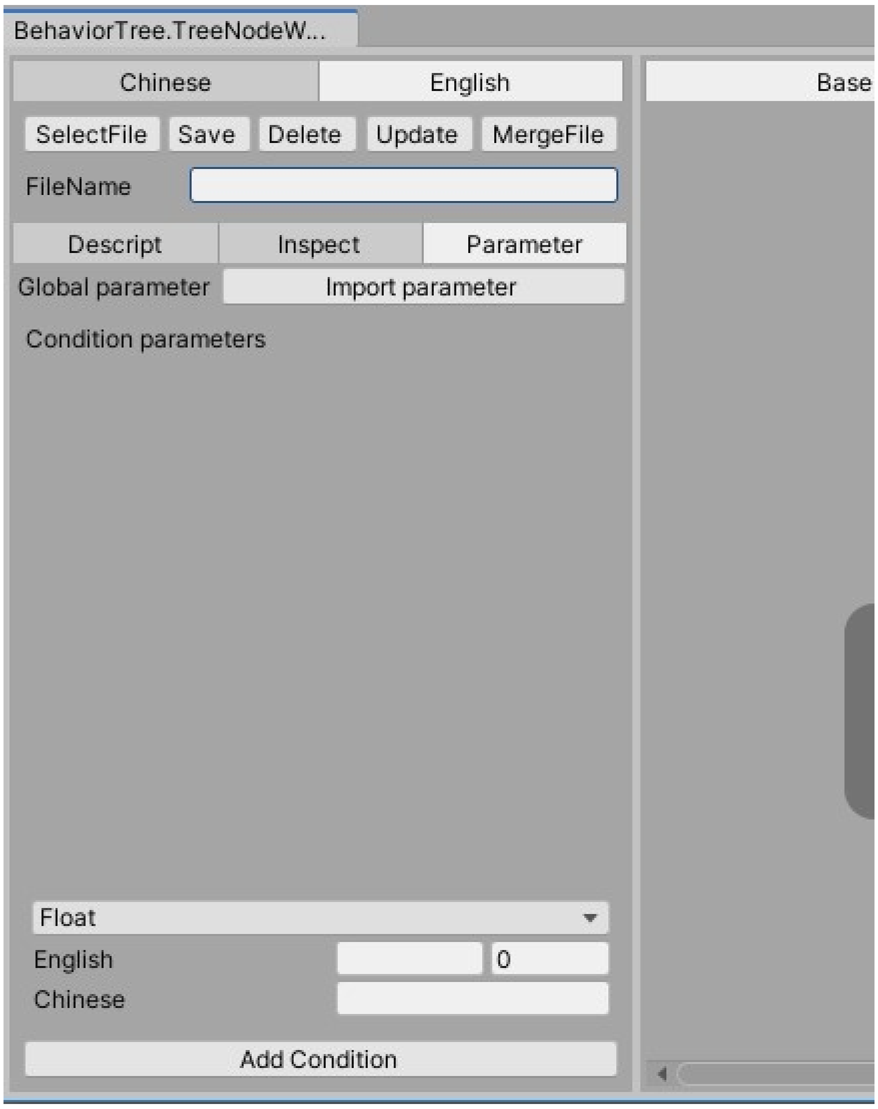
1. Please use Unity5.6 or later (Unity5.6 or later is not tested),

Import Package-> Custom Package… BehaviorTree.unitypackage

2. Open the editor window

Window -> BehaviorTree



Select language button：Chinese、English

Switch to a different language, language table path

Assets\BehaviorTree\GameData\CSVAssets\table_text_localization.csv

3. The editor window that opens, As shown above

(3.1)**SelectFile**： Click the Select File button to open the selection window and select a saved configuration file to open

(3.2)**Save**： Enter the file name in the file name input box, click **Save** Button to save the configuration file in Json format to the directory

Assets\BehaviorTree\GameData\BehaviorTree

(3.3)**Delete**： Click the Delete button to delete the file, filled in the file name input box

(3.4)**Update**： Click the **Update** button

Update all files in the folder:Assets\BehaviorTree\GameData\BehaviorTree

Savethe modification to the

directory :Assets\BehaviorTree\GameData\BehaviorTree\Json

The specific modification logic must be in Implemented in a function:

`ConfigFileUpdate.UpdateData` 函数中实现

(3.5)**MergeFile**： Click the **MergeFile** button，

Merge all files at **Assets\BehaviorTree\GameData\BehaviorTree** and save them to **Assets\StreamingAssets\Bina\behavior_tree_config.bytes** as binary files

4. Options dialog

(4.1)Descript:Description of the configuration file

For example: NPC AI configuration files and so on

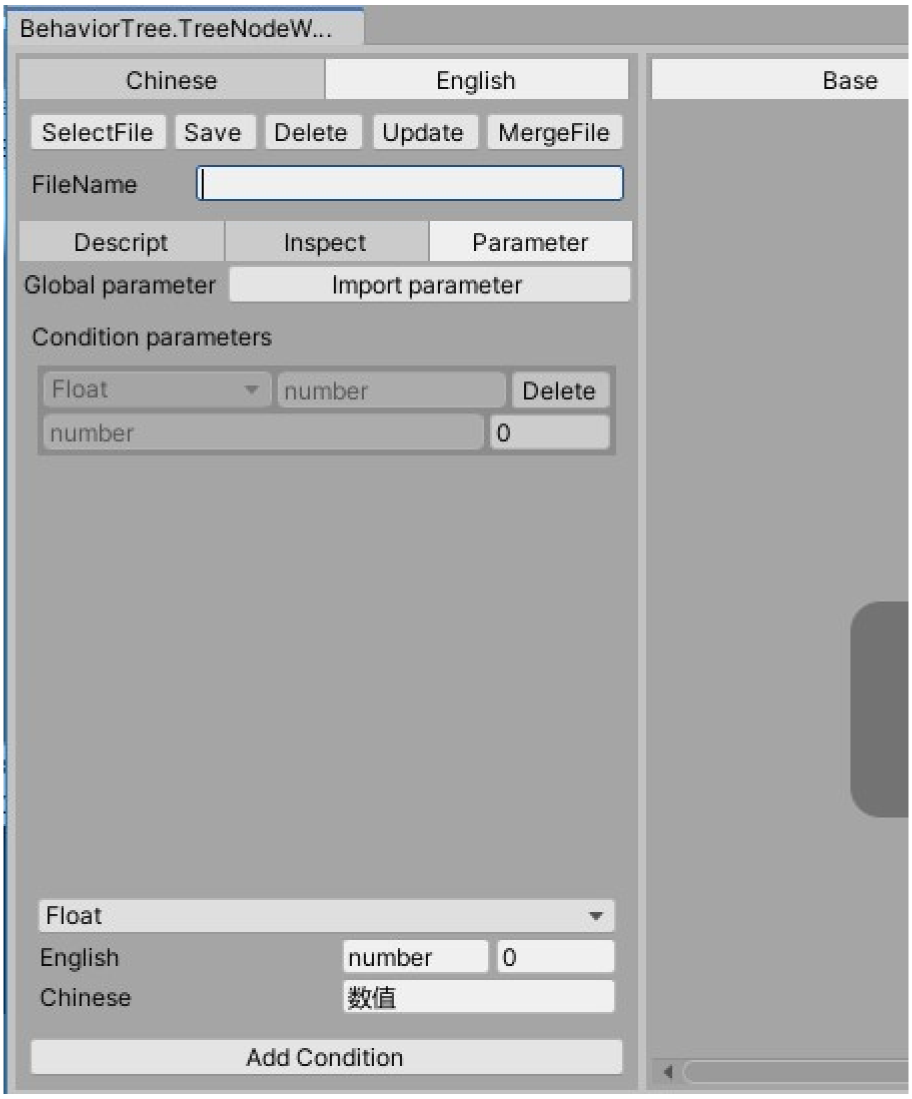
(4.2)Inspector: Attribute parameters of a behavior tree node， Tell in the background

(4.3)Parameter： All environment variables configured by the behavior tree

(4.3.1) The environment variable type contains：float、int、long、bool、string

(4.3.2) Click the **Import parameter** ButtonImport variables from the configuration table into the current configuration file,The configuration table directory: Assets\BehaviorTree\GameData\CSVAssets\table_behaviortree.csv， Contains variable English name, Chinese name, type, and default value

(4.3.3) Under the window， Select variable type, fill in English name, Chinese name, default value, click Add Condition button, add variable value configuration file



FileName

Global parameter

Condition parameters

Float number

number 0

Float

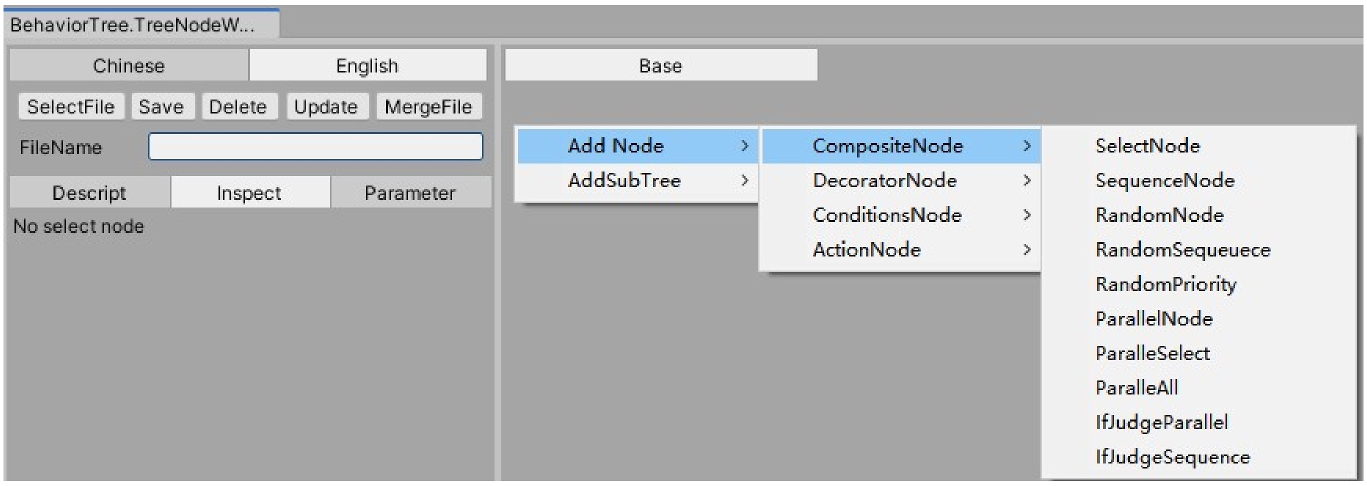
English number 0

Chinese 数值

5. Edit the behavior tree node

(5.1)Add a node： Right-click in the blank area on the right of the window, select a node to be added from the menu bar, and click to add the node to the configuration file

Note: You need to add a composite node as the root node of the behavior tree (also known as the entry node)

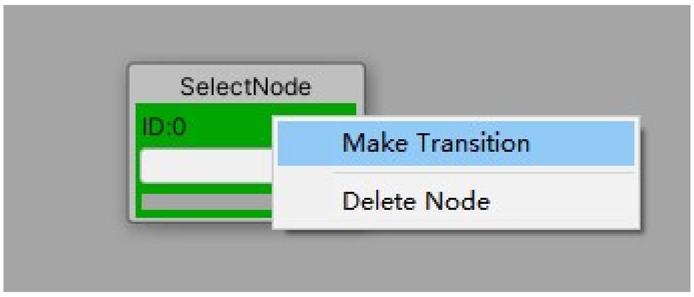


(5.2)Remove node： Right-click a Node, and choose **Delete Node** from the pop-up menu bar

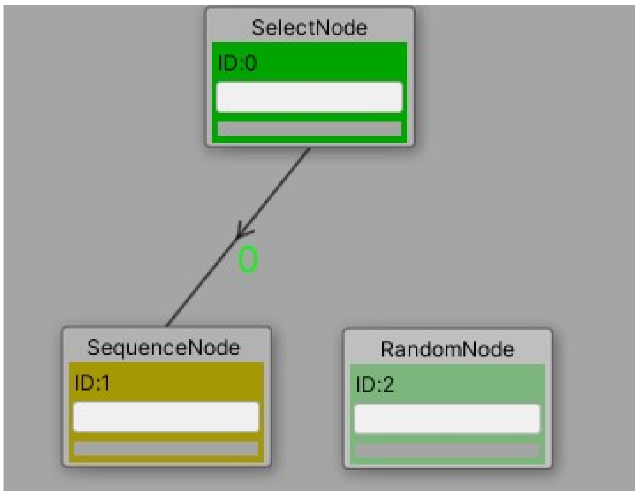
(5.3)Node Add child nodes：

(5.3.1)Follow steps (5.1) to add multiple nodes to the configuration file

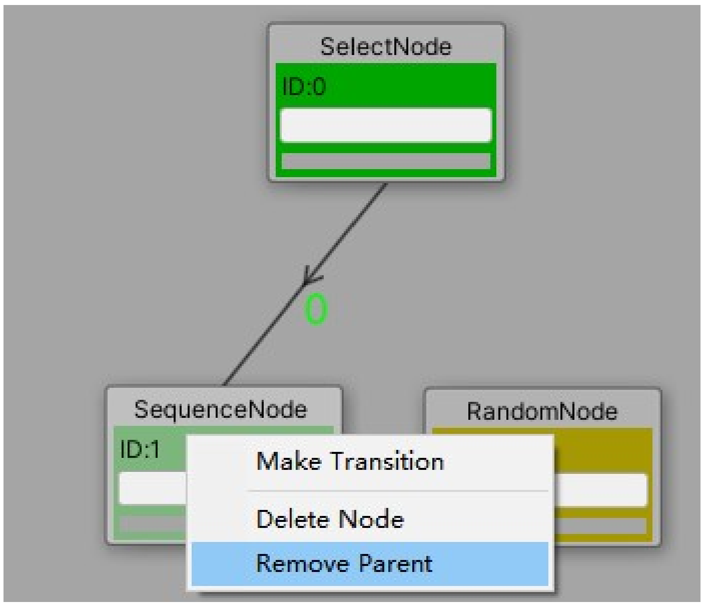
(5.3.2)Select a composite node, right mouse button, Popup menu bar, select **Make Transition**



(5.3.3)Pull the mouse to pull out a line from the selected node, drag the line above other nodes, and click the left mouse button to add it as a child node

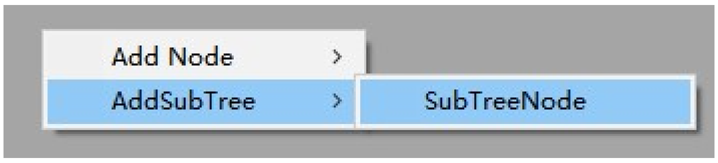


(5.4)Remove the parent node，Select a node that has a parent node, right mouse button, and choose **Remove Parent** from the pop-up menu



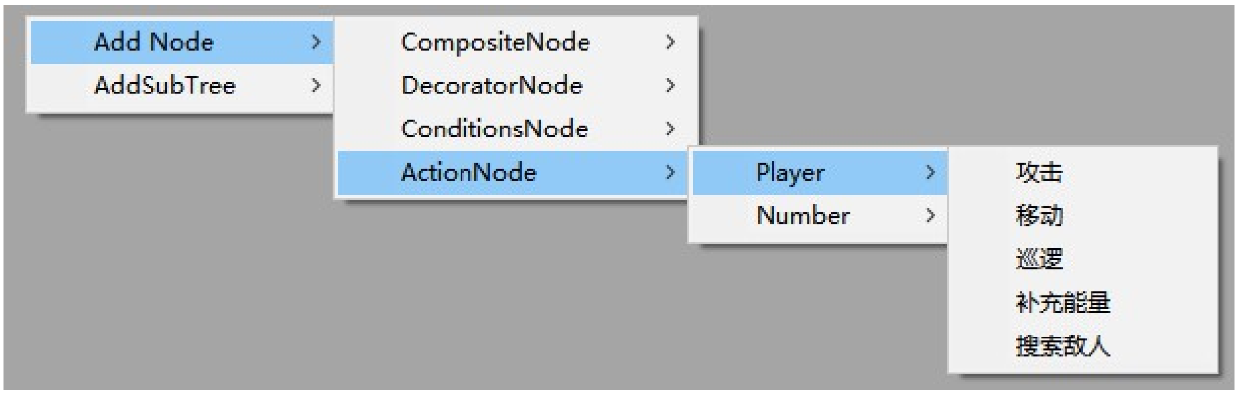
(5.5)Add the subtree：Right-click in the blank area to pop up the menu bar: Add subtree - > subtree

Subtree nodes are also composite nodes

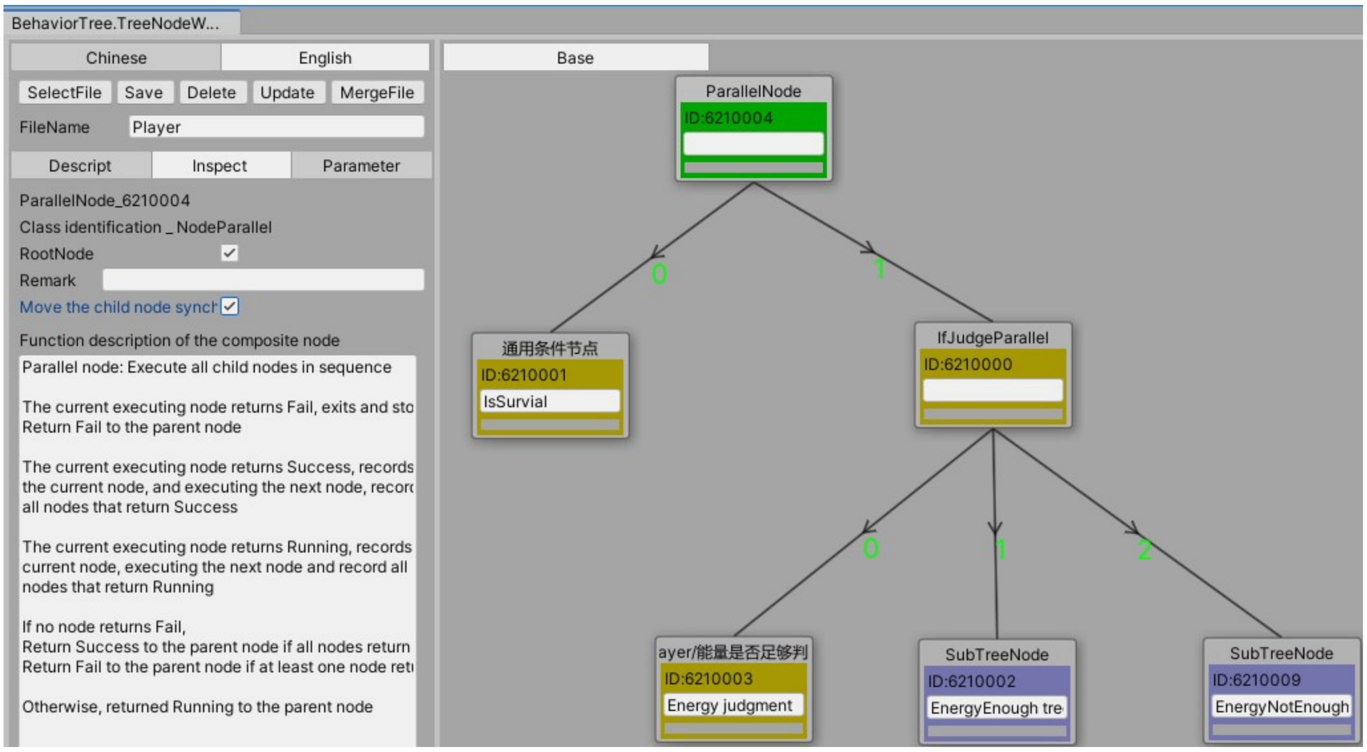


(1) Add leaf node：Condition node and Action node

Right-click in a blank position , **Add Node** -> **Action Node/ConditionsNode** is added by clicking the left mouse button



Here is a behavior tree configuration file that I edited



FileName Player

ParallelNode_6210004
Class identification _ NodeParallel
RootNode ✓
Remark
Move the child node synch ✓

Function description of the composite node

Parallel node: Execute all child nodes in sequence

The current executing node returns Fail, exits and sto
Return Fail to the parent node

The current executing node returns Success, records
the current node, and executing the next node, recorc
all nodes that return Success

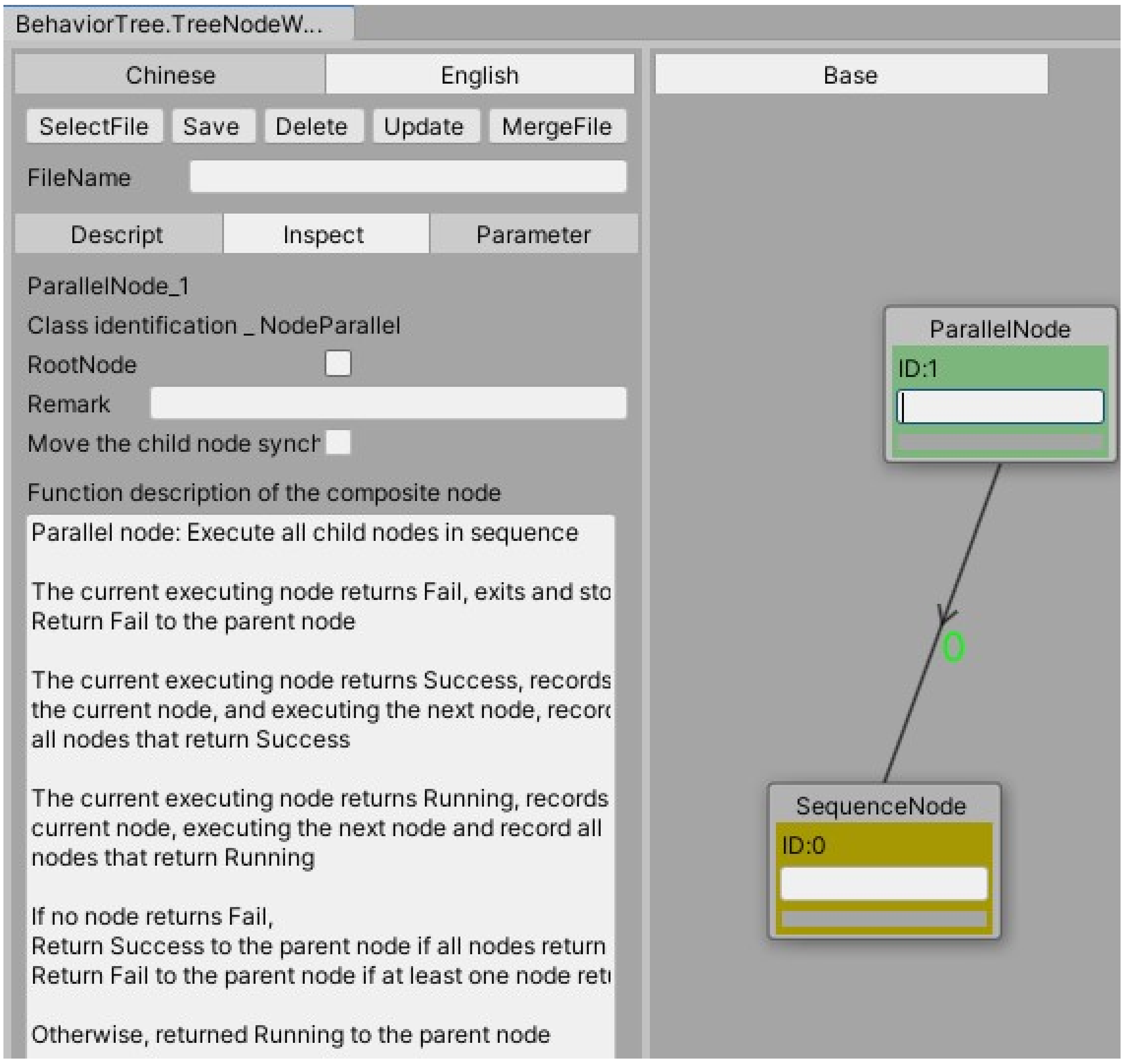
The current executing node returns Running, records
current node, executing the next node and record all
nodes that return Running

If no node returns Fail,
Return Success to the parent node if all nodes return
Return Fail to the parent node if at least one node retu

Otherwise, returned Running to the parent node

6. Interpretation of the **Inspector in (4.2)**

Select a node, and then select the **Inspectorl** option, which shows the selected node's properties, parameters, and node description



ParallelNode_1

Class identification _ NodeParallel

RootNode

Remark

Move the child node synch

Function description of the composite node

Parallel node: Execute all child nodes in sequence

The current executing node returns Fail, exits and sto
Return Fail to the parent node

The current executing node returns Success, records
the current node, and executing the next node, recor
all nodes that return Success

The current executing node returns Running, records
current node, executing the next node and record all
nodes that return Running

If no node returns Fail,
Return Success to the parent node if all nodes return
Return Fail to the parent node if at least one node ret

Otherwise, returned Running to the parent node

As shown above, the node is selected and the contents are displayed at the bottom of the view panel panel

(6.1) **ParallelNode**

(6.1.1)Node type and node ID

(6.1.2)**Class Identification**: The name of the script class that writes the node

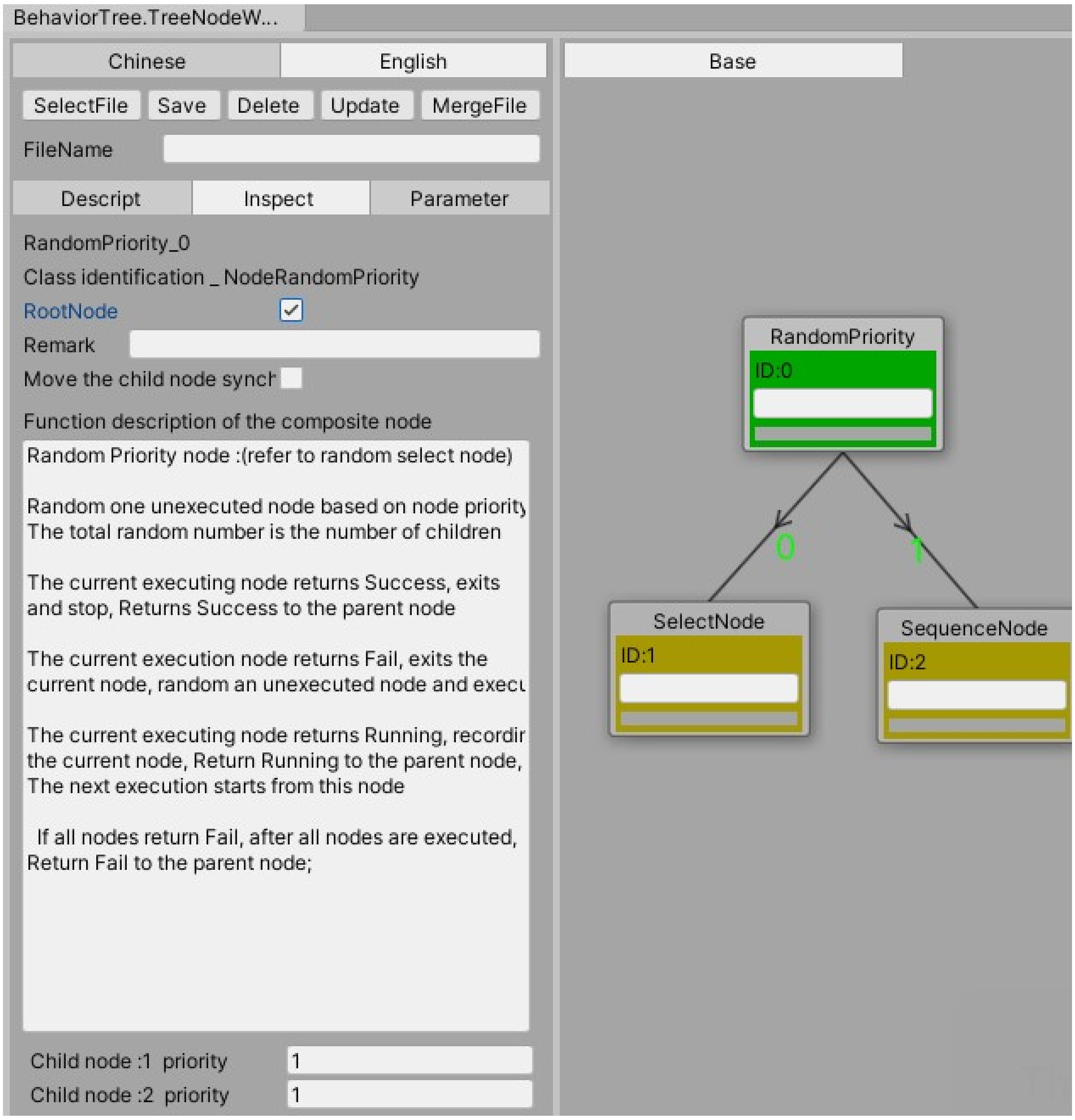
(6.1.3)**RootNode**: To select a node, select the RootNode in the **Inspector**

(6.1.4)**Remark**: A simple description of a node for quick understanding of logic in a behavior tree

(6.1.5)**Move the child node synchronous**: If a node has child nodes and this option is checked, the child nodes will also be moved when the node is dragged

(6.1.6)**Function description of the composite node**:Logical description of the currently selected composite node

(6.2) **RandomPriority** Node


BehaviorTree.TreeNodeW...

Chinese English

SelectFile Save Delete Update MergeFile

FileName

Descript Inspect Parameter

RandomPriority_0

Class identification _ NodeRandomPriority

RootNode ✓

Remark

Move the child node synch

Function description of the composite node

Random Priority node :(refer to random select node)

Random one unexecuted node based on node priority
The total random number is the number of children

The current executing node returns Success, exits and stop, Returns Success to the parent node

The current execution node returns Fail, exits the current node, random an unexecuted node and execu

The current executing node returns Running, recordir the current node, Return Running to the parent node, The next execution starts from this node

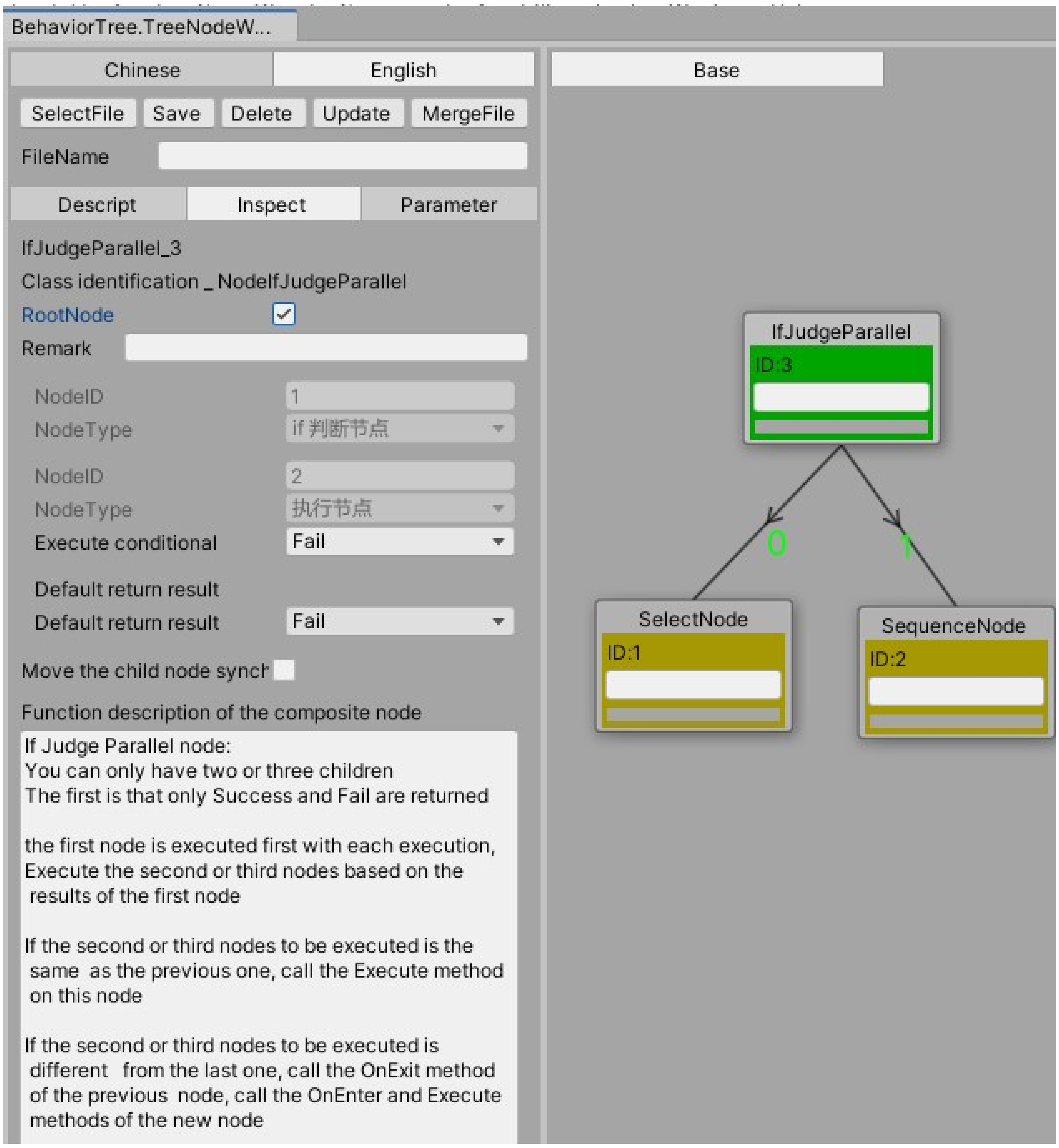
If all nodes return Fail, after all nodes are executed, Return Fail to the parent node;

Child node :1 priority	1
Child node :2 priority	1

(6.2.1) The random weight node has an extra term： Random priority for each child node need to be filled in

Child node: priority

(6.3) **IfJudgeParallel** node


BehaviorTree.TreeNodeW...

Chinese | English | Base

SelectFile | Save | Delete | Update | MergeFile

FileName

Descript | Inspect | Parameter

IfJudgeParallel_3

Class identification _ NodeIfJudgeParallel

RootNode ✓

Remark

NodeID 1

NodeType if 判断节点

NodeID 2

NodeType 执行节点

Execute conditional Fail

Default return result

Default return result Fail

Move the child node synch

Function description of the composite node

If Judge Parallel node:
You can only have two or three children
The first is that only Success and Fail are returned

the first node is executed first with each execution,
Execute the second or third nodes based on the
results of the first node

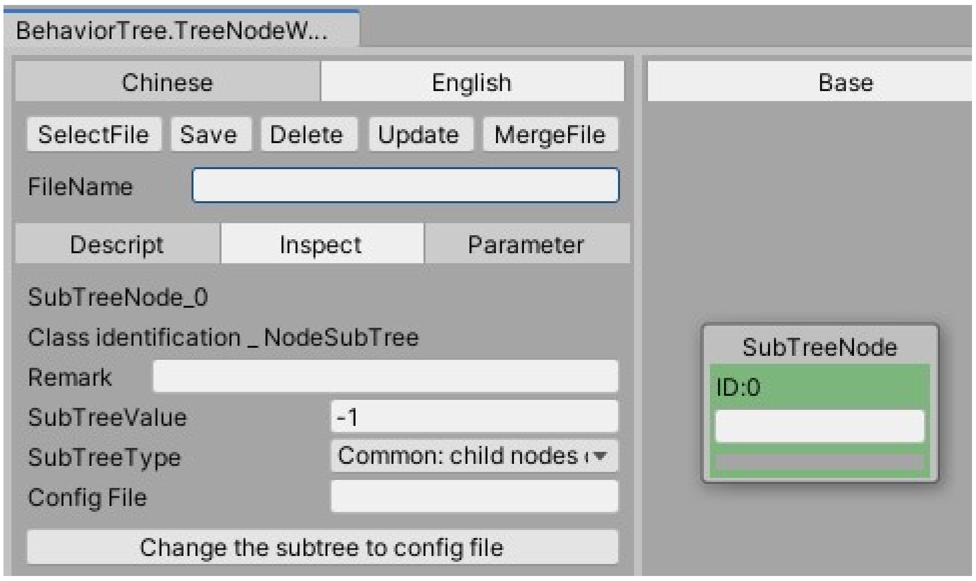
If the second or third nodes to be executed is the
same as the previous one, call the Execute method
on this node

If the second or third nodes to be executed is
different from the last one, call the OnExit method
of the previous node, call the OnEnter and Execute
methods of the new node

(6.3.1) node can be configured with two or three child nodes

(6.3.2) You can select the execution conditions of the second and third nodes in the **Inspector**. The execution conditions are the execution results returned by the first node, which can only be Fail or Success

(6.4) SubTree Node


BehaviorTree.TreeNodeW...

Chinese | English | Base

SelectFile | Save | Delete | Update | MergeFile

FileName

Descript | Inspect | Parameter

SubTreeNode_0

Class identification _ NodeSubTree

Remark

SubTreeValue: -1

SubTreeType: Common: child nodes c

Config File

Change the subtree to config file

SubTreeNode

ID:0

(6.4.1) Subtree types: there are two types

Common： child nodes can be editord

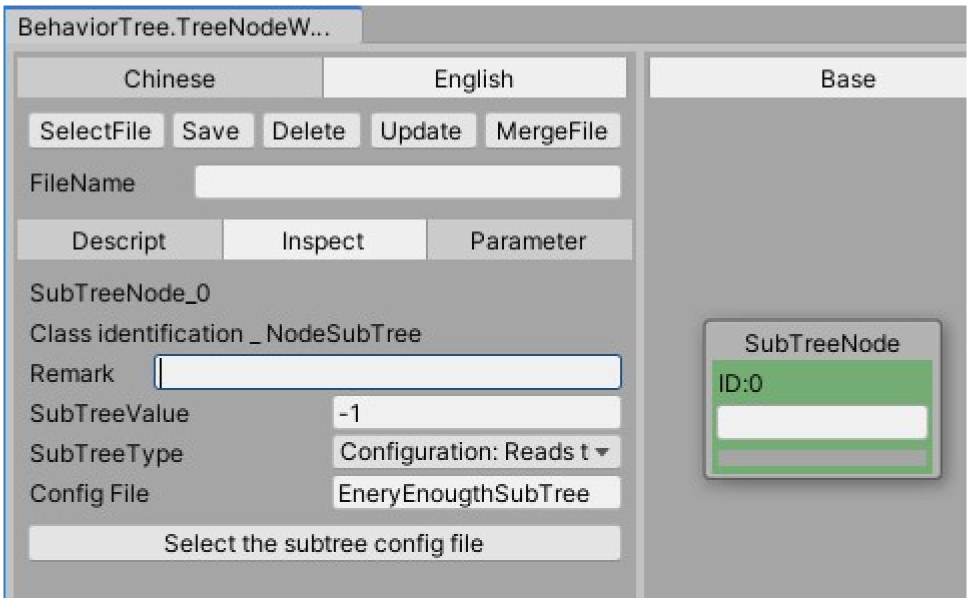
Configuration： reads the configuration file

(6.4.2) If the subtree type is **Common： child nodes can be editord**

Double-click a subtree node to open a new editing panel. You can add nodes, delete nodes, and other operations in the newly opened subtree editing panel

(6.4.2.1) If the configured subtree can be shared by other modules, you can save the configured subtree as an independent configuration file, enter the file name in the **Config File**, and click **Change the subtree to config file**

(6.4.3)The subtree type is Configuration: reads the configuration file, Click the "**Select the subtree config file**" button, select a configuration file in the open window as the subtree configuration file, and double-click the subtree node to open and view the configuration of the selected file, but can not be modified here



FileName

SubTreeNode_0

Class identification _ NodeSubTree

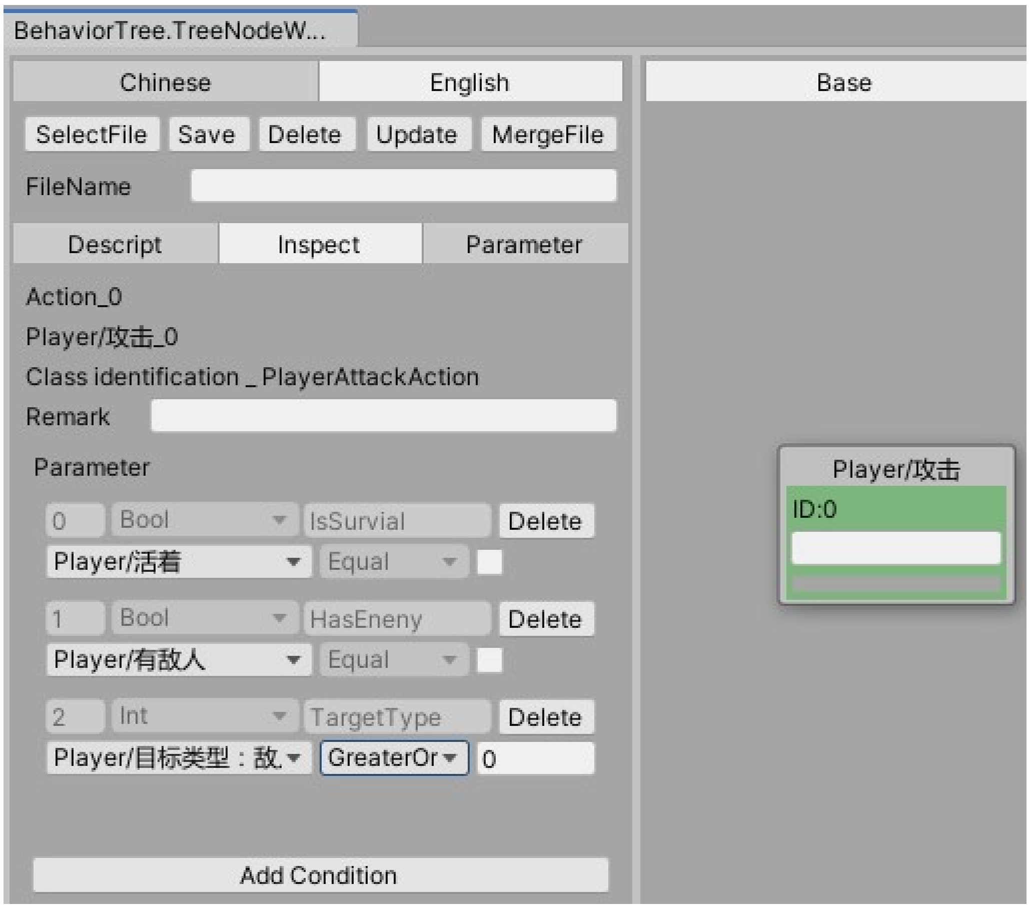
Remark

SubTreeValue -1

SubTreeType Configuration: Reads t

Config File EneryEnougthSubTree

(6.5) Action Node



FileName

Action_0

Player/攻击_0

Class identification _ PlayerAttackAction

Remark

Parameter

0 Bool IsSurvial Delete

Player/活着 Equal

1 Bool HasEneny Delete

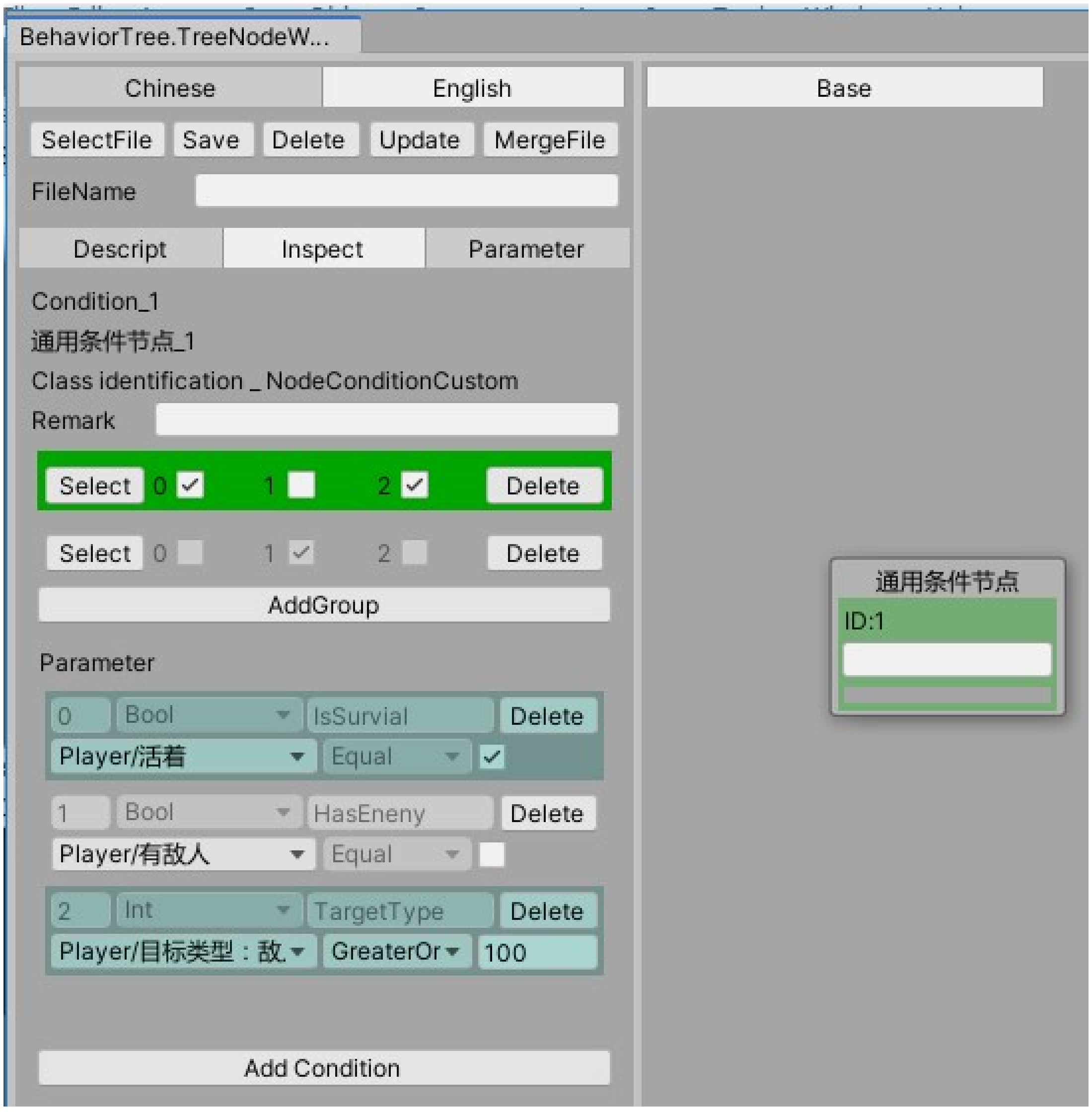
Player/有敌人 Equal

2 Int TargetType Delete

Player/目标类型：敌 GreaterOr 0

Click the “**Add Condition**” button to add parameters. When the behavior node needs to fill in some information, it can add parameters to it. The parameter value of the current behavior node configuration can be obtained in the code

(6.6) **Conditions Node**


BehaviorTree.TreeNodeW...

Chinese | English | Base

SelectFile | Save | Delete | Update | MergeFile

FileName

Descript | Inspect | Parameter

Condition_1
通用条件节点_1
Class identification _NodeConditionCustom
Remark

Select 0 ✓ 1 ☐ 2 ✓ Delete
Select 0 ☐ 1 ✓ 2 ☐ Delete
AddGroup

Parameter

0 Bool IsSurvial Delete
Player/活着 Equal ✓

1 Bool HasEneny Delete
Player/有敌人 Equal ☐

2 Int TargetType Delete
Player/目标类型：敌 GreaterOr 100

Add Condition

Condition node is divided into two kinds,

one is as above: **general condition node**

the other is a **custom condition node**

(6.6.1) **general condition node**：

(6.6.1.1)Click “**Add Condition**” button, add parameter, and then click “**Add Group**” button. As shown above, the node has three parameters and two groups

(6.6.1.2)The execution logic of the node is as follows

The first group:Two conditions，

IsSurvial = true，

TargetType >= 100，

Return SUCCESS if all are meet the conditions

The two group， One conditions，

HasEneny = false，

Return SUCCESS if meet the conditions

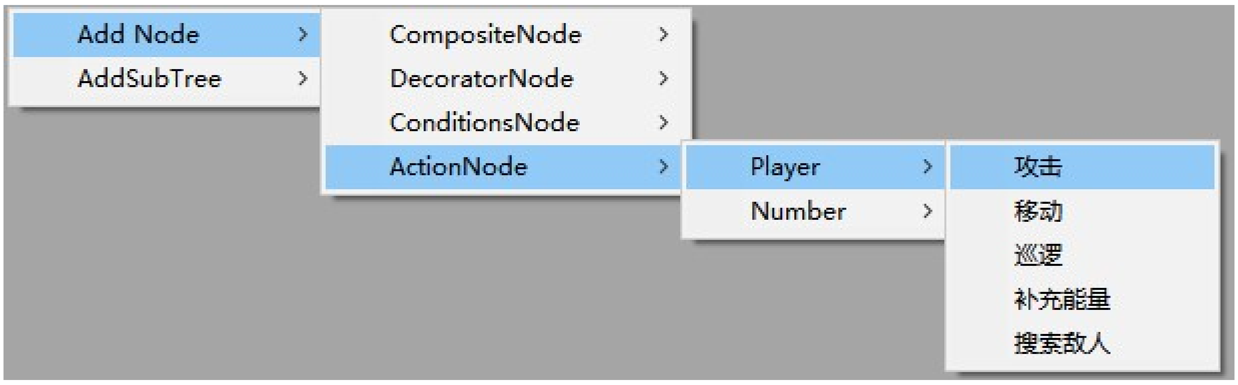
Otherwise return Fail

(6.6.1.3) Advantages of adding groups:Some logical decisions may require many different combinations, and multiple groups can be added to accommodate a variety of complex configurations

(6.6.2) Custom condition node：

When some logic is complex, or the value of some parameter variables is not convenient to be added as the environment variable of the behavior tree, it is necessary to define the condition node of XXX logic judgment by code logic

7. How are nodes used in the editor added to the editor?



In general, **Composite Node** and **Decorator Node** do not need to be added, while conditional nodes and behavior nodes are added continuously as requirements change.

Open `BehaviorConfigNode.cs`

The Composite Node is added in `BehaviorConfigNode.PrimaryNode()` method

For example

```
                    Config<NodeSelect>(EnumNames.GetEnumName<NODE_TYPE>(NODE_TYPE.SELECT),
(int)NODE_TYPE.SELECT);
```

Custom node

Action Node implementation `ActionBase`

ConditionNode implementation `ConditionBase`

in `BehaviorConfigNode.Init()` method to add **Custom Action Node**, **Condition Node**

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

To add default parameters for some nodes, you can use the

`BehaviorConfigNode.ConfigDefaultParameter<T>(List<string> parameterList) where T : NodeBase, new()` method to add

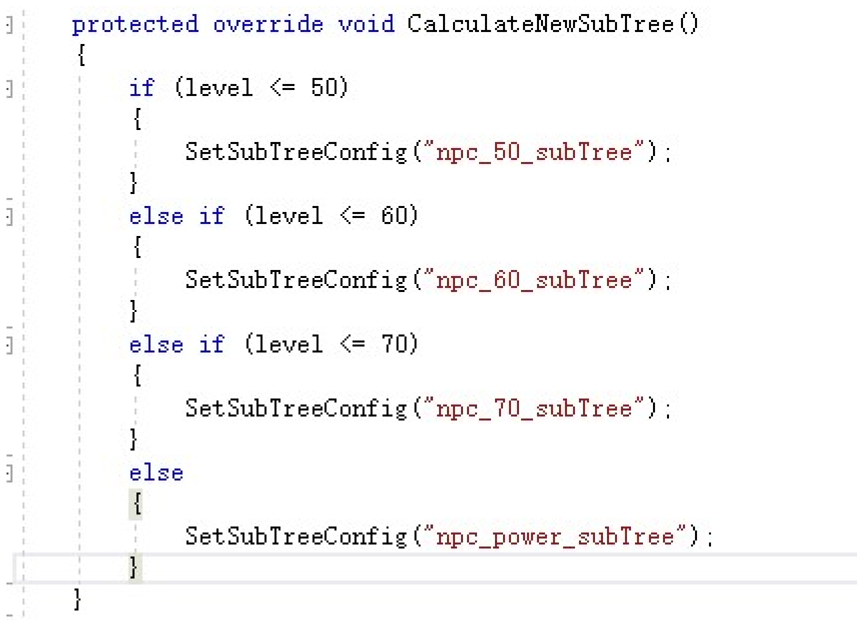
8. Extension : Dynamic subtree

When a character needs several different AI configurations at different levels or conditions, dynamic subtrees can be used, which can then be dynamically replaced by different AI subtrees through code logic.

Add the way:

Define subtreeClass implementation `NodeSubTreeDynamicBase`

`overwrite CalculateNewSubTree()` function, In this function, determine which subtree configuration file is currently being used and call `SetSubTreeConfig(string config)` function, For example,



```
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. The behavior tree is edited, How is it used in the project?

open the scene: **Human**， Can run to view the AI effect

(1)ConfigLoad.cs Class load configuration files

Assets\BehaviorTree\Resources\behavior_tree_config.bytes

(2)BehaviorData.cs analysis parsing configuration files

(3)SpriteManager.cs management BaseSprite.cs

(4)Instantiate BtConcrete.cs in the BaseSprite.Init function （Behavior tree instance）

(5)SpriteBTUpdateManager.cs Management classes for behavior trees

Add BaseSprite to SpriteManager, add BTConcrete of BaseSprite to SpriteBTUpdateManager

in SpriteManager.Update function call SpriteBTUpdateManager.Update

Delete BaseSprite in SpriteManager, Remove BaseSprite's BTConcrete from SpriteBTUpdateManager

(6) ActionBase、ConditionBase、NodeSubTreeDynamicBase implementation IBTActionOwner，You can modify it according to your own project