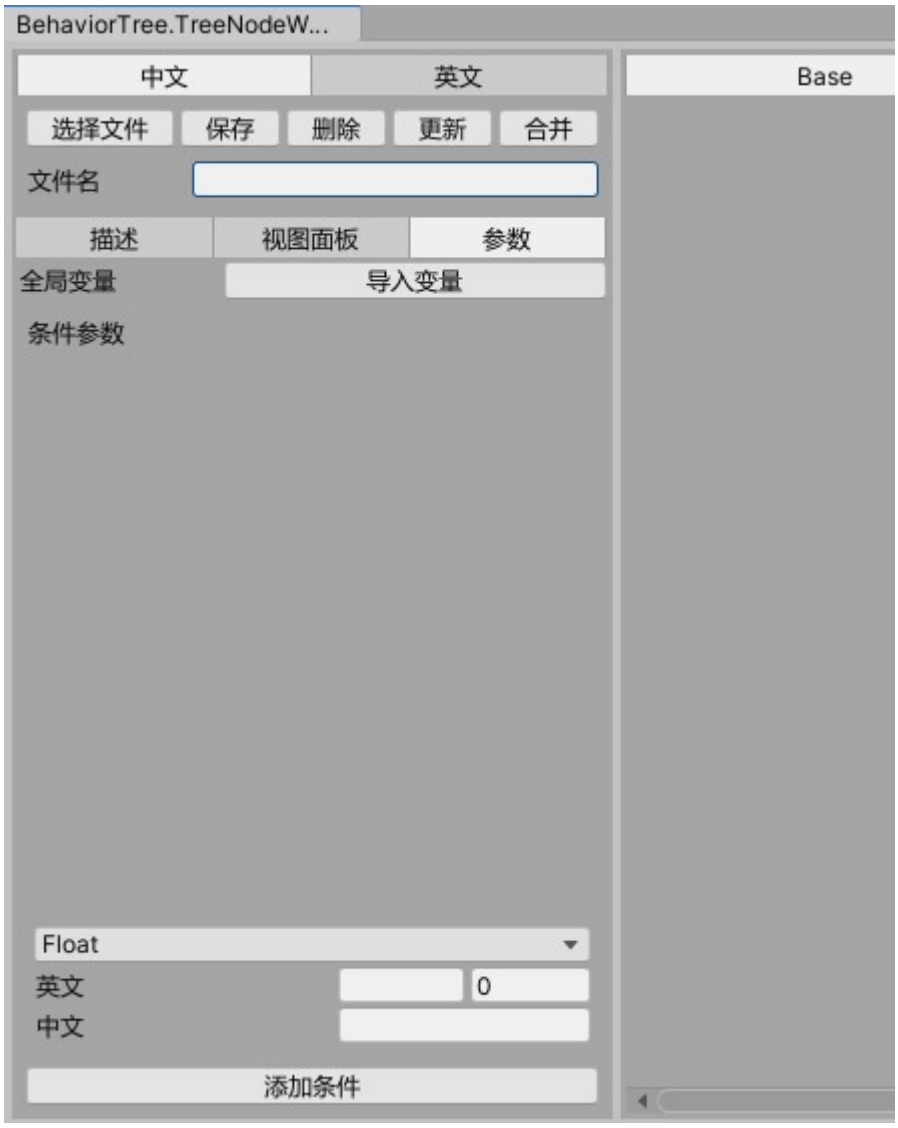
1. 请使用 Unity5.6 及以上版本(Unity5.6 以下版本没有测试)，导入 Import Package-> Custom Package... BehaviorTree.unitypackage

2. 打开编辑器窗口

Window -> BehaviorTree



选择语言按钮：中文、英文

切换为不同的语言，语言表路径

Assets\BehaviorTree\GameData\CSVAssets\table_text_localization.csv

3. 打开的编辑器窗口如上图

(3.1)选择文件：点击选择文件按钮，打开选择窗口，选择一个已保存的配置文件打开

(3.2)保存：首先在下方 文件名 输入框中输入文件名，然后点击保存，将配置文件以

Json 格式保存在目录 Assets\BehaviorTree\GameData\BehaviorTree

(3.3)删除：点击删除按钮，删除文件名输入框所填文件

(3.4)批量更新：点击批量更新按钮，将 Assets\BehaviorTree\GameData\BehaviorTree 下所有文件，经过修改后保存至 Assets\BehaviorTree\GameData\BehaviorTree\Json，具体修改逻辑需在 ConfigFileUpdate.UpdateData 函数中实现

(3.5)合并：点击合并按钮，将 Assets\BehaviorTree\GameData\BehaviorTree 下所有文件以二进制形式合并保存至 Assets\StreamingAssets\Bina\behavior_tree_config.bytes 和

Assets\BehaviorTree\Resources\behavior_tree_config.bytes 两个目录下

4. 多选框

(4.1)描述: 对该配置文件的描述：某某 NPC AI 配置文件等等

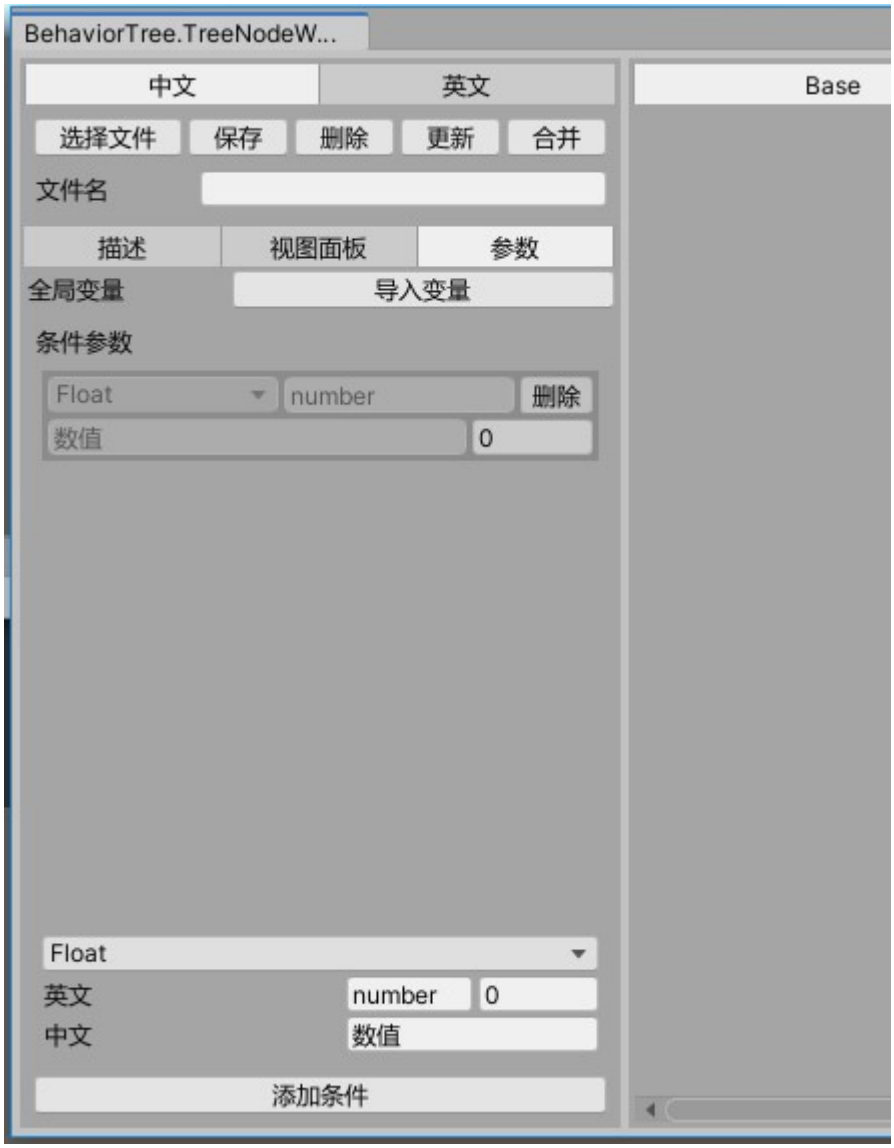
(4.2)视图面板: 行为树节点的属性参数，在后边讲述

(4.3)参数：行为树配置的所有环境变量

(4.3.1) 环境变量类型包含：float、int、long、bool、string

(4.3.2) 点击 导入变量 按钮，将配置表中变量导入至当前配置文件中，配置表目录为 Assets\BehaviorTree\GameData\CSVAssets\table_behaviortree.csv，包含变量英文名、中文名、类型、以及默认值

(4.3.3) 在窗口下方，选择变量类型，填写英文名、中文名、默认值，点击添加条件按钮，将变量添加值配置文件



5. 编辑行为树节点

(5.1)添加节点：在窗口右侧空白处鼠标右键，在弹出菜单栏中选择需要添加的节点，点击鼠标左键，即可将节点添加至配置文件

注：需要添加一个组合节点作为行为树的跟节点(也叫入口节点)

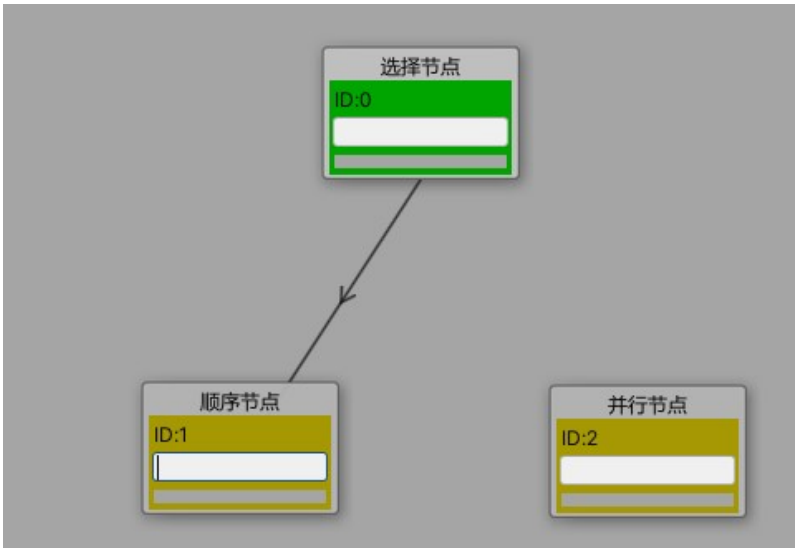
(5.2)删除节点：选择一个节点，鼠标右键，在弹出菜单栏中选择 Delete Node

(5.3)节点添加子节点：

(5.3.1)按照步骤 (5.1) 在配置文件中添加多个节点

(5.3.2)选择一个组合节点，鼠标右键，弹出菜单栏，选择 **连线**

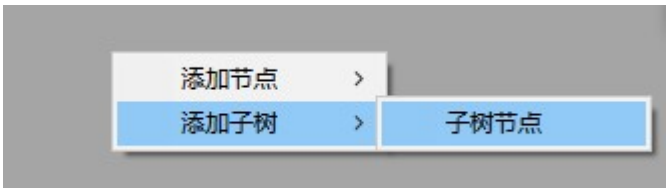
(5.3.3)拉动鼠标，从选中节点上拉出一条连线，将连线拖拽到其他节点上方，点击鼠标左键，即可添加为子节点



(5.4)删除父节点，选择一个有父节点的节点，鼠标右键，在弹出菜单来中选择移除父节点

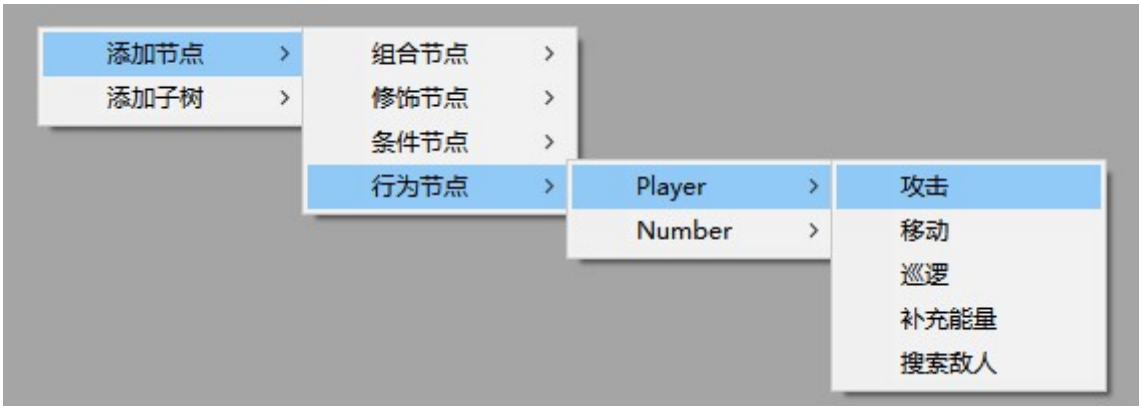
(5.5)添加子树：在空白处鼠标右键，弹出菜单栏中：添加子树-> 子树

子树节点也是组合节点

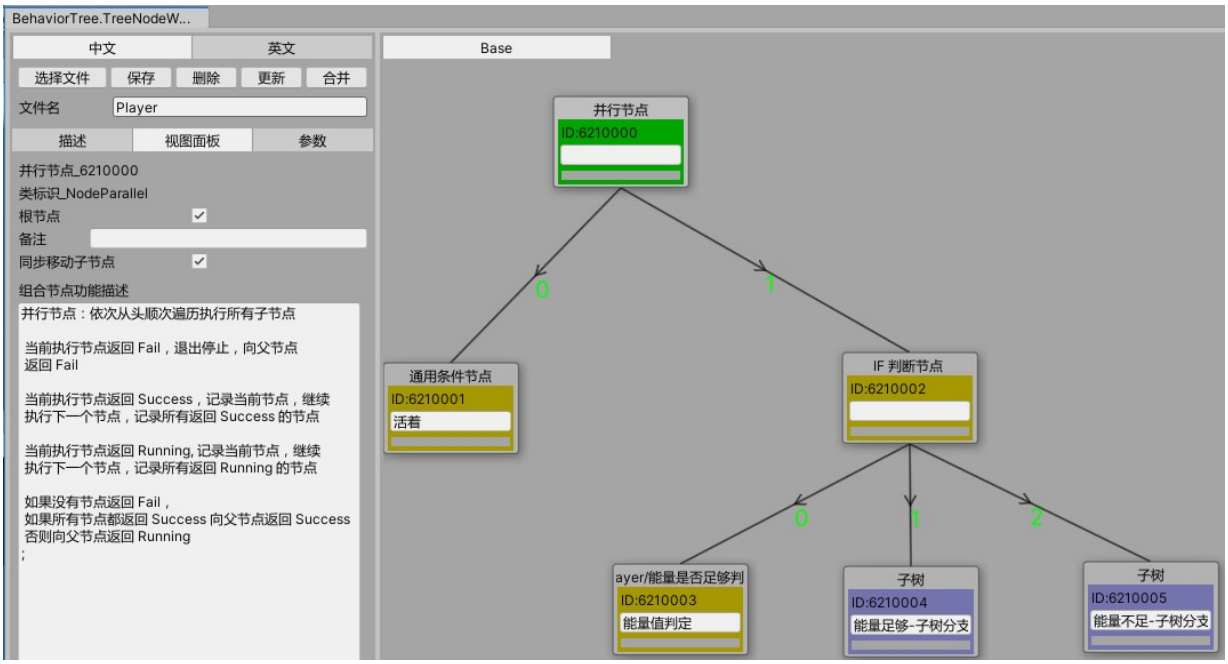


(1) 添加叶子节点：条件节点和行为节点

在空白位置鼠标右键，添加节点-> 行为节点/条件节点 点击鼠标左键添加

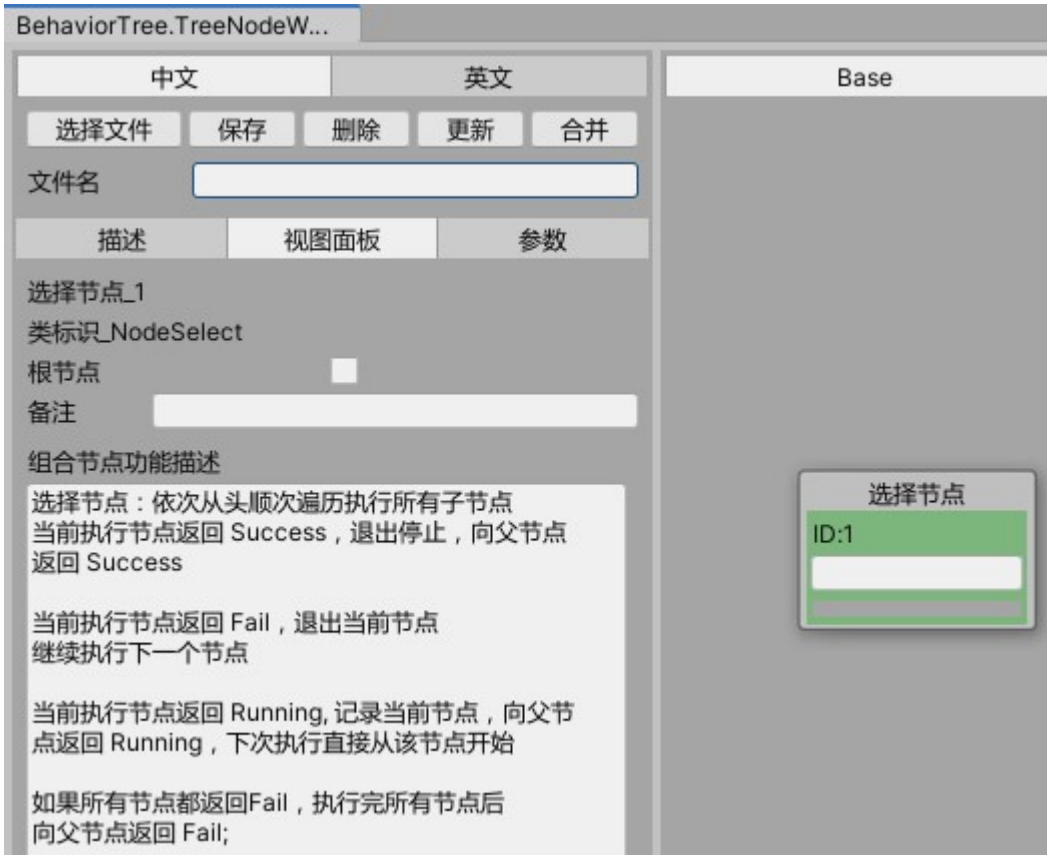


下面是一个我编辑的行为树配置文件



6. 在 (4.2) 处省略了的 **视图面板**在此处讲解

选择一个节点，然后选择 **视图面板** 选项，展示了所选节点的属性、参数，以及节点描述



如上图选择了节点，在 视图面板 面板下方显示的各项内容

(6.1) 选择一个并行节点

(6.1.1)节点类型以及节点 id

(6.1.2)类标识：编写节点的代码脚本类名

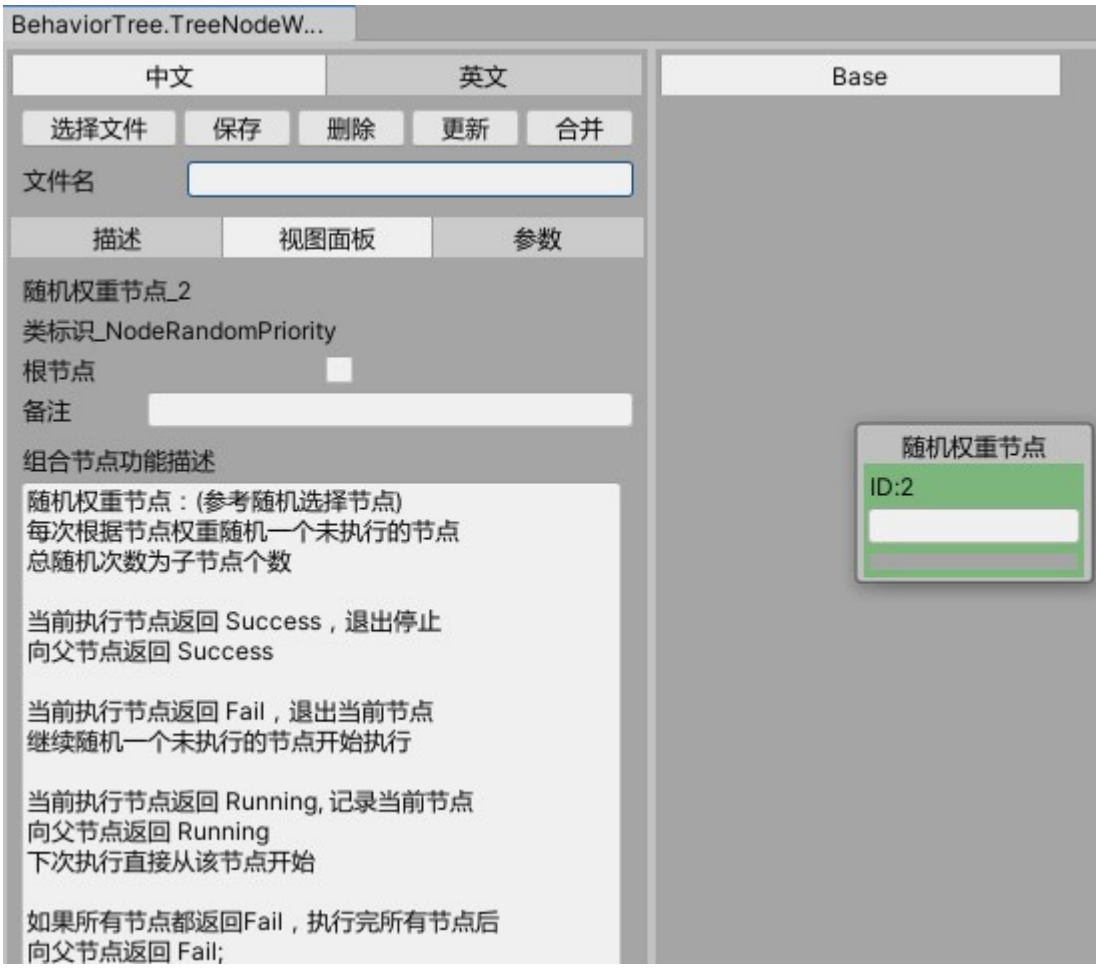
(6.1.3)跟节点：当选择一个节点为跟节点时，需要在 视图面板 面板将跟节点勾选

(6.1.4)备注：对节点的一个简单的描述，方便在行为树中快速理解逻辑

(6.1.5)同步移动子节点：当节点有子节点，且勾选该选项，拖拽节点时子节点也会跟着一起移动

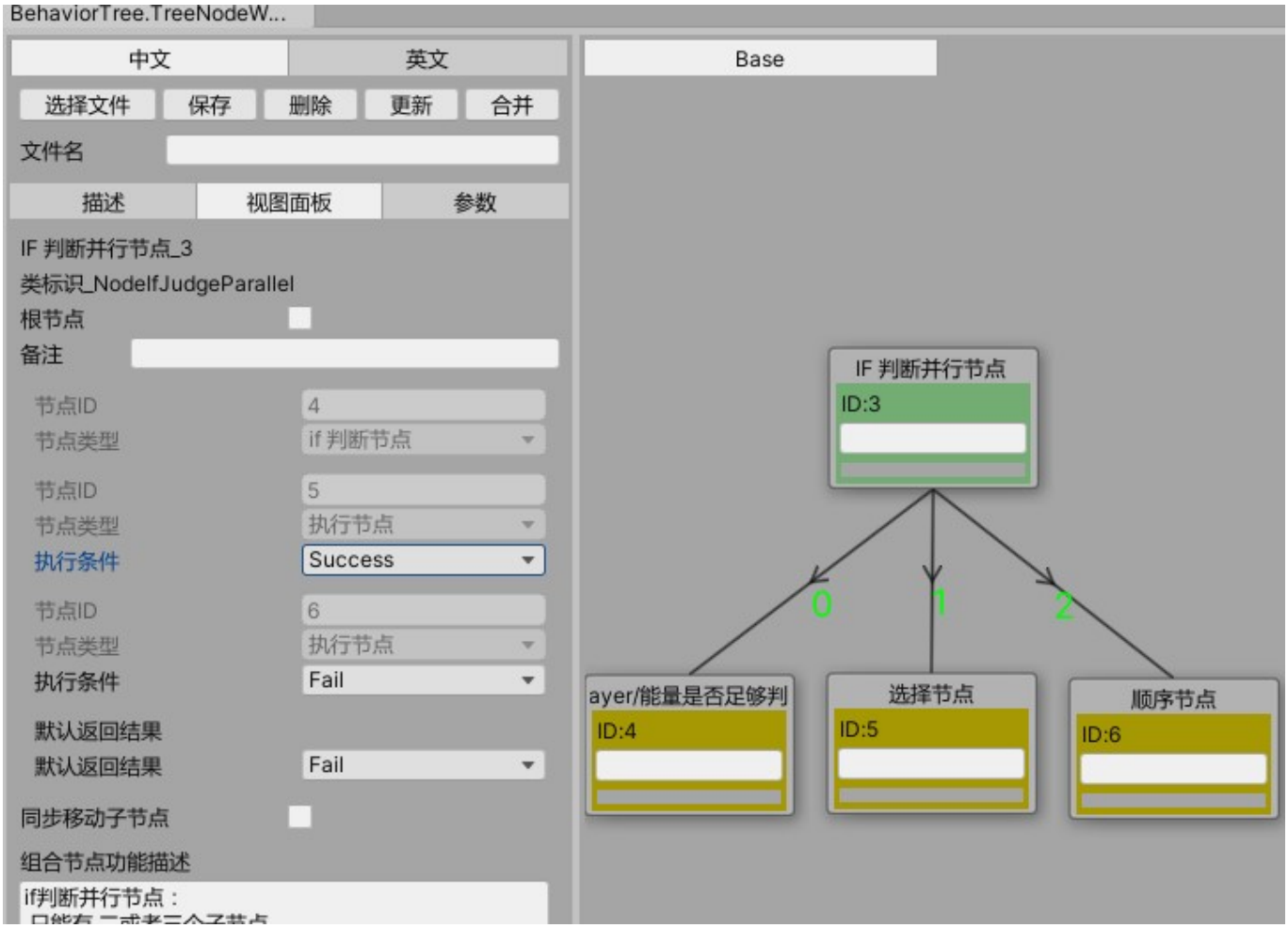
(6.1.6)组合节点功能描述：当前选择的组合节点的逻辑说明

(6.2) 随机权重节点



(6.2.1) 随机权重节点多了一项：需要填写每个子节点的随机权值

(6.3) IF 判断并行节点



(6.3.1) IF 判断节点可以配置 两个、三个子节点

(6.3.2) 可以在视图面板面板选择第二、三个节点的执行条件，这个执行条件就是第一个节点返回的执行结果，只可以是 Fail 或 Success

(6.4) 子树节点

(6.4.1) 子树类型：分为两种

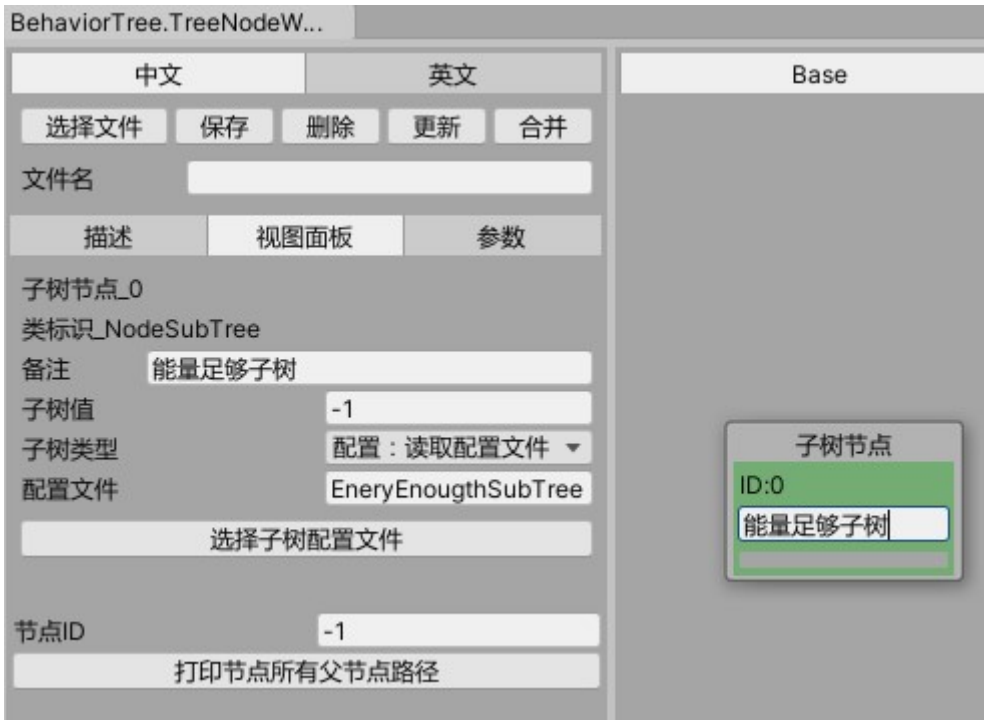
普通：可编辑子树节点

配置：读取配置文件

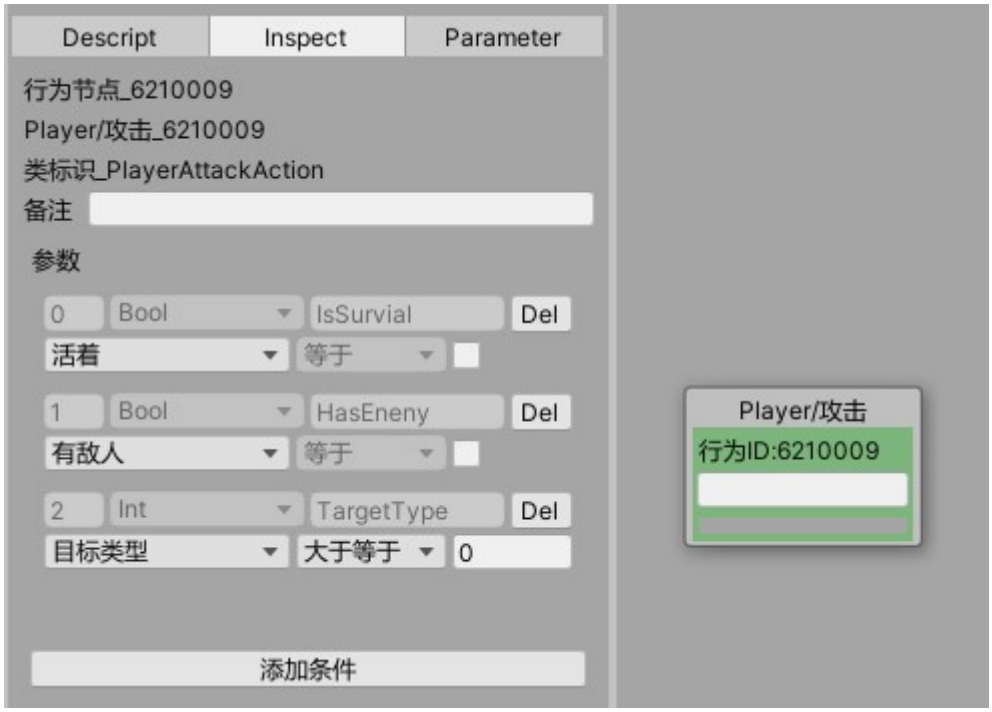
(6.4.2) 子树类型为 普通：可编辑子树节点时，双击子树节点，可以打开一个新的编辑面板，在新打开的子树编辑面板中可以添加节点、删除节点、等各种操作

(6.4.2.1) 如果后续发现配置的子树可以被其他模块共用时，可以将已经配置好子树存储为一个单独的配置文件，在配置文件处出入需要保存的文件名，点击将子存储为配置文件按钮

(6.4.3)子树类型为 配置：读取配置文件 时，点击 选择子树配置文件 按钮，在打开窗口充选择一个配置文件，作为子树配置文件，然后双击子树节点，可以打开查看所选文件的配置，但是不可以在此处修改



(6.5) 行为节点



点击 添加条件 按钮，可以添加参数，当行为节点需要填一些信息的时候，可以给他添加参数，在代码中可以获取当前行为节点配置的参数值

(6.6) 条件节点

条件节点分两种，一种是如上：通用条件节点，另一种是自定义的条件节点

(6.6.1) 通用条件节点：

(6.6.1.1)点击添加参数按钮，添加参数，然后点击 添加组按钮，如上可以看到，该节点有三个参数，两个组

(6.6.1.2)该节点的执行逻辑如下

判断第一个组的两个条件，IsSurvial = true， TargetType >= 100，如果都满足返回 Success

判断第二个组的一个条件，HasEneny = false，如果满足 返回 Success

否则 返回 Fail

(6.6.1.3) 添加组的优点：有些逻辑判断可能需要多种不同的组合，添加多个组即可满足各种复杂的配置

(6.6.2) 自定义条件节点：

当某些逻辑比较复杂，或者有些参数变量的值不方便添加为行为树的环境变量时，需要自定义 xxx 逻辑判断的条件节点，通过代码逻辑来判断

7. 编辑器中可使用的节点是如何添加到编辑器的？

一般情况下，组合节点和修饰节点是不需要修改添加的，而条件节点和行为节点会根据需求变化不断添加。

打开 BehaviorConfigNode.cs

组合节点添加在 BehaviorConfigNode.PrimaryNode() 函数中

如下

```
Config<NodeSelect>("选择节点", (int)NODE_TYPE.SELECT);
```

自定义节点

行为节点继承 ActionBase

条件节点继承 ConditionBase

在 BehaviorConfigNode.Init() 函数中添加自定义的行为、条件节点

```
Config<PlayerAttackAction>("Player/攻击");
```

为了方便为某些节点添加默认参数，可以在

BehaviorConfigNode.ConfigDefaultParameter<T>(List<string> parameterList) where T : NodeBase, new() 函数中添加

8. 扩展：动态子树

当一个角色在不同等级或者条件下，需要多种不同的 AI 配置，可以使用动态子树，然后通过代码逻辑动态的替换为不同的 AI 子树。

添加方法如下:

定义 子类 继承 NodeSubTreeDynamicBase

重写 CalculateNewSubTree() 方法，在该方法中判断当前要使用哪个子树配置文件，然后调用 SetSubTreeConfig(string config) 方法，如下

```
protected override void CalculateNewSubTree()
{
    if (level <= 50)
    {
        SetSubTreeConfig("npc_50_subTree");
    }
    else if (level <= 60)
    {
        SetSubTreeConfig("npc_60_subTree");
    }
    else if (level <= 70)
    {
        SetSubTreeConfig("npc_70_subTree");
    }
    else
    {
        SetSubTreeConfig("npc_power_subTree");
    }
}
```

9. 行为树编辑完毕，项目中如何使用？

在打开项目附带的 Human Scene，即可运行查看 AI 效果

(1)ConfigLoad 类加载配置文件

Assets\BehaviorTree\Resources\behavior_tree_config.bytes

(2)BehaviorData 类解析配置文件

(3)SpriteManager 为 BaseSprite 的管理类，

(4)BaseSprite.Init 方法中实例化 BTConcrete（行为树实例）

(5)SpriteBTUpdateManager 为 行为树的管理类

在 SpriteManager 添加 BaseSprite 的时候，将 BaseSprite 的 BTConcrete 添加到 SpriteBTUpdateManager

在 SpriteManager.Update 中驱动 SpriteBTUpdateManager.Update

在 SpriteManager 删除 BaseSprite 的时候，将 BaseSprite 的 BTConcrete 从 SpriteBTUpdateManager 移除

(6) ActionBase、ConditionBase、NodeSubTreeDynamicBase 继承了 IBTActionOwner，可以根据自己项目修改