，当着手操作的时候并没有想象中的这么简单；

人脑是相当复杂的结构，由数以亿万计的神经元细胞连接而成，神经网络错综复杂，而计算机中只存在01的判定，所谓AI在我理解中也应该就是若干个判断语句链接在一起构成的。

一条if语句其实就可以称作一个简易的AI了，但如果想让这个AI聪明起来就需要更多的if语句，if语句累积起来就可以凑成相当大的神经网络系统。

举个例子，就像是吃早餐：

面前摆放有一个包子，如果只有一条if语句的话，就只能判断对这个包子是吃亦或是不吃，那么条件也肯定就是自身的饥饿程度。

对吃包子再加上一条if语句，比如像是包子的馅料，如果我是个素食主义者，而包子是肉馅的，及时我已经饿了，但我也不会吃这个包子。

在此基础上再加上一条if语句，如果我现在已经快要饿死了，那素食主义者肯定就没有需要再坚持的必要了。

就像这样，三条简单的if语句就构成了一个简易的逻辑网格，那么更多的逻辑网格拼凑在一起岂不是就出现了类似人类的思想？

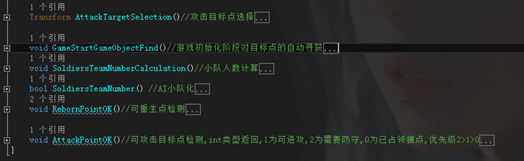
从早上起床的时间，选择穿着的服装，早餐吃的什么，乘坐怎么样的交通工具，这些够可以使用一连串的条件来判断结果。

有时候未免会出现模棱两可的情况，似乎进入某个选项都会有冲突，人类都可以通过抛硬币来结局，计算机自然也是可以通过随机数来解决的。

当然了，这只是我作为AI尚未入门者的片面认识，如果有理解错误的地方还希望能够斧正。

扯得有些远了，不过这也能够佐证一个尚可AI的书写难度。

为了保证代码的易读性，我选择把控制AI中的所有复杂算法全部封装成了方法，目前还只是封装了几个，如图所示



首先控制AI要知道自己是属于哪个阵营的，随后需要知道各个目标点现有的占领情况，这些都是不可避免的，在游戏的初始化中就调用了该方法。

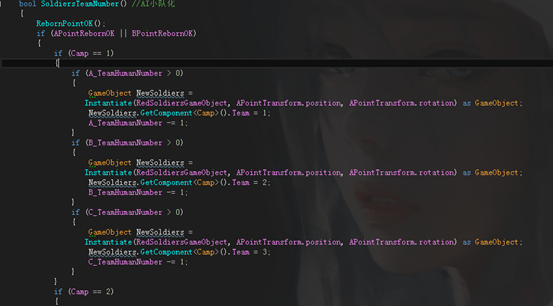
随后我想到在玩过的战争FPS游戏中，玩家普遍都是会分有小队的，指挥官会分小队下达命令，如果要一个士兵一个士兵的下达命令将会显得指令赘余，所以在游戏刚开始的阶段，就根据设定的阵营士兵数量对AI分成小队。

这里我不想再设定更复杂的多种小队了，直接让阵营中不管士兵数量都分为ABC三个小队，整除一下，剩余的士兵再统一划拨给A小队。

既然已经给士兵区分了小队，那么在士兵身上挂载的仅用于判断属性用的脚本上就要增加上一个小队选项，这样指挥官就可以通过不同的小队下达进攻亦或是防守指令。

在之前的设定中，目标点占领只有在值达到100的时候才可以使用该目标点重生。

这点也同样封装进了方法中：



在之前的代码中除了变量声明带有注释外，算法部分都是没有注释的，应该不影响理解。

该方法是返回一个初始化小队实例化完成的bool值。

首先执行的RebornPointOK（）方法是判断目标点是否允许重生。

如果A点和B点都为可重生状态则进入之后的函数，根据小队成员的现有数量判断小队是否满了，不然就给该小队实例化一名成员。

其中的Camp值为最先声明好的控制AI的阵营，1为红色，2为蓝色。

最后本日的项目进展就到此为止了，此外还封装了一个判断目标点进攻优先级的顺序，算法还不是太成熟，就先不展示出来了。