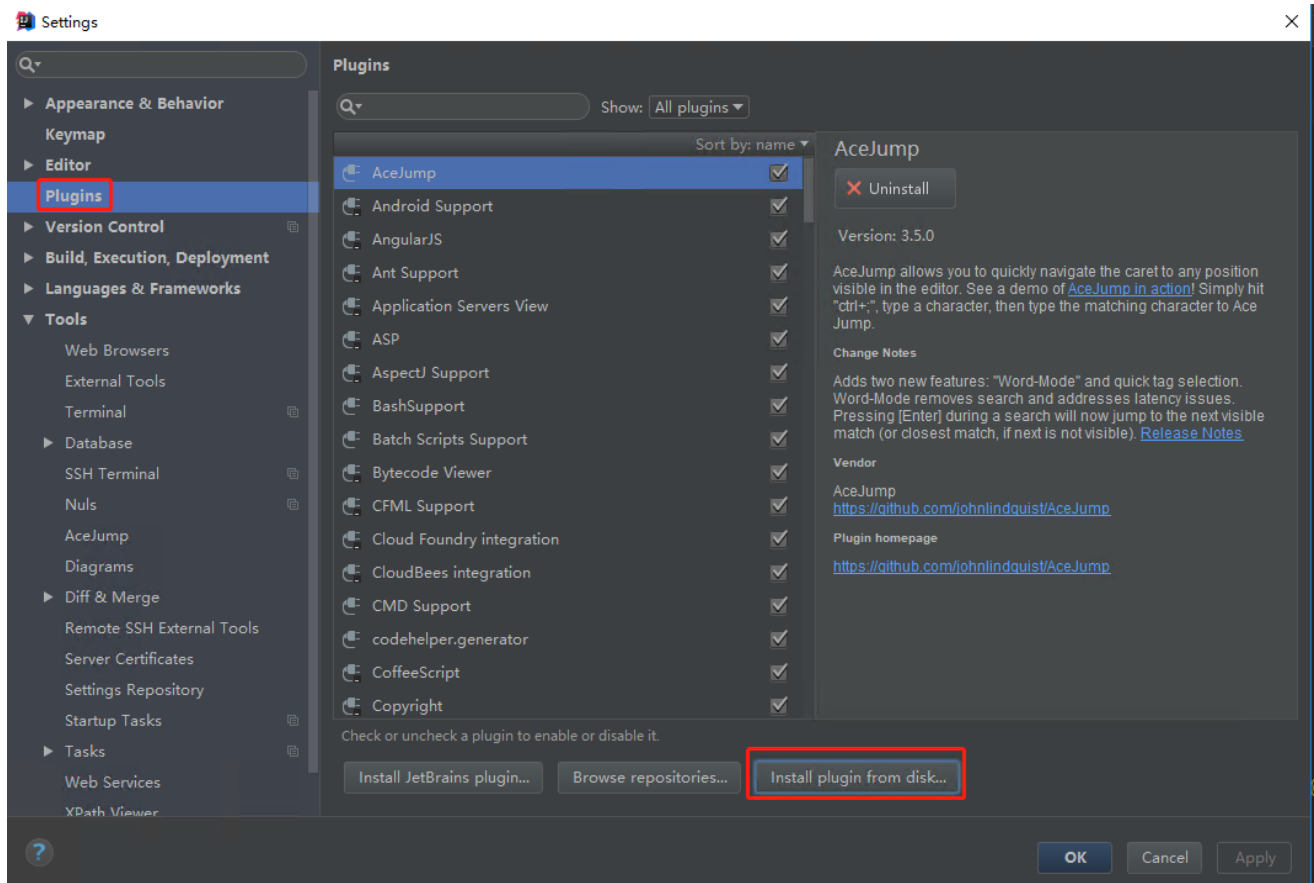


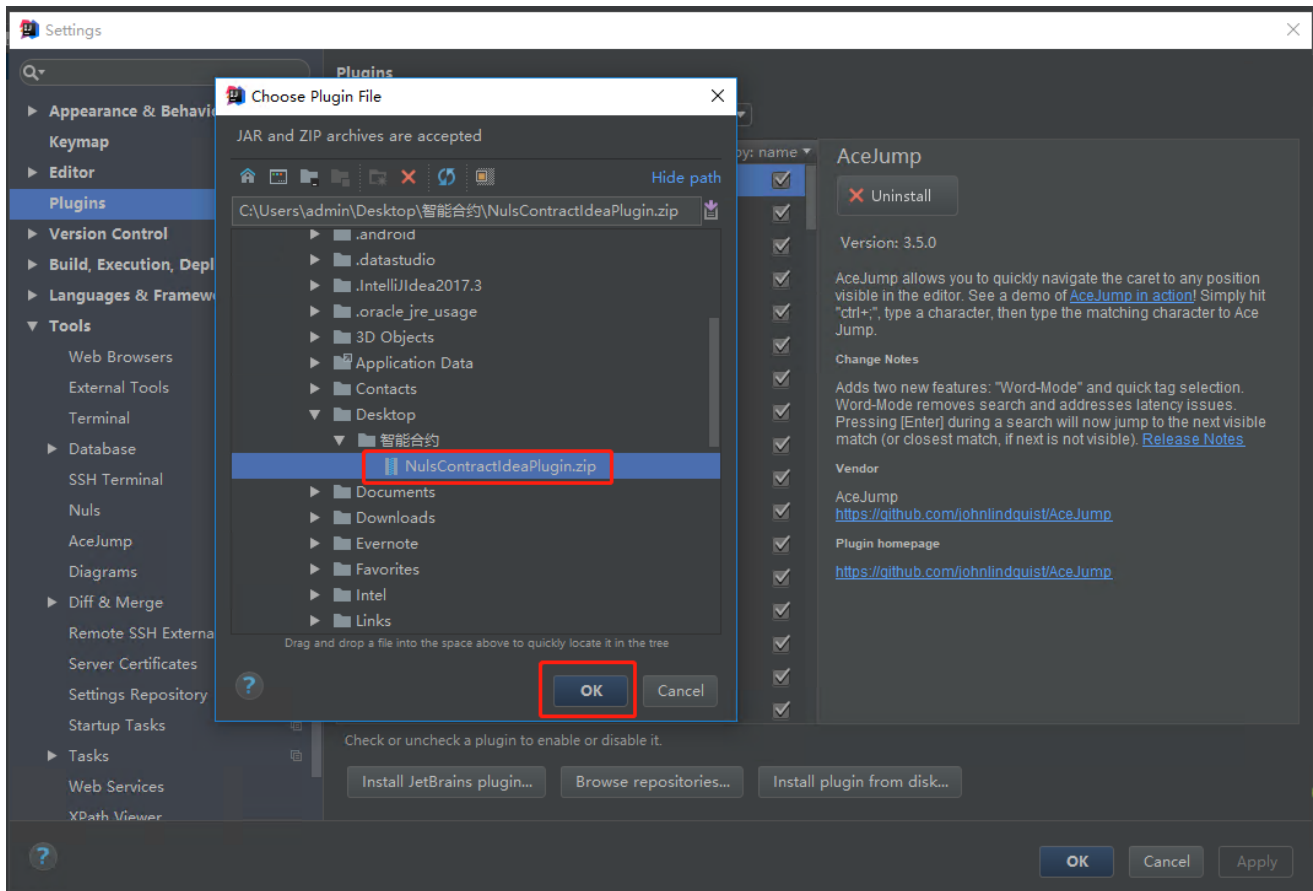
NULS IDEA plugin manual

1 Install the NULS plugin in IDEA

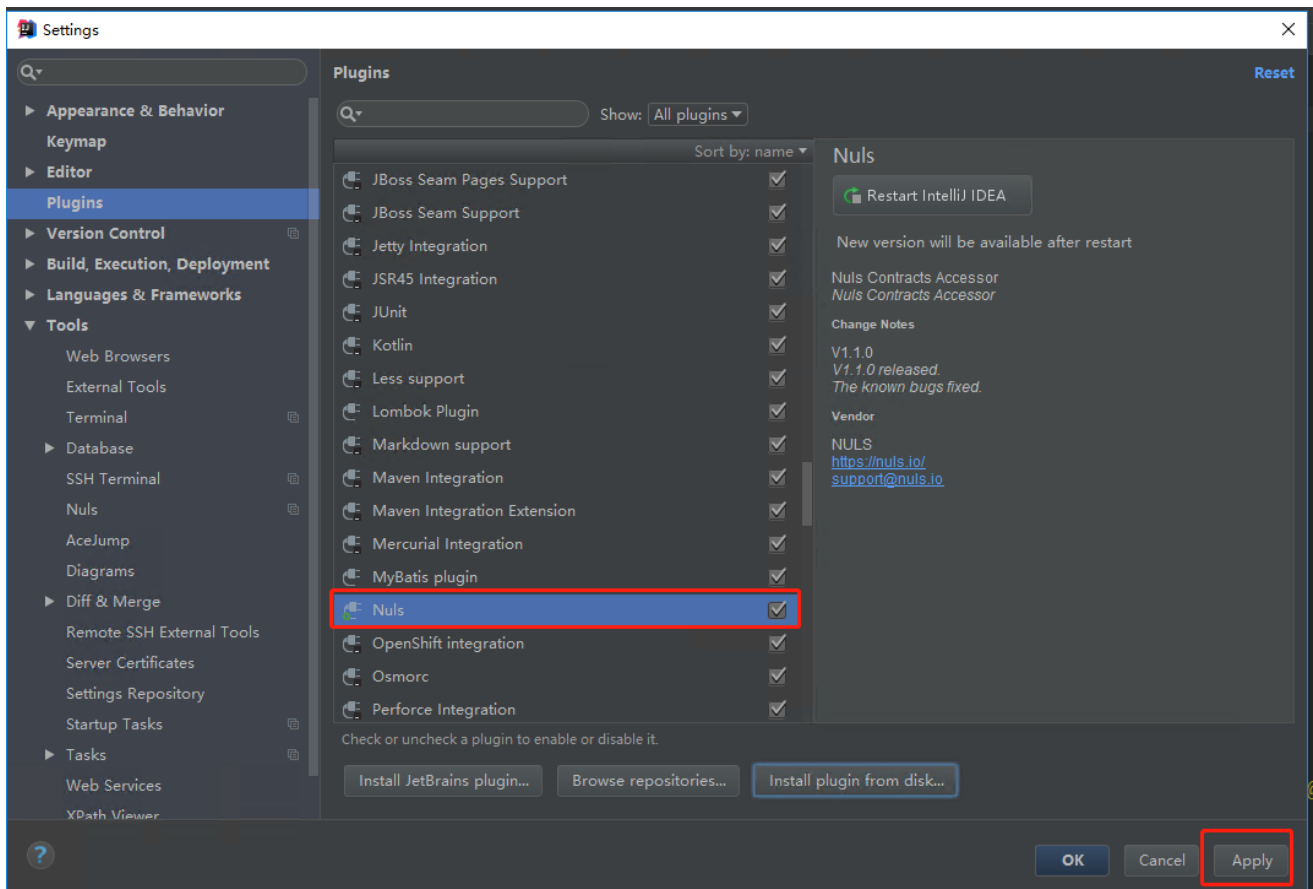
- Get the NULS plugin ZIP package and store it on disk and Click File->Settings->Plugin->Install plugin from disk



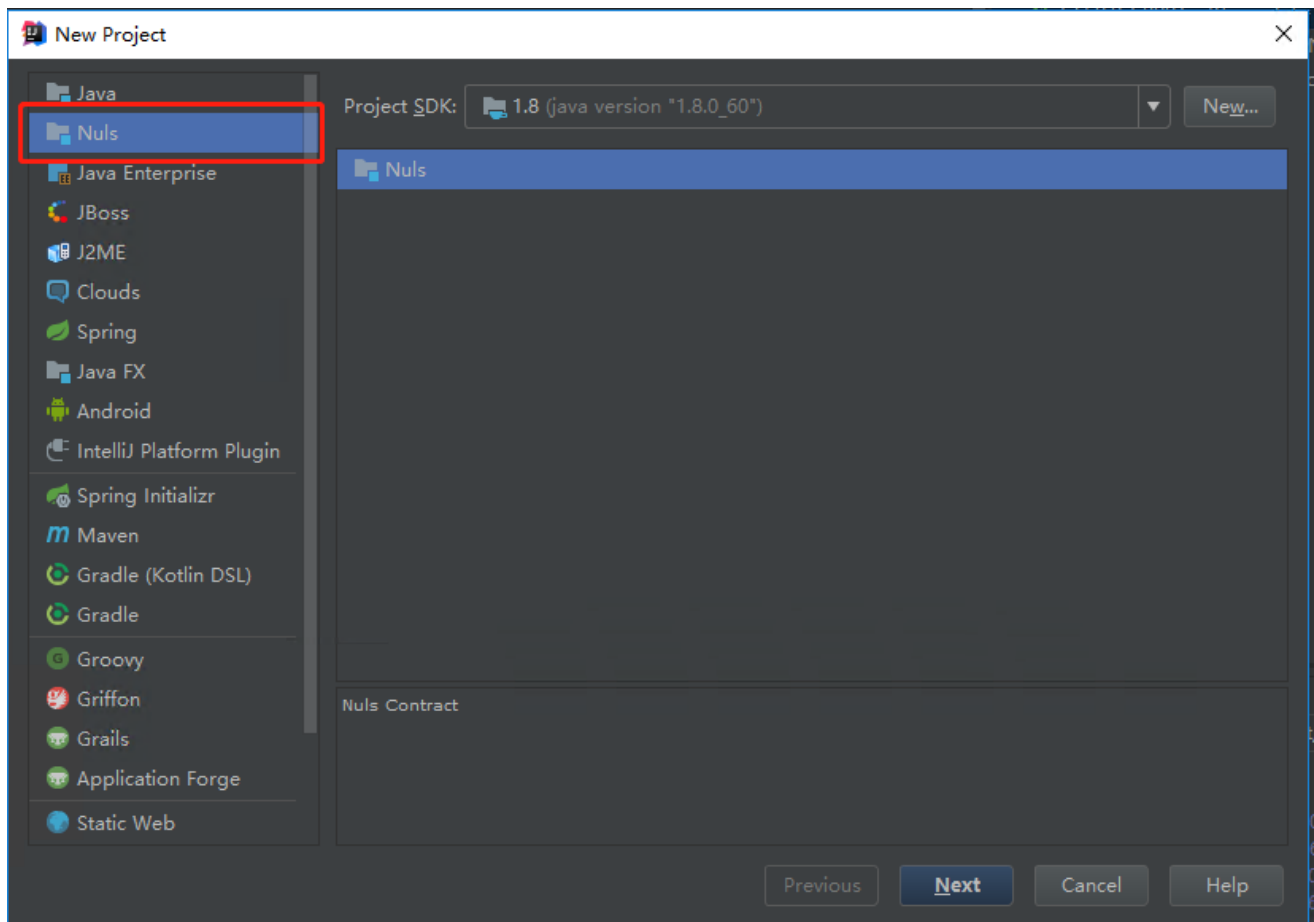
- Select the ZIP package you obtained earlier, then click OK



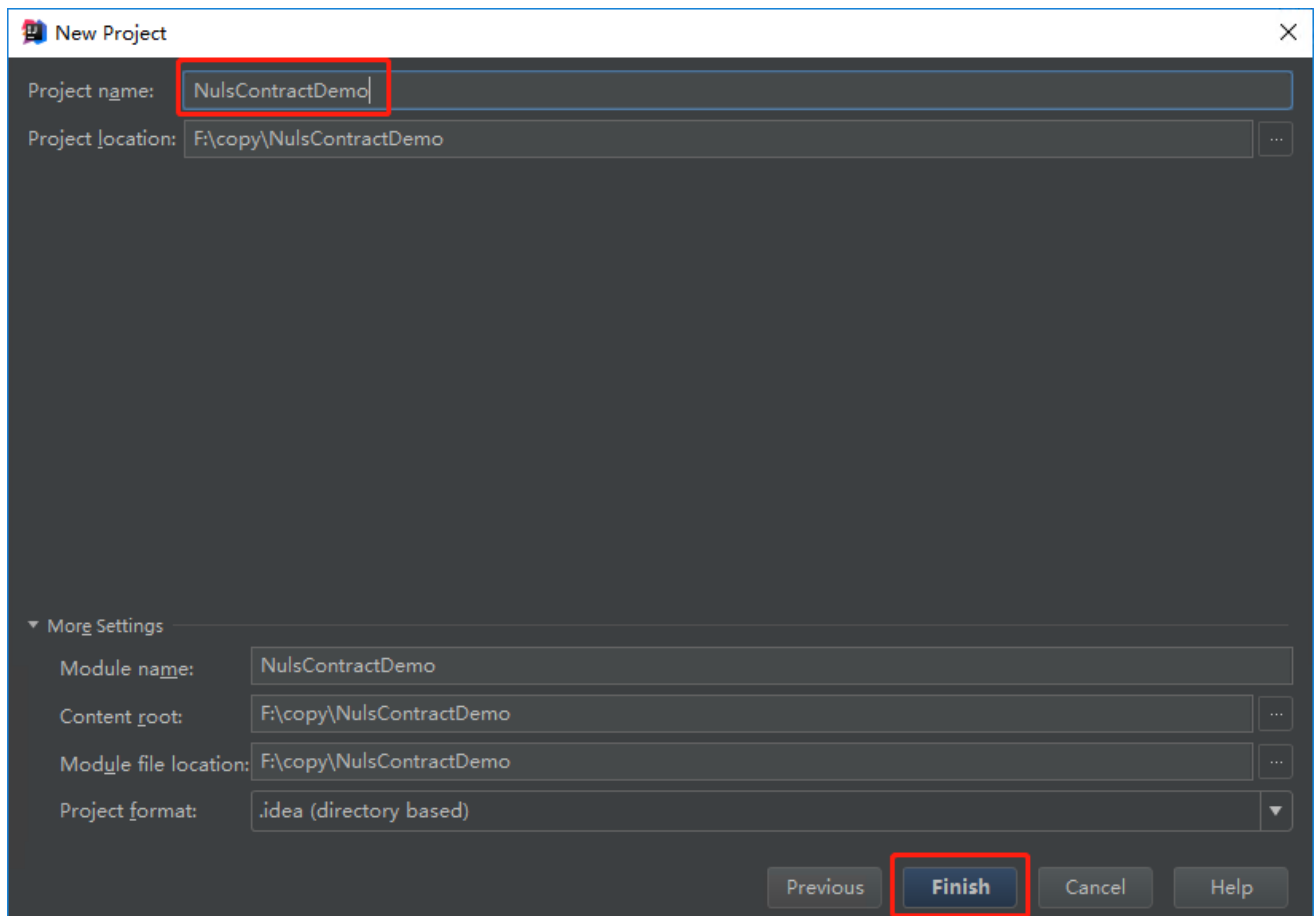
- Check the NULS plugin and click Apply



2 Create a NULS project

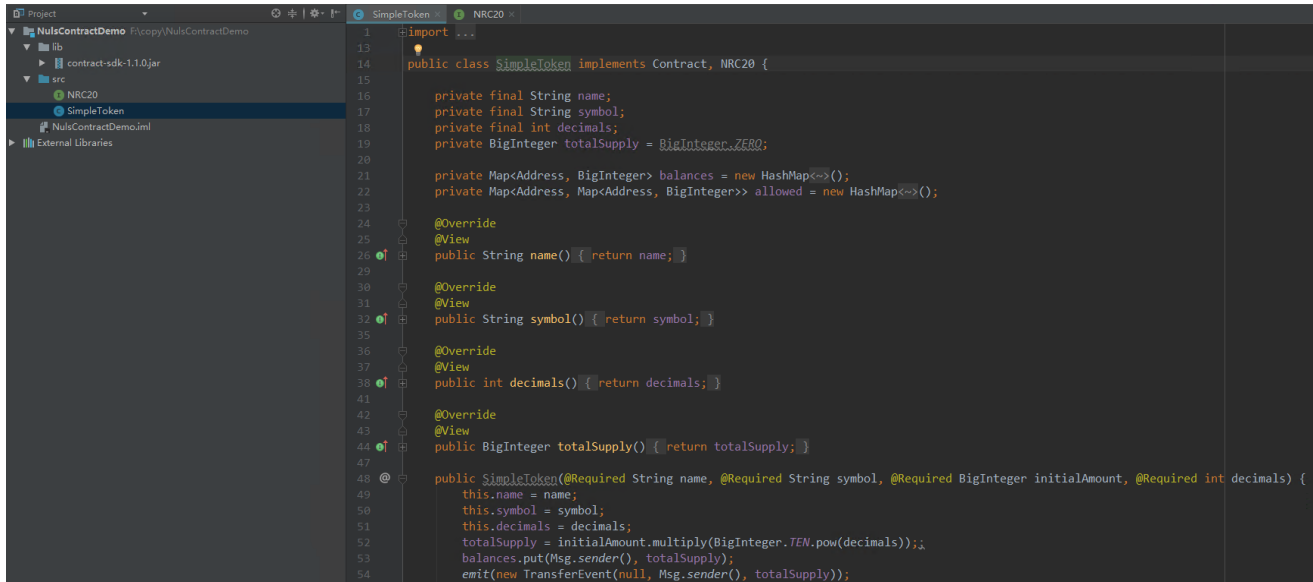


Enter the Project name and click finish



3 Writing smart contracts

Smart contract code writing can be viewed [developer documentation](#)

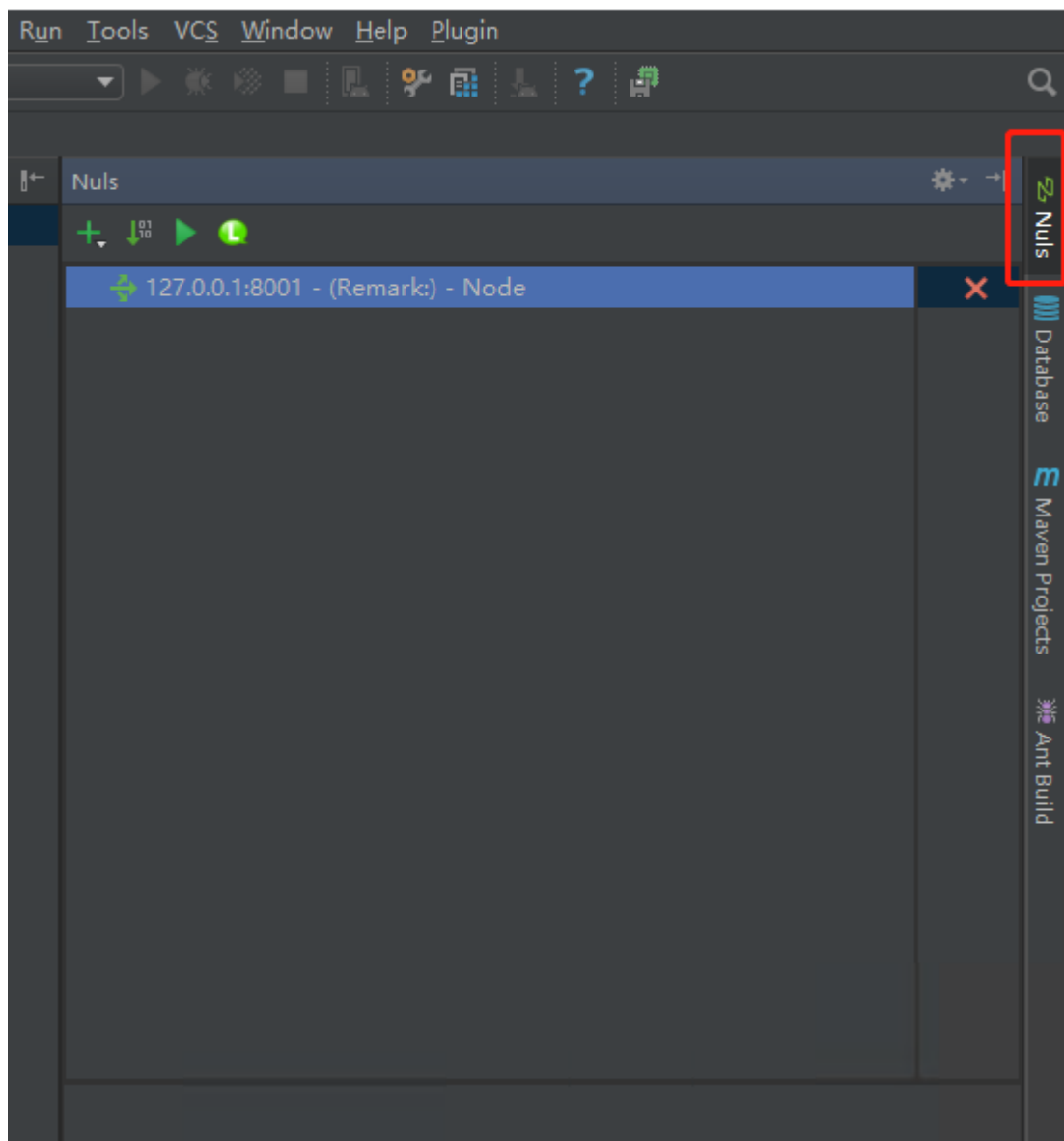


The screenshot shows an IDE with a project named 'NulsContractDemo'. The left sidebar displays the project structure, including 'lib', 'contract-sdk-1.1.0.jar', 'src', 'NRC20', 'SimpleToken', 'NulsContractDemo.iml', and 'External Libraries'. The main editor window shows the 'SimpleToken' class, which implements the 'Contract' and 'NRC20' interfaces. The code includes private fields for 'name', 'symbol', 'decimals', and 'totalSupply', and methods for 'name()', 'symbol()', 'decimals()', and 'totalSupply()'. The constructor 'SimpleToken' takes parameters for 'name', 'symbol', 'initialAmount', and 'decimals', and initializes the 'totalSupply' and 'balances' maps. The 'emit' method is used to trigger a 'TransferEvent'.

```
1  import ...
13
14  public class SimpleToken implements Contract, NRC20 {
15
16      private final String name;
17      private final String symbol;
18      private final int decimals;
19      private BigInteger totalSupply = BigInteger.ZERO;
20
21      private Map<Address, BigInteger> balances = new HashMap<>();
22      private Map<Address, Map<Address, BigInteger>> allowed = new HashMap<>();
23
24      @Override
25      @View
26      public String name() { return name; }
27
28      @Override
29      @View
30      public String symbol() { return symbol; }
31
32      @Override
33      @View
34      public int decimals() { return decimals; }
35
36      @Override
37      @View
38      public BigInteger totalSupply() { return totalSupply; }
39
40      public SimpleToken(@Required String name, @Required String symbol, @Required BigInteger initialAmount, @Required int decimals) {
41          this.name = name;
42          this.symbol = symbol;
43          this.decimals = decimals;
44          totalSupply = initialAmount.multiply(BigInteger.TEN.pow(decimals));
45          balances.put(Msg.sender(), totalSupply);
46          emit(new TransferEvent(null, Msg.sender(), totalSupply));
47      }
48  }
```

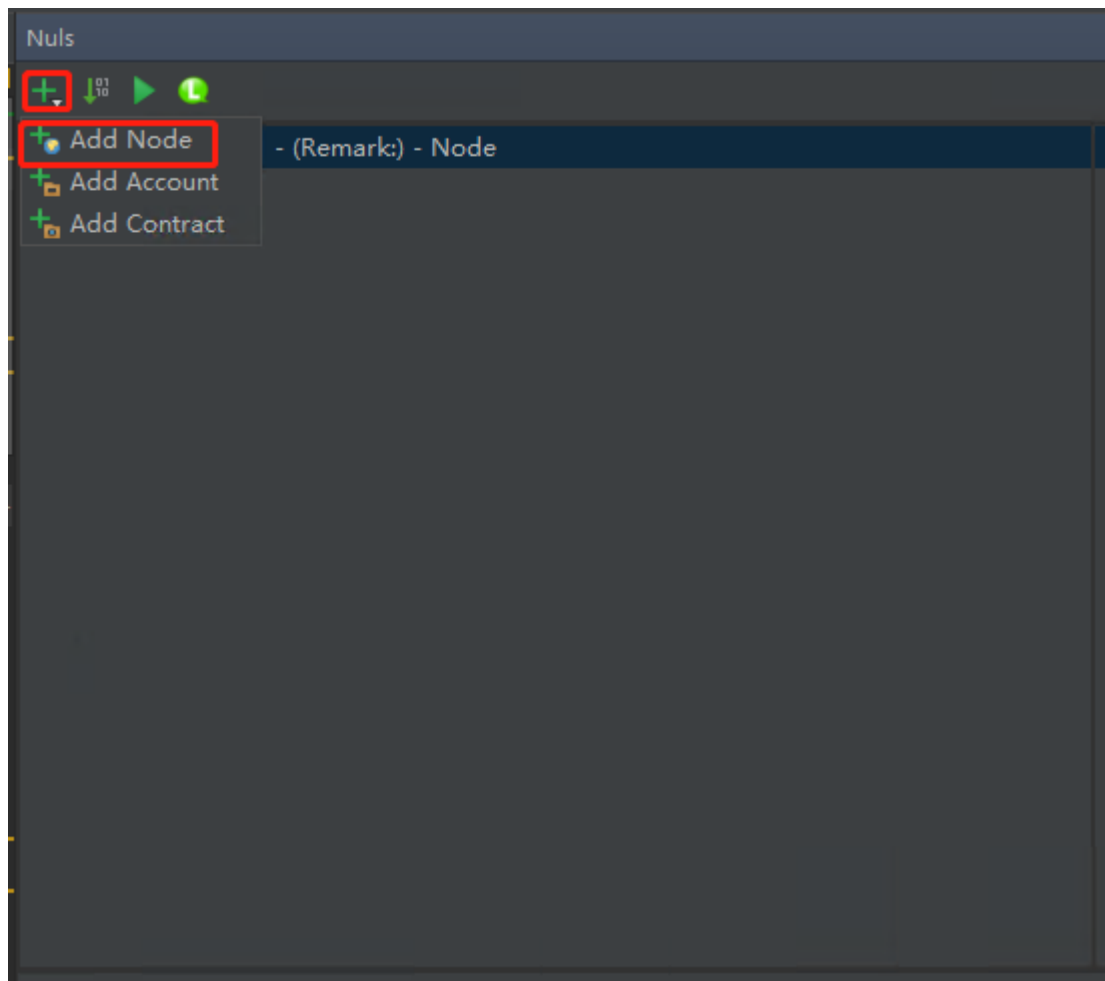
4 Set up nodes and accounts for deployment contracts

- Click the NULS plugin on the right to bring up the NULS plugin panel

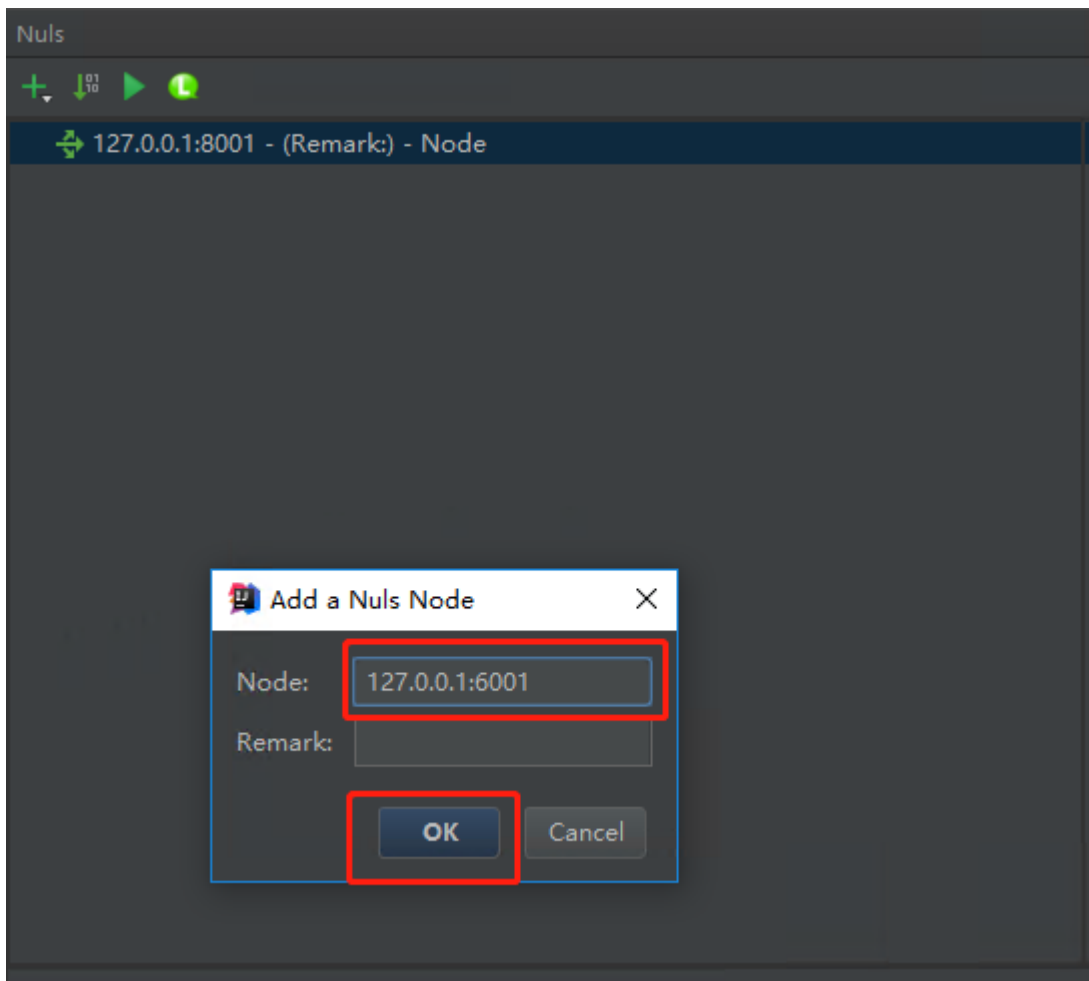


- The plugin will automatically generate a 127.0.0.1:8001 node, and the developer can add a node to deploy the contract. The recommended method is to start the wallet locally and then add the address of the wallet as the node address.

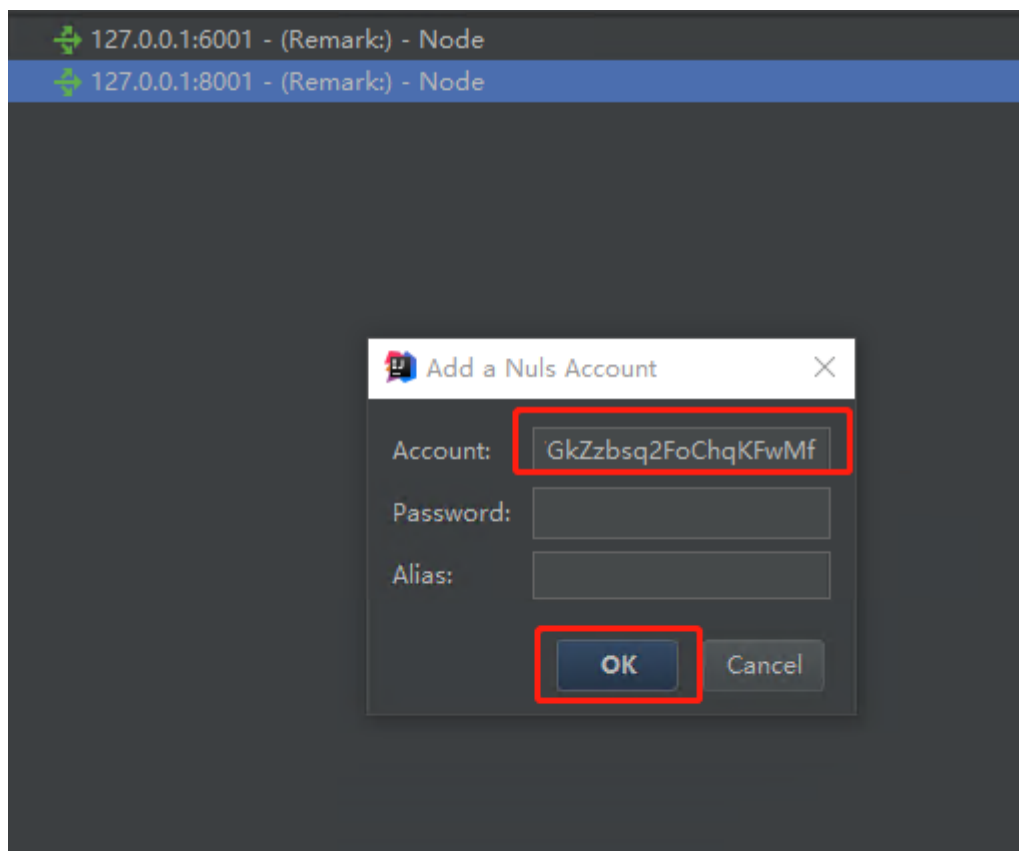
Click the + sign in the upper left corner of the panel and select Add Node.



- Enter the Node address and click OK

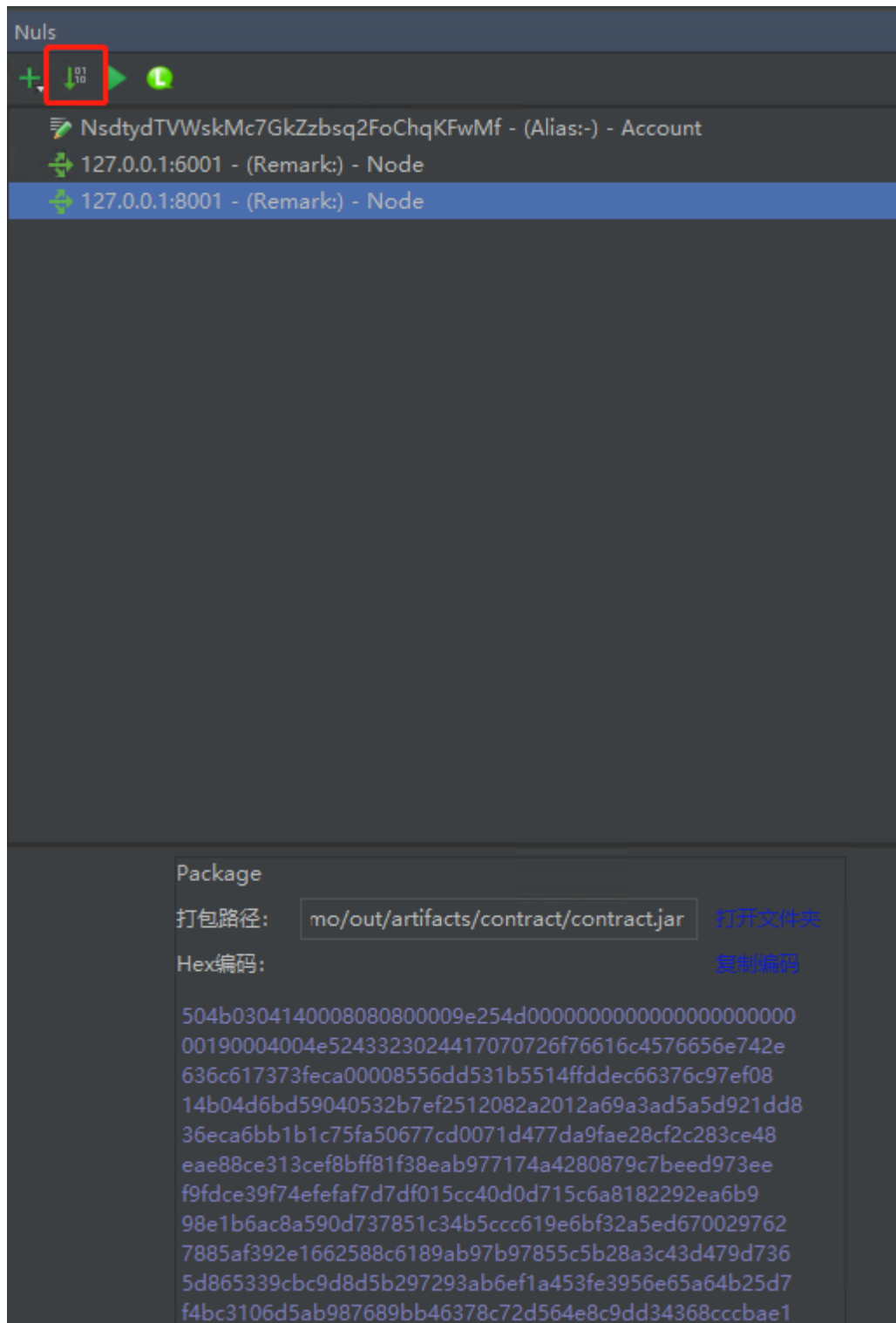


- Click the + sign in the upper left corner of the panel, select Add Account, enter the Account address, and click OK.

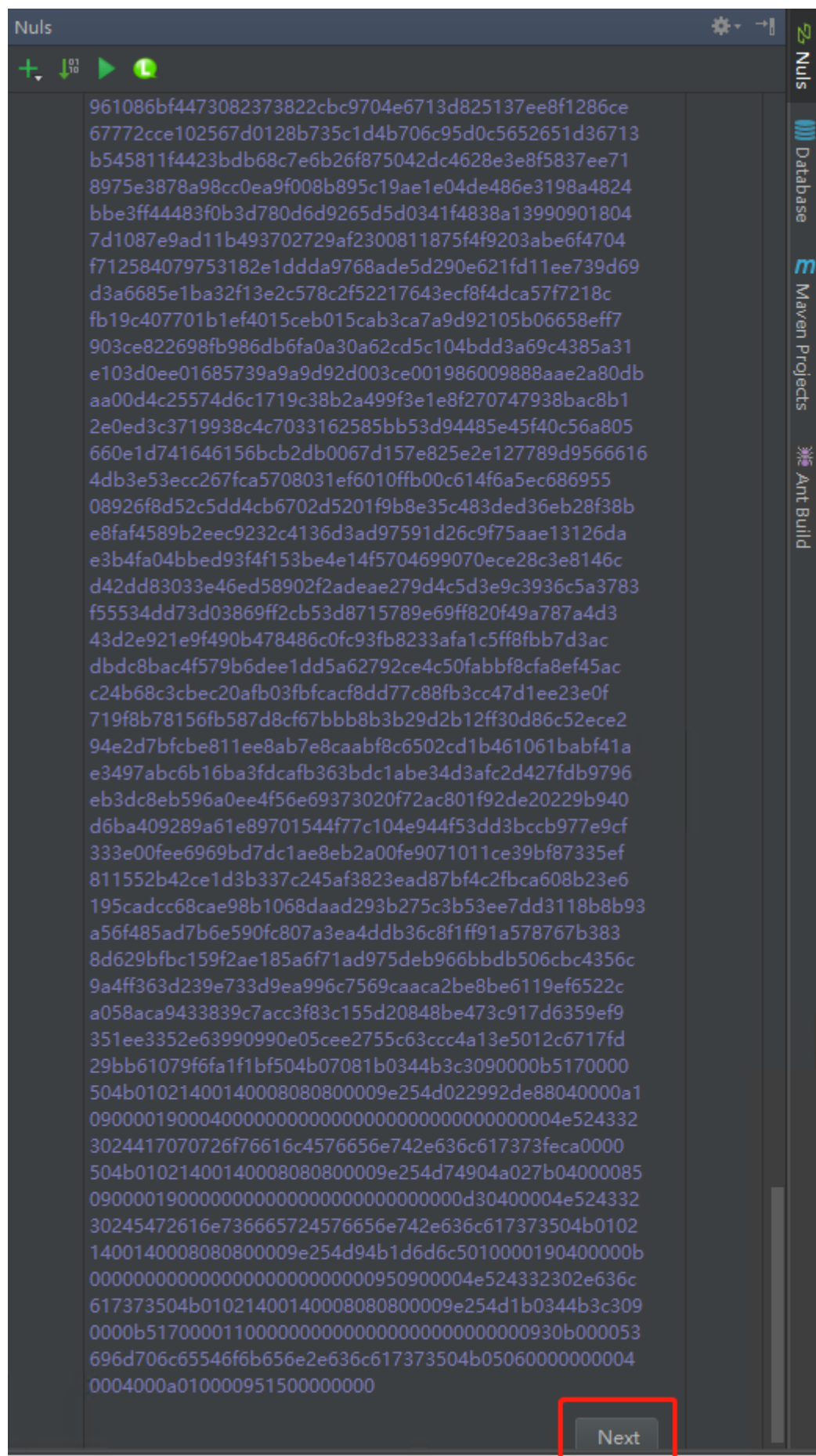


5 Package contract

- Click the second button at the top left of the plugin to package the current Project directly. The packaged output has two forms: Jar and Hex



- Click [Copy Coding] to copy the HEX code to the [Deployment Contract] interface of the wallet for contract deployment (optional step)



6 Deployment contract

- On the deployment page, you can select the node and account for the deployment contract from the drop-down list. The default value of JarFilePath is the packaging path of the previous step.

The screenshot shows the Nuls deployment interface. At the top, there is a list of nodes and accounts. The first row is 'NsdtydTVWskMc7GkZzbsq2FoChqKfWMf - (Alias:-) - Account'. The next two rows are '127.0.0.1:6001 - (Remark:-) - Node' and '127.0.0.1:8001 - (Remark:-) - Node'. The third row is highlighted. To the right of each row is a red 'X' icon. Below the list, there is a form for deployment configuration. The 'Node:' dropdown is set to '127.0.0.1:8001'. The 'Account:' dropdown is set to 'NsdtydTVWskMc7GkZzbsq2FoChqKfWMf'. The 'JarFilePath:' text field contains 'F:/copy/NulsContractDemo/out/artifacts/con'. Below these are fields for 'name:', 'symbol:', 'initialAmount:', and 'decimals:'. There is a checkbox labeled '高级' (Advanced) and a 'TxnHash' field. At the bottom right, there are 'Test Deploy' and 'Deploy' buttons.

Node/Account	Remark	Type
NsdtydTVWskMc7GkZzbsq2FoChqKfWMf	(Alias:-)	Account
127.0.0.1:6001	(Remark:-)	Node
127.0.0.1:8001	(Remark:-)	Node

Node: 127.0.0.1:8001

Account: NsdtydTVWskMc7GkZzbsq2FoChqKfWMf

JarFilePath: F:/copy/NulsContractDemo/out/artifacts/con

name:

symbol:

initialAmount:

decimals:

☐ 高级

TxnHash

Test Deploy

Deploy

- The parameter immediately following JarFilePath is the parameter of the contract constructor.

The screenshot shows the Nuls IDE interface. At the top, there's a header bar with the 'Nuls' logo and some icons. Below it is a toolbar with a plus sign, a download icon, a play icon, and a lightbulb icon. The main area is a list of nodes, each with a green icon, a text label, and a red 'X' icon. The nodes are:

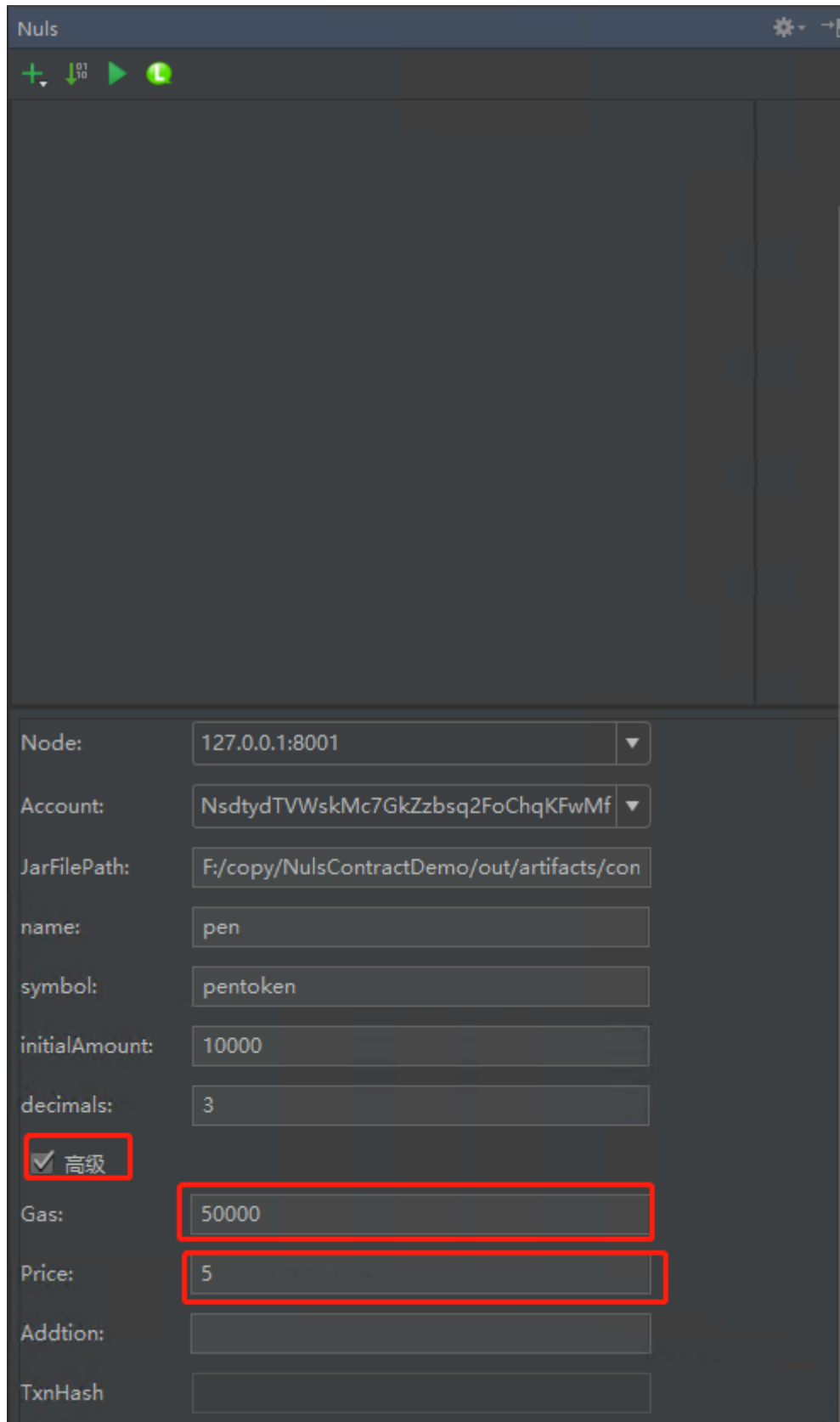
- NsdydTVWskMc7GkZzbsq2FoChqKfWmf - (Alias:-) - Account
- 127.0.0.1:6001 - (Remark:-) - Node
- 127.0.0.1:8001 - (Remark:-) - Node

On the right side, there's a sidebar with icons for 'Nuls', 'Database', 'Maven Projects', and 'Ant Build'. Below the node list, there's a form for contract deployment. The form has the following fields:

- Node: 127.0.0.1:8001
- Account: NsdydTVWskMc7GkZzbsq2FoChqKfWmf
- JarFilePath: F:/copy/NulsContractDemo/out/artifacts/con
- name: (empty)
- symbol: (empty)
- initialAmount: (empty)
- decimals: (empty)
- ☐ 高級
- TxnHash: (empty)

At the bottom right, there are two buttons: 'Test Deploy' and 'Deploy'. A red rectangle highlights the 'name', 'symbol', 'initialAmount', and 'decimals' fields.

- Click [Advanced] to set the Gas value and the price value. The Gas value range is 1-10000000. It is recommended to set the Gas value to be large to avoid the failure of the Gas to cause the deployment contract to fail.



The screenshot shows the Nuls IDE interface with a dark theme. At the top, there's a title bar with 'Nuls' and some icons. Below it, a toolbar contains a green plus icon, a green minus icon, a green play icon, and a green lightbulb icon. The main area is a large, empty dark rectangle. At the bottom, there's a configuration panel with various fields and a checkbox.

Node:	127.0.0.1:8001
Account:	NsdydTVWskMc7GkZzbsq2FoChqKFwMf
JarFilePath:	F:/copy/NulsContractDemo/out/artifacts/con
name:	pen
symbol:	pentoken
initialAmount:	10000
decimals:	3
<input checked="" type="checkbox"/> 高级	
Gas:	50000
Price:	5
Addition:	
TxnHash	

- Click [Test Deploy] to deploy the test to the contract. If successful, it will return the Success message.

Nuls

+ 01 10 ▶ L

Node: 127.0.0.1:8001 ▼

Account: NsdydTVWskMc7GkZzbsq2FoChqKFwMf ▼

JarFilePath: F:/copy/NulsContractDemo/out/artifacts/con

name: pen

symbol: pentoken

initialAmount: 10000

decimals: 3

☒ 高級

Gas: 50000

Price: 5

Addtion:

TxnHash

Success!

Test Deploy

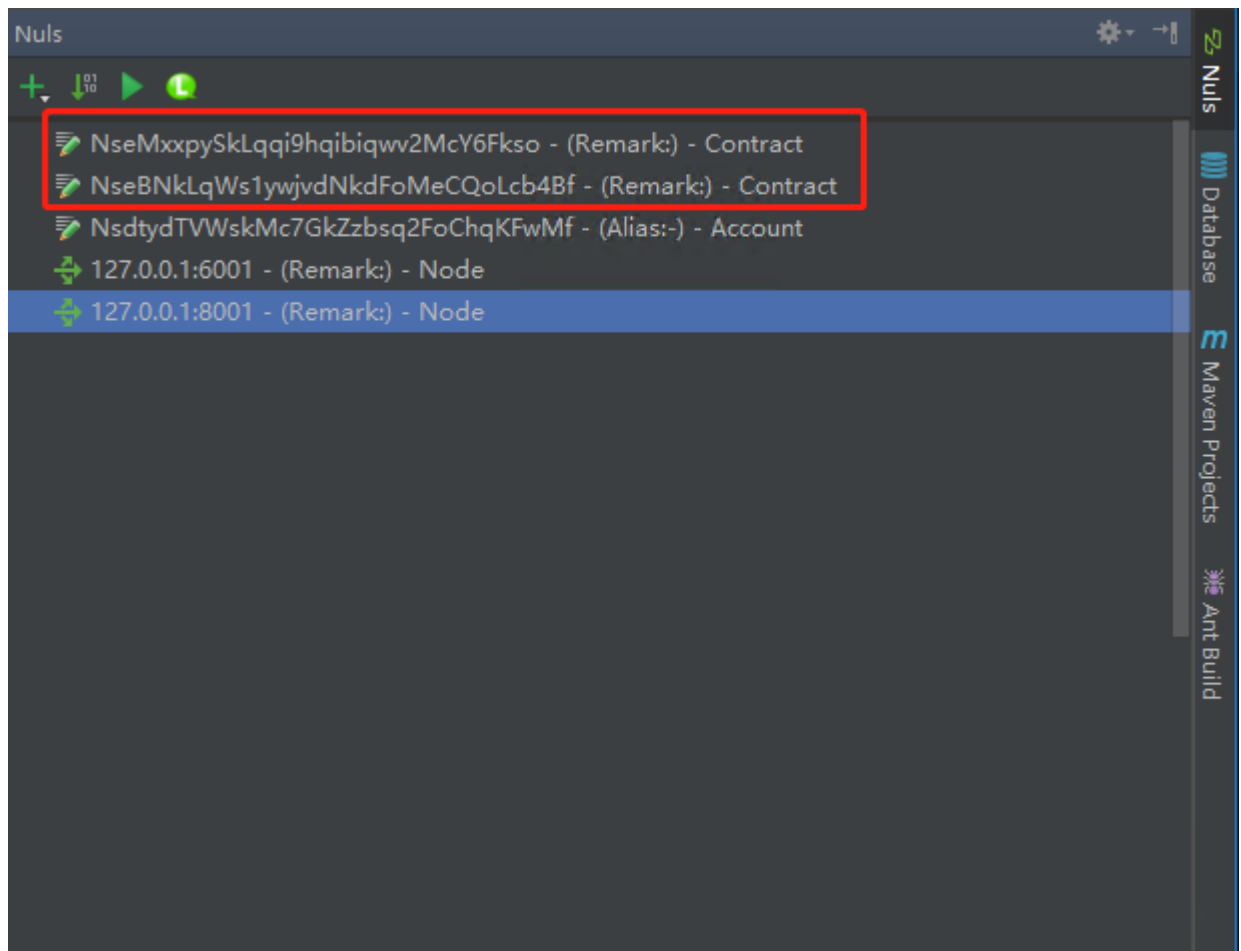
Deploy

- If the test is successful, click [Deploy], after the transaction confirmation of the created contract is successful, the transaction details will be returned, indicating that the contract is successfully deployed.

The screenshot shows the Nuls IDE interface. The main editor displays a JSON object representing a transaction. The right sidebar contains icons for Nuls, Database, Maven Projects, and Ant Build.

```
{
  "success": true,
  "data": {
    "hash": "002080dae1eebfaf0a7092b604b91ac285337094bad5499c53bf73a49a6f4399f366",
    "type": 100,
    "time": 1536207160439,
    "blockHeight": 8435,
    "fee": 703560,
    "value": 0,
    "remark": null,
    "scriptSig": "2102487a1fbff4cff1b99544fbdf582c12be4668aa2311e69daffe1f79550e84c127004730450",
    "status": 1,
    "confirmCount": 0,
    "size": 6108,
    "inputs": [
      {
        "fromHash": "0020eb700b735ef65afaddf8e2bfff69a04cee2efd4925120b70222f18e2768e63920",
        "fromIndex": 0,
        "address": "NsdtydTVWskMc7GkZzbsq2FoChqKFwMf",
        "value": 998939994247680
      }
    ],
    "outputs": [
      {
        "txHash": "002080dae1eebfaf0a7092b604b91ac285337094bad5499c53bf73a49a6f4399f366",
        "index": 0,
        "address": "NsdtydTVWskMc7GkZzbsq2FoChqKFwMf",
        "value": 998939993544120,
        "lockTime": 0,
        "status": 0
      }
    ]
  }
}
```

- View deployed contracts above the panel



- Click on the successful deployment contract to view all the methods of the contract

Nuls

NseMxxpySkLqqi9hqibiqwv2McY6Fkso - (Remark:) - Contract

NseBNkLqWs1ywjvdNkdFoMeCQoLcb4Bf - (Remark:) - Contract

NsdydTVWskMc7GkZzbsq2FoChqKFwMf - (Alias:-) - Account

127.0.0.1:6001 - (Remark:) - Node

127.0.0.1:8001 - (Remark:) - Node

Node: 127.0.0.1:8001

Account: NsdydTVWskMc7GkZzbsq2FoChqKFwMf

Contract: NseMxxpySkLqqi9hqibiqwv2McY6Fkso

TxnHash

Got Contract:NseMxxpySkLqqi9hqibiqwv2McY6Fkso
Method Count:14,
Methond Names:-

name - () return String
symbol - () return String
decimals - () return int
totalSupply - () return BigInteger
<init> - (String name, String symbol, BigInteger initialAmount, int decimals) return void
allowance - (Address owner, Address spender) return BigInteger
transferFrom - (Address from, Address to, BigInteger value) return boolean
balanceOf - (Address owner) return BigInteger