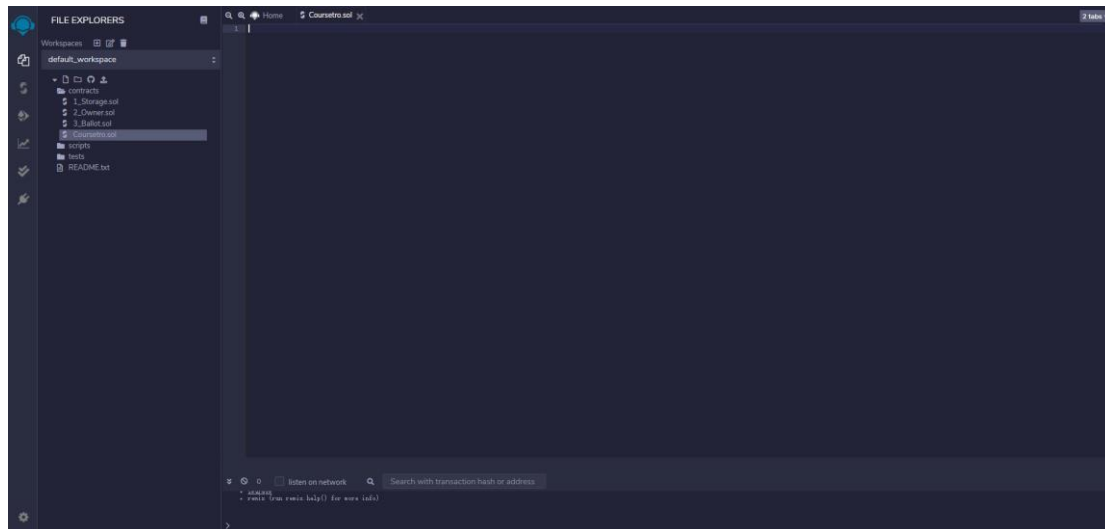
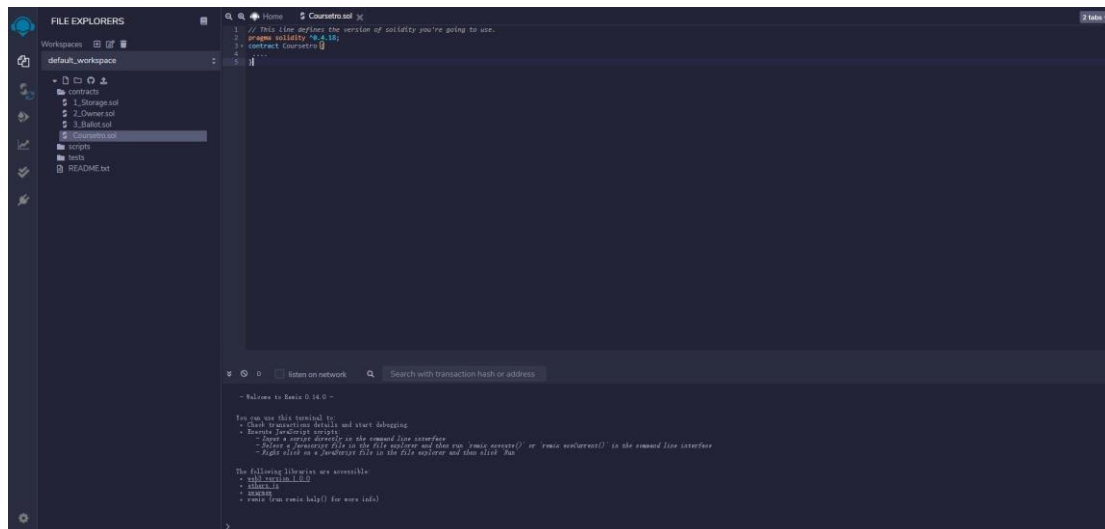


Go to the website: <http://remix.ethereum.org>

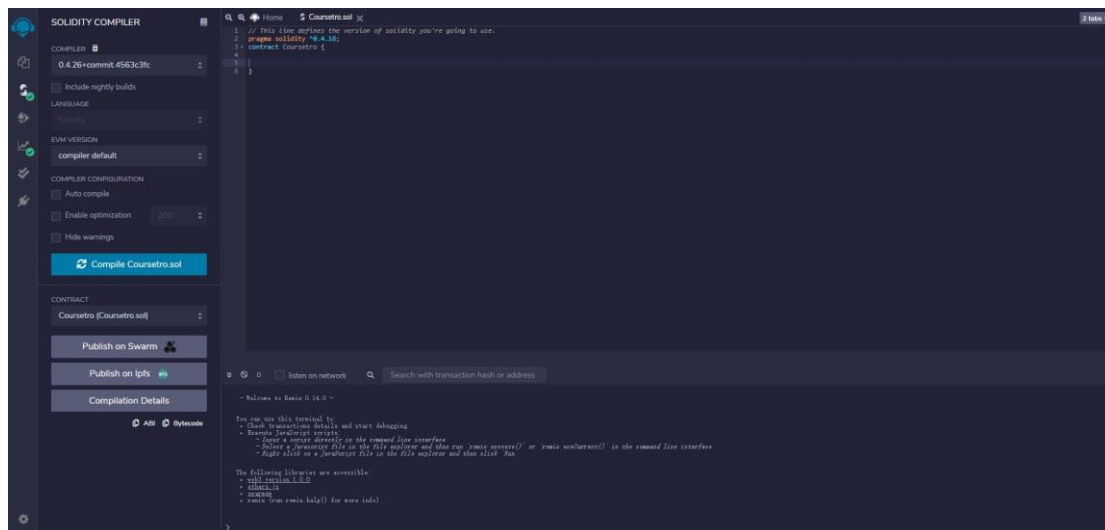
And create a file named **Coursetro.sol**



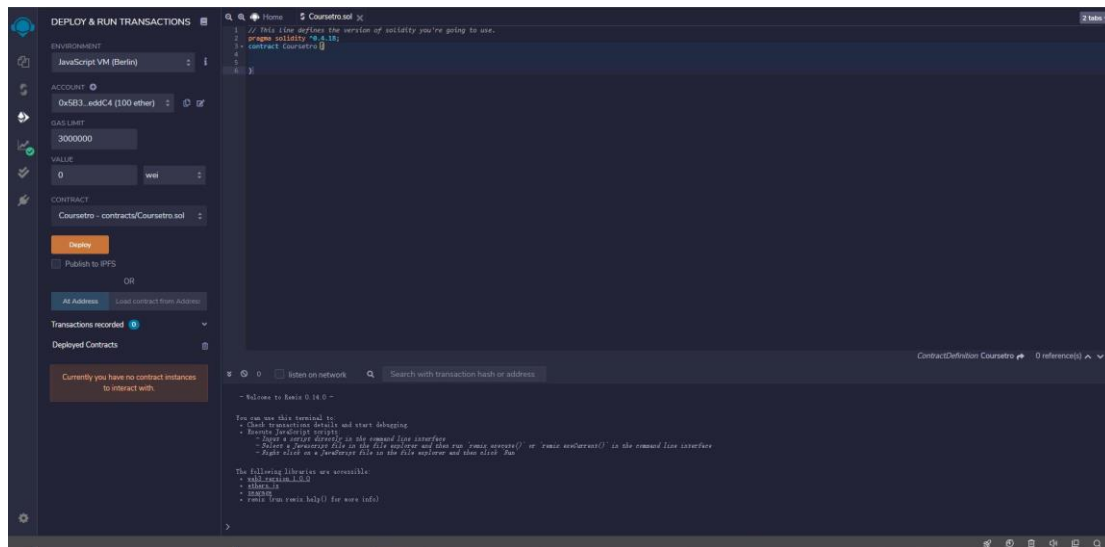
We can enter at the right side



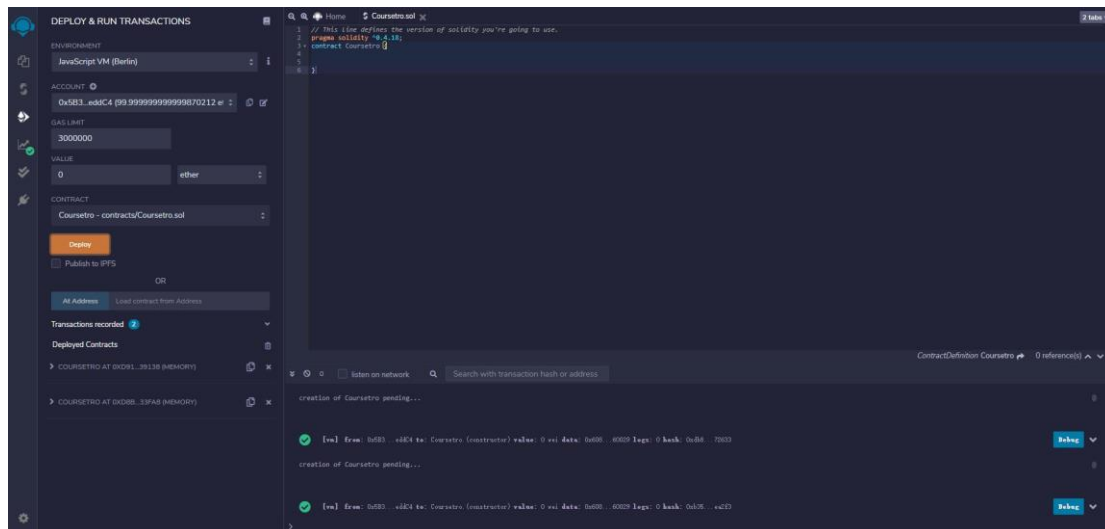
We can go to **SOLIDITY COMPILER**(on the left) and click **Compile**



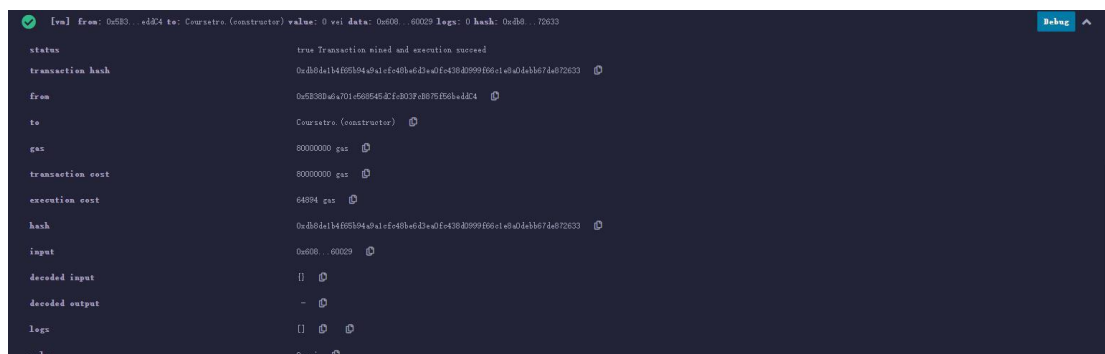
Then go to **DEPLOY & RUN TRANSACTIONS** (on the left), and make the setting



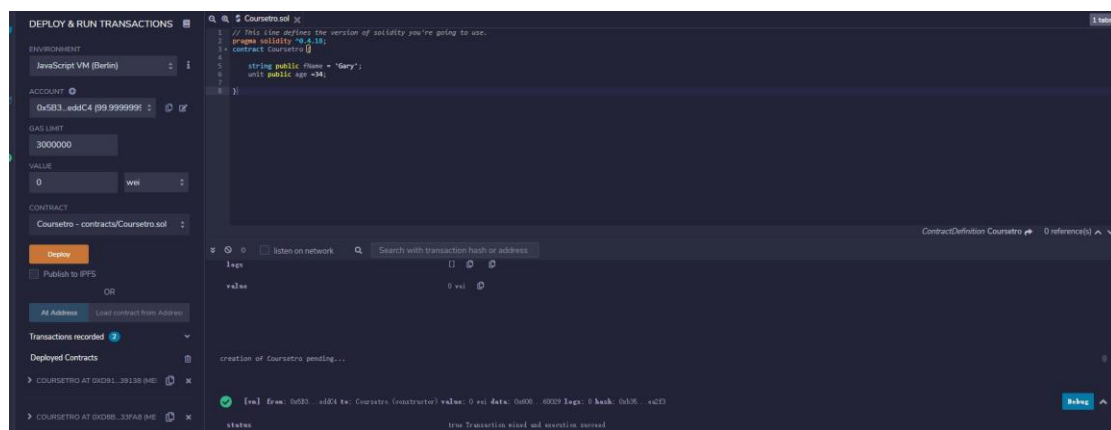
Click **Deploy**, run to test it



We can see the running detail on the bottom



In Solidity, you define a variable by first specifying its type



Let's do the testing now !!!!!!!

Write down these in the right:

```
pragma solidity ^0.4.18;
```

```
contract Coursetro {
```

```
    string fName;
```

```
    uint age;
```

```
    function Coursetro() public {
```

```
        fName = 'Gary';
```

```
        age = 34;
```

```
    }
```

```
    // setInstructor accepts 2 parameters, _fName and _age.
```

```
    // Once called, we set our string fName to the returned
```

```
    // _fName, and same with age.
```

```
    function setInstructor(string _fName, uint _age) public {
```

```
        fName = _fName;
```

```
        age = _age;
```

```
    }
```

```
    // The getInstructor() function is defined as being
```

```
    // constant, and it returns a string and a uint. This is where
```

```
    // we return the fName and age variable once it's called.
```

```
    function getInstructor() public constant returns (string, uint) {
```

```
        return (fName, age);
```

```
    }
```

```
}
```

Then we can click **Deploy**, but make sure click **Compile (SOLIDITY COMPILER)**

After the **setInstructor** button, we can type in to testing

Then click on the **getInstructor** button and you will notice it now returns the inputted value!

The screenshot shows the Remix IDE interface. On the left, the 'DEPLOY & RUN TRANSACTIONS' panel is active. It shows the account '0x083...ad8c4 (99.999999%)', a gas limit of 3000000, and a value of 0 wei. The contract 'Coursestro - contracts/Coursestro.sol' is selected. Below this, there are buttons for 'Deploy', 'Publish to RPS', and 'At Address'. The 'Transactions recorded' section shows the deployment of 'COURSESTRO AT 0XD8B...33FA8 (ME)'. The 'Deployed Contracts' section shows the contract 'COURSESTRO AT 0XD8B...33FA8 (ME)' with buttons for 'setInstructor' and 'getInstructor'. The 'Low level interactions' section shows the 'CALLDATA' field and a 'Transact' button.

```
1 // This line defines the version of solidity you're going to use
2 pragma solidity ^0.4.18;
3 contract Coursestro {
4     string _name;
5     uint _age;
6
7     function Coursestro() public {
8         _name = 'Gary';
9         _age = 34;
10    }
11
12    // setInstructor accepts 2 parameters, _fname and _age.
13    // Once called, we set the string _fname to the returned
14    // _fname, and save with _age.
15    function setInstructor(string _fname, uint _age) public {
16        _name = _fname;
17        _age = _age;
18    }
19
20    // The getInstructor() function is defined as being
21    // constant, and it returns a string and a uint. This is where
22    // we return the _fname and _age variables once it's called.
23    function getInstructor() public constant returns (string, uint) {
24        return (_name, _age);
25    }
26 }
27
```

creation of Coursestro pending...

From	To	Value	Data	Gas	Logs	Hash
0x083...ad8c4	Coursestro (contract)	0 wei	0x00...	60000	0	0x00...

status: true Transaction mined and execution succeed

transaction hash	from	to	gas	transaction cost	execution cost
0x00...a5b4...	0x083...ad8c4	Coursestro (contract)	60000000 gas	80000000 gas	8000 gas

The screenshot shows the 'COURSESTRO AT 0XD8B...33FA8 (ME)' contract interface. The 'setInstructor' button is highlighted in orange, and the input field shows 'Gary', 34. Below this, the 'getInstructor' button is highlighted in blue. The 'Low level interactions' section shows the 'CALLDATA' field and a 'Transact' button.

setInstructor 'Gary', 34

getInstructor

0: string: 'Gary'

1: uint256: 34

Low level interactions

CALLDATA

Transact

When enter new values, need to click **setInstructor** first, then get input value

The screenshot shows the 'COURSESTRO AT 0XD8B...33FA8 (ME)' contract interface. The 'setInstructor' button is highlighted in orange, and the input field shows 'Garryyy', 344. Below this, the 'getInstructor' button is highlighted in blue. The 'Low level interactions' section shows the 'CALLDATA' field and a 'Transact' button.

setInstructor 'Garryyy', 344

getInstructor

0: string: 'Garryyy'

1: uint256: 344

Low level interactions

CALLDATA

Transact

▼

COURSETRO AT 0XD8B...33FA8 (ME)

×

setInstructor

^

_fName:

fagg

_age:

8888

transact

getInstructor

0: string: fagg

1: uint256: 8888