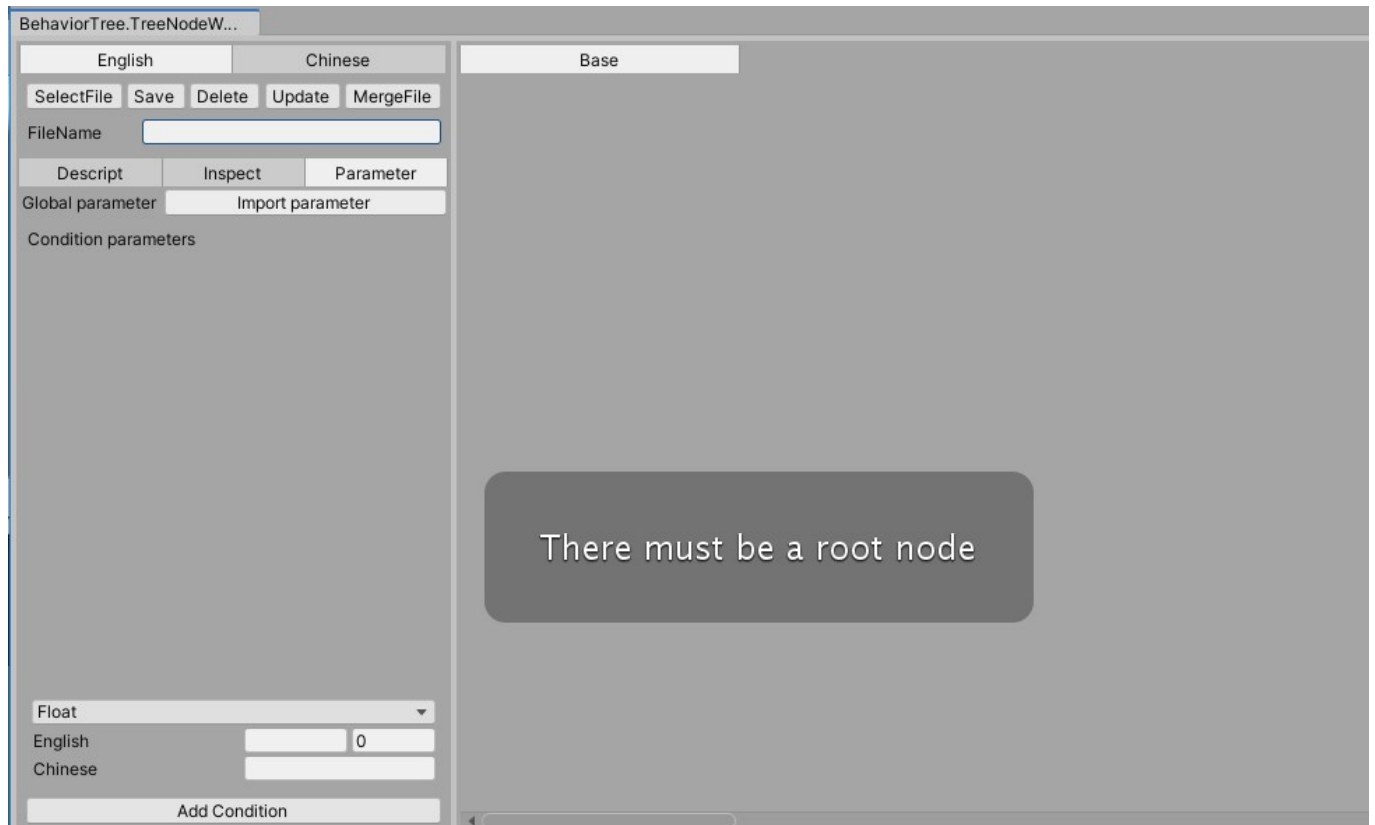


1. Please use Unity5.6 or later (Unity5.6 or later is not tested),

Import Package-> Custom Package... BehaviorTree.unitypackage

2. Open the editor window

Window -> BehaviorTree



(2.1) Select language button: **Chinese**、**English**

Switch to a different language, language table path

Assets\BehaviorTree\GameData\CSVAssets\table_text_localization.csv

(2.2) **There must be a root node** notification this means that the behavior tree must have a composite node as a root node, How to add nodes will be explained in the following section

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

Save the modification to the

directory :Assets\BehaviorTree\GameData\BehaviorTree\Json

The specific modification logic must be implemented in a function:

`ConfigFileUpdate.UpdateData`

(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

This can be a description of the function of the configuration file, as well as some remarks

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

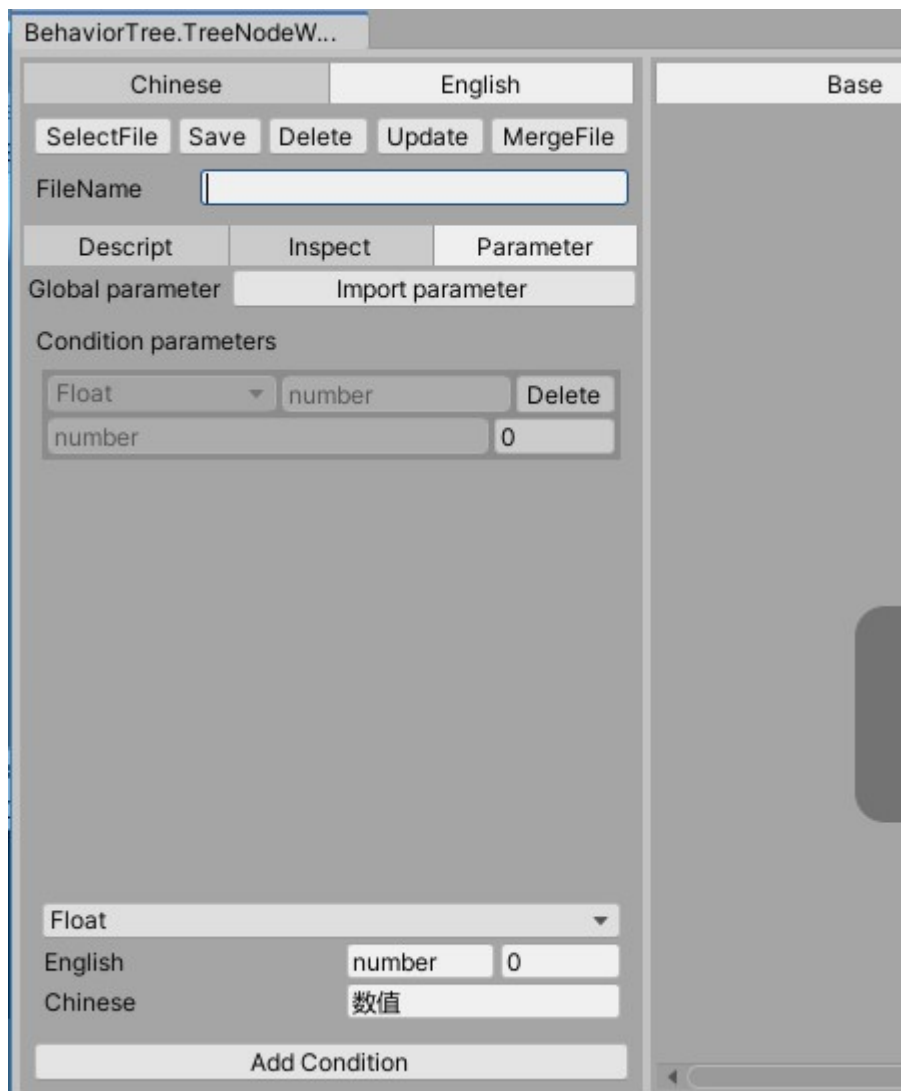
(4.3)**Parameter:** All environment variables configured by the behavior tree

(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

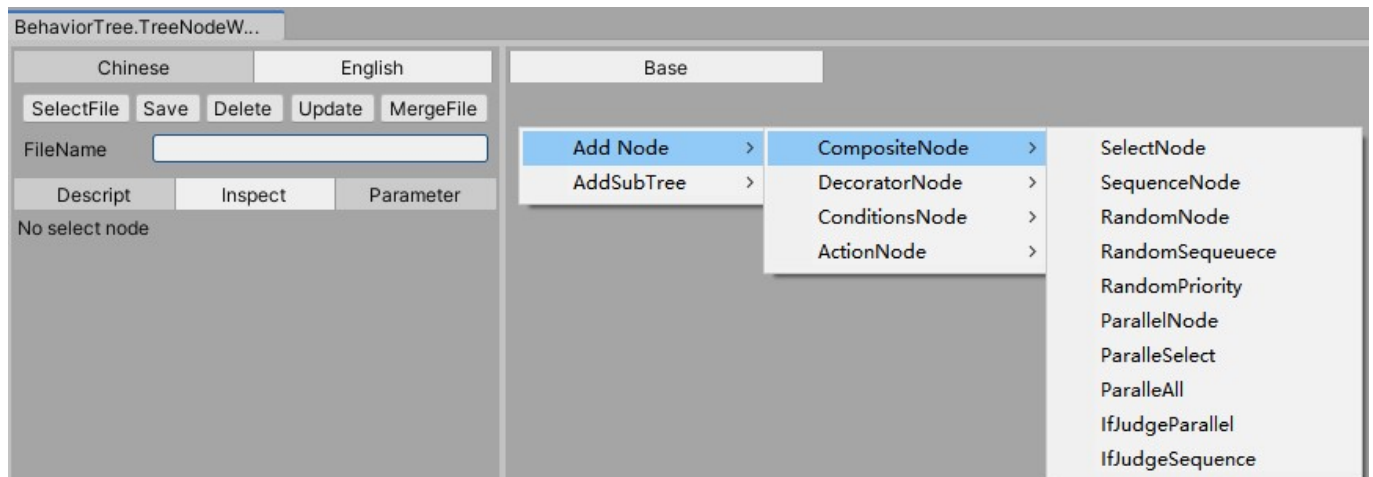


5. Edit the behavior tree node

(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)



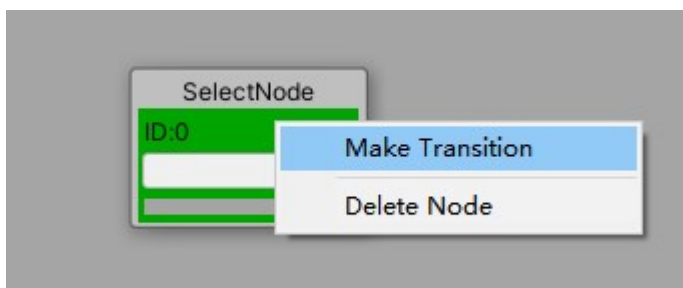
(5.2)Remove node: Right-click a Node, and choose **Delete Node** from the pop-up menu bar

(5.3)Node Add child nodes:

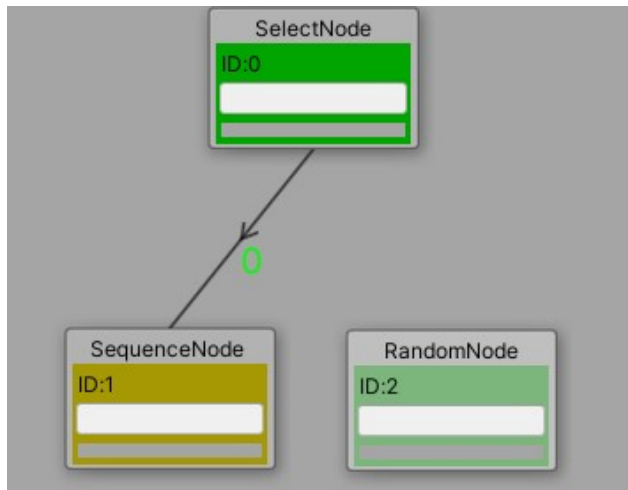
(5.3.1)Follow steps (5.1) to add multiple nodes to the configuration file

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

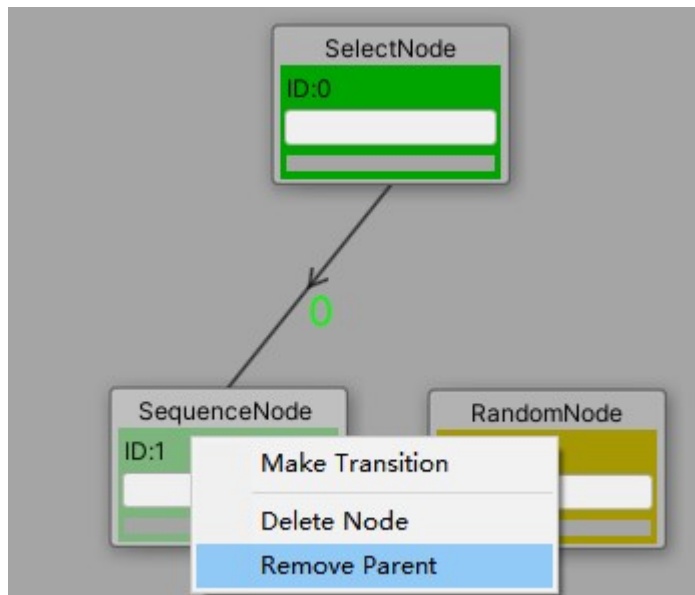
Make Transition



(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

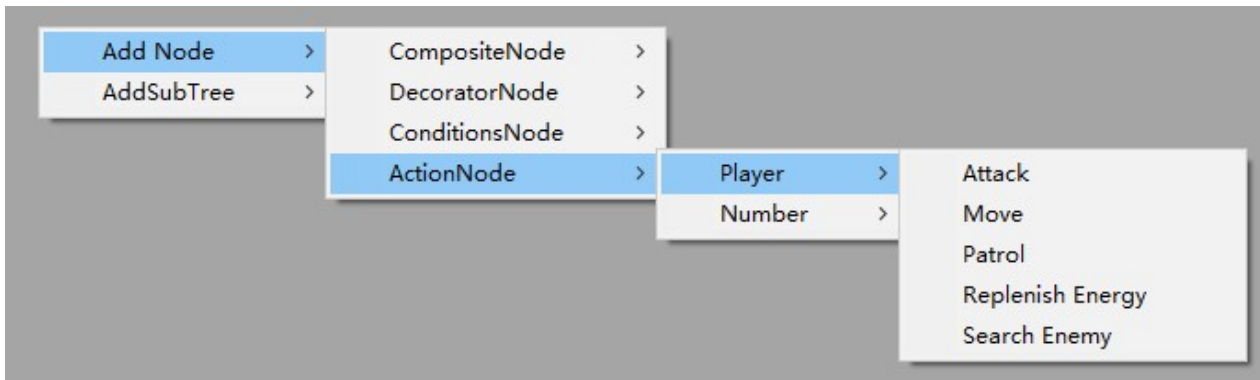
Subtree nodes are also composite nodes



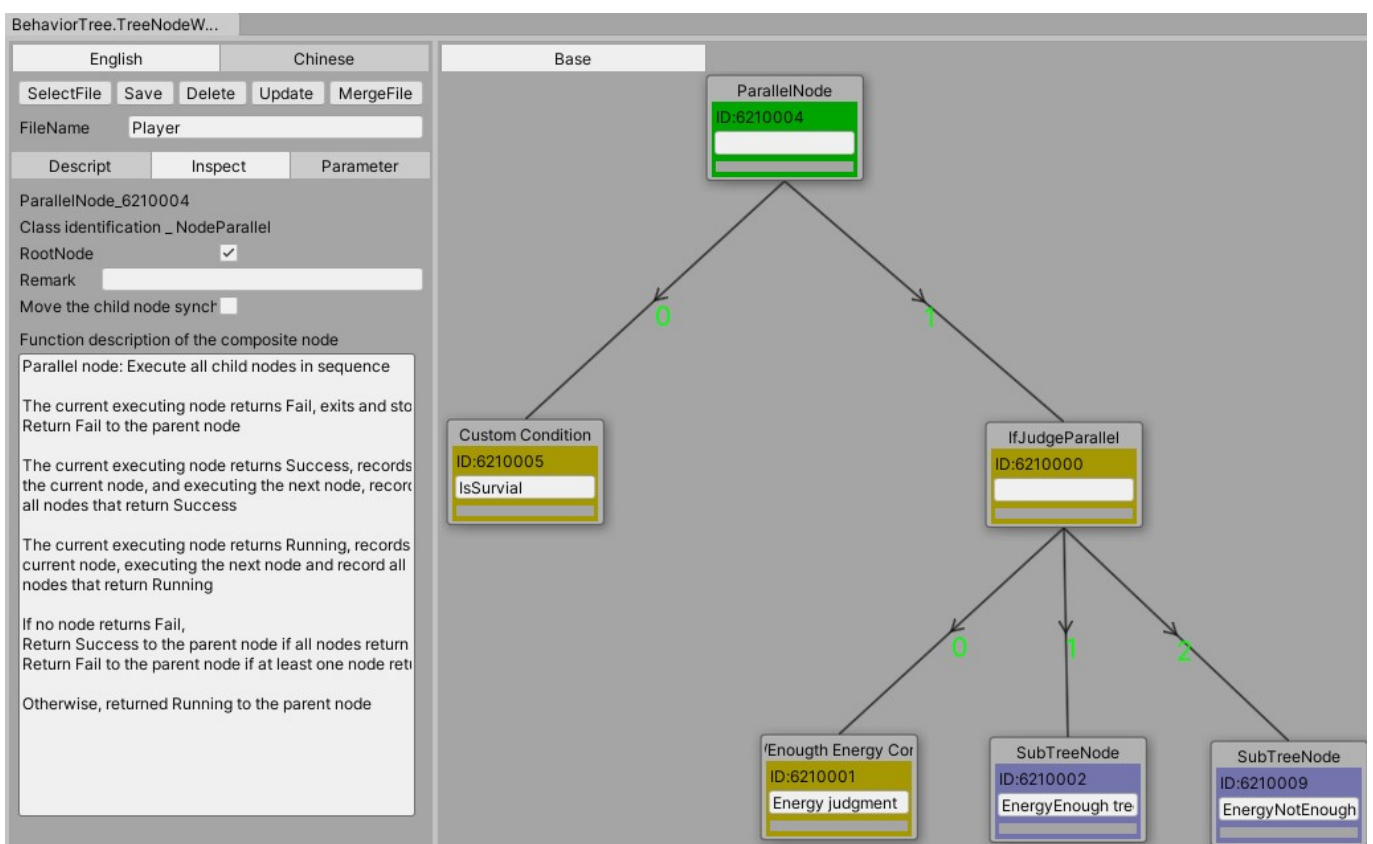
(1) Add leaf node: Condition node and Action node

Right-click in a blank position , **Add Node** -> **Action Node/ConditionsNode** is

added by clicking the left mouse button

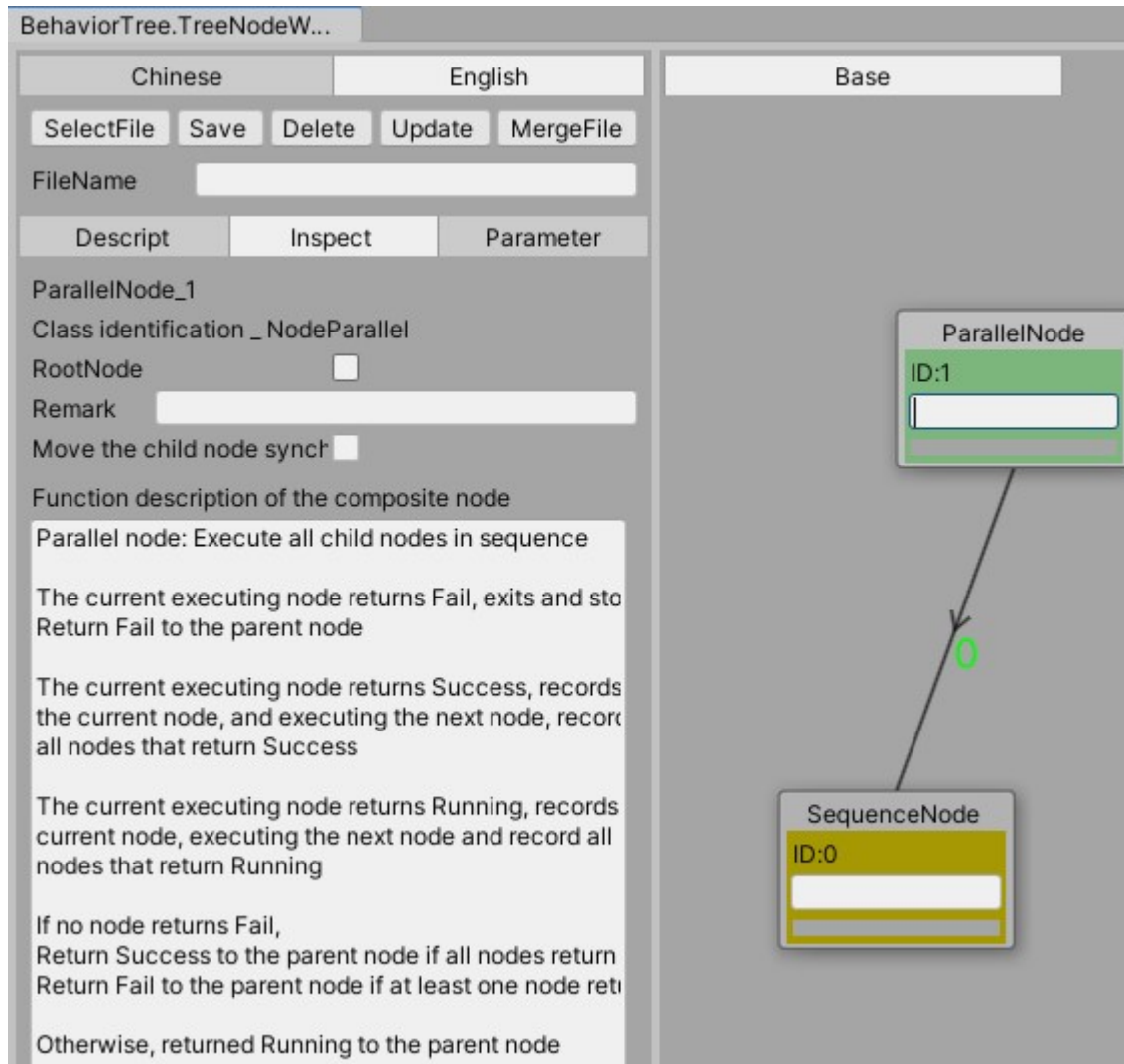


Here is a behavior tree configuration file that I edited



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

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



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:** Check the node option box in the **Inspector** , This node will act as the root node of the behavior tree, which is the entry node of the behavior tree

(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

The screenshot displays the BehaviorTree.TreeNodeW... interface. On the left, the 'Chinese' tab is active, showing the 'RandomPriority_0' node configuration. The 'Class identification _ NodeRandomPriority' is selected, and the 'RootNode' checkbox is checked. The 'Function description of the composite node' section contains the following text:

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 execute

The current executing node returns Running, recording 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;

At the bottom, the 'Child node :1 priority' is set to 1, and the 'Child node :2 priority' is set to 1.

On the right, the 'Base' tab is active, showing a tree diagram. The 'RandomPriority' node (ID:0) is the root, with two children: 'SelectNode' (ID:1) and 'SequenceNode' (ID:2). The 'SelectNode' is highlighted with a green arrow and the number 0, and the 'SequenceNode' is highlighted with a green arrow and the number 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

The screenshot displays the 'BehaviorTree.TreeNodeW...' window, which is used for configuring nodes in a behavior tree. The window is divided into two main sections: a configuration panel on the left and a visual tree diagram on the right.

Configuration Panel (Left):

- Language:** Chinese (selected), English.
- Buttons:** SelectFile, Save, Delete, Update, MergeFile.
- FileName:** [Empty text field]
- Tabs:** Descript, Inspect (selected), Parameter.
- Node Name:** IfJudgeParallel_3
- Class identification:** _ NodeIfJudgeParallel
- RootNode:** ☒
- Remark:** [Empty text field]
- NodeID:** 1
- NodeType:** if 判断节点
- NodeID:** 2
- NodeType:** 执行节点
- Execute conditional:** Fail
- Default return result:** Fail
- Default return result:** Fail
- Move the child node sync:** ☐
- 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

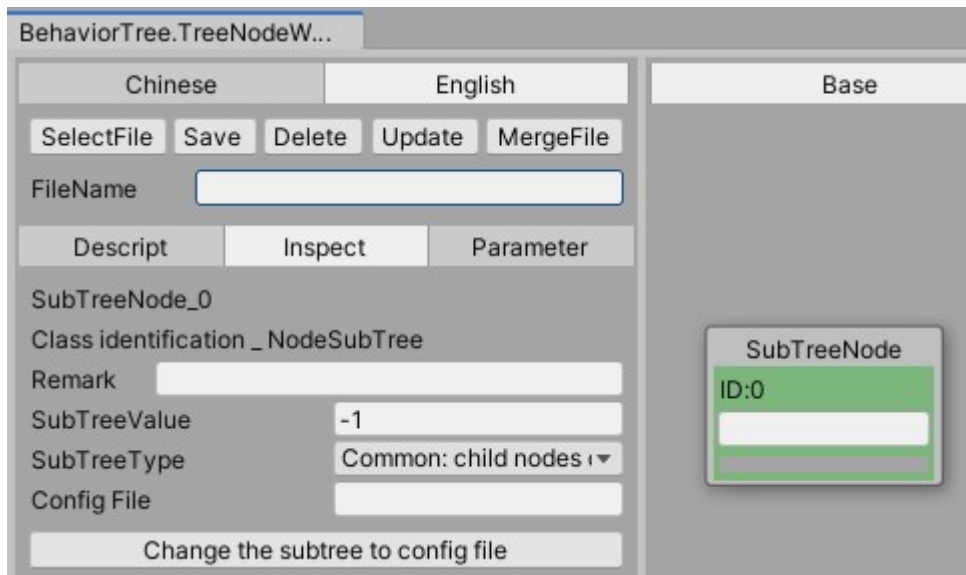
Visual Tree Diagram (Right):

- The root node is **IfJudgeParallel** (ID:3), highlighted in green.
- It has two children:
 - SelectNode** (ID:1), highlighted in yellow.
 - SequenceNode** (ID:2), highlighted in yellow.
- Arrows connect the parent node to the children, with a green '0' on the left arrow and a green '1' on the right arrow, indicating the execution flow.

(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



(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

BehaviorTree.TreeNodeW...

Chinese	English
SelectFile	Save
Delete	Update
MergeFile	

FileName

Descript	Inspect	Parameter
SubTreeNode_0		
Class identification _ NodeSubTree		
Remark <input type="text"/>		
SubTreeValue	-1	
SubTreeType	Configuration: Reads t ▼	
Config File	EneryEnoughSubTree	
Select the subtree config file		

Base

SubTreeNode

ID:0

(6.5) Action Node

BehaviorTree.TreeNodeW...

Chinese	English
SelectFile	Save
Delete	Update
MergeFile	

FileName

Descript	Inspect	Parameter
Action_0		
Player/攻击_0		
Class identification _ PlayerAttackAction		
Remark <input type="text"/>		
Parameter		
0	Bool	IsSurvial
Player/活着	Equal	<input type="checkbox"/>
1	Bool	HasEneny
Player/有敌人	Equal	<input type="checkbox"/>
2	Int	TargetType
Player/目标类型 : 敌	GreaterOr	0
Add Condition		

Base

Player/攻击

ID: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



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

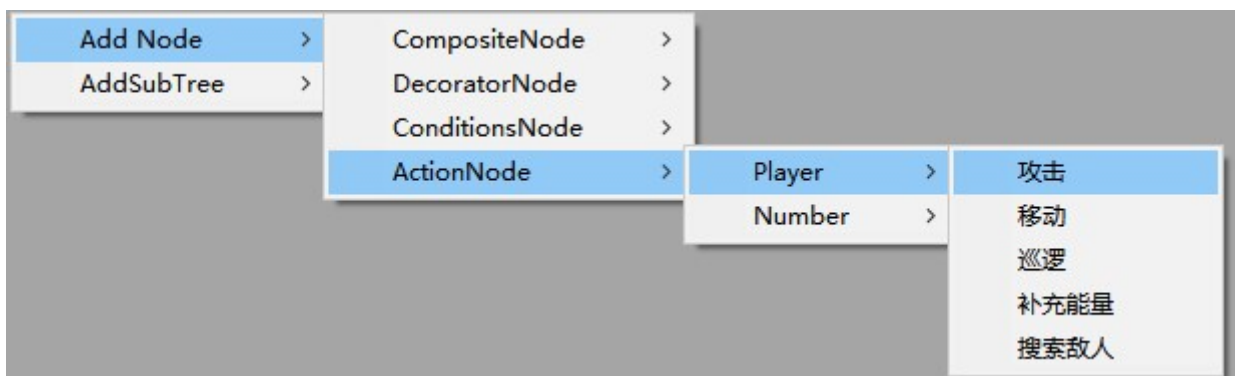
Otherwise return Fail

(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 Action 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 `extends` `ActionBase`

ConditionNode `extends` `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` `extends` `NodeSubTreeDynamicBase`

`override CalculateNewSubTree()` function. In this function, determine which subtree

configuration file is currently being used and call `SetSubTreeConfig(string config)` function. For example,

```

1 protected override void CalculateNewSubTree()
2 {
3     if (level <= 50)
4     {
5         SetSubTreeConfig("npc_50_subTree");
6     }
7     else if (level <= 60)
8     {
9         SetSubTreeConfig("npc_60_subTree");
10    }
11    else if (level <= 70)
12    {
13        SetSubTreeConfig("npc_70_subTree");
14    }
15    else
16    {
17        SetSubTreeConfig("npc_power_subTree");
18    }
19 }

```

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

(7) `ConfigLoad` to load the config file

10. The example scene

Open Demo\Human.unity and run

This is my email account: qiang.li.9631@gmail.com

Please email if you have any questions