# DecisionNode使用规范

## 功能介绍：

DecisionNode在思想上基于决策树模型，在实现上又基于Task

它的目标是实现更灵活的技能释放决策，同时还要足够的简便

DecisionNode中集成了一个简易的状态机，两种决策模式（状态和概率）

使用它们，你可以创建出层次分明易懂的技能释放决策

## 常规API介绍：

常规情况下，你使用且仅需使用“saltsheep.ssl.api.DecisionNode”类

**创建新的DecisionNode：**

new DecisionNode()

**设置决策模式：**

node.markStateSwitch(String state, DecisionNode subWhenTrue, DecisionNode subWhenFalse)

根据某个状态来进行决策，当对应状态为true，交给subWhenTrue决策，当对应状态为false，交给subWhenFlase决策

node.markProbability()

给子节点添加概率，决策时根据权重来将决策权交给指定的子节点

node.markCustom(IDecision)

自定义你的决策，如果你不了解DecisionNode的运行底层，不要尝试它

**如果你使用状态模式：**

此时此刻，你将会关注到决策树拥有的状态机，你可以放心的是，整个决策树上的每一个节点的状态机都是同一个，设置子节点时，它会被共享。为了减少bug，你应该尽可能地将你的责任限制于调控状态，为此，使用下列两个方法：

node.getState(String state), node.setState(String state, boolean flag)

Extras: StateMachine node.getStates() machine.reset(), machine.setDefault(boolean)

**如果你使用概率模式：**

node.addSubOdds(DecisionNode sub, double probability)

使用它来添加新的子决策，概率 1=100%

总概率超过1后，多余部分不会被触发（子决策的分布按照添加顺序）

不要传入负数概率，我会直接给你抛出一个错误

Ps：事实上，你可以不写markProbability。添加概率子节点时会自动设置模式

**如果你使用自定义模式（可选）：**

你总有一天会觉得我提供的功能不够多

使用node.addSub(DecisionNode sub) 添加新的子决策，它们会按序添加

你在markCustom中传入的IDecision是一个函数接口

IDecision应该返回将触发子决策的对应索引

负数和超出范围的索引将会让此次决策不做任何事情

**安排决策的工作：**

当节点没有设置任何决策模式时，它将会被视作为树叶，决策树的决策会在树叶上终止，它应该作为一个工作的容器。

如何为决策节点安排工作？

node.setWork(Task)

传入saltsheep.ssl.api.Task，当决策树经过这个节点时，这个Task就会被启动

注：在1.22.0和1.22.1中，只有树叶的Task会被启动，但是在1.22.2开始，只要是决策经过的节点，其Task都会被启动

**一个实例：**

var DecisionNode = Packages.saltsheep.ssl.api.DecisionNode;

var Task = Packages.saltsheep.ssl.api.Task;

var main;

function initDecisionTree(e) {

    main = new DecisionNode();

    var yes = new DecisionNode(), no = new DecisionNode();

    main.markStateSwitch("canSeePlayer", yes, no);

    no.setWork(Task.run(function() {

        e.npc.say("我看不见人，你们在哪里？");

    }));

    var yes\_a = new DecisionNode(), yes\_b = new DecisionNode(), yes\_c = new DecisionNode();

    //yes.markProbability(); If you use addSubOdds, it is contained in addSubOdds.

    // If the total probability is less than 1.0, when the rest happen, it will do nothing.

    yes.addSubOdds(yes\_a, 0.1);

    yes.addSubOdds(yes\_b, 0.15);

    yes.addSubOdds(yes\_c, 0.3);

    yes\_a.setWork(Task.run(function() {

        e.npc.say("你在我面前晃悠什么？");

    }));

    yes\_b.setWork(Task.run(function() {

        e.npc.say("（盯着你）");

    }));

    yes\_c.setWork(Task.run(function() {

        e.npc.say("哈！我看到你了。");

    }));

}

function updateStates(e) {

    var SheepMathHelper = Packages.saltsheep.ssl.api.SheepMathHelper;

    var tar = e.npc.world.getAllPlayers()[0];

    main.setState("canSeePlayer", SheepMathHelper.isInFan(e.npc, tar, 180)>=0);

}

function init(e) {

    initDecisionTree(e);

}

function tick(e) {

    updateStates(e);

    main.decide();

}