

WETH 代币包装

什么是 WETH ?

WETH (Wrapped ETH)是 ETH 的带包装版本。我们常见的 WETH , WBTC , WBNB , 都是带包装的原生代币。那么我们为什么要包装它们?

在2015年, ERC20标准出现, 该代币标准旨在为以太坊上的代币制定一套标准化的规则, 从而简化了新代币的发布, 并使区块链上的所有代币相互可比。不幸的是, 以太坊本身并不符合 ERC20 标准。

WETH 的开发是为了提高区块链之间的互操作性, 并使 ETH 可用于去中心化应用程序 (dApps)。它就像是给原生代币穿了一件智能合约做的衣服: 穿上衣服的时候, 就变成了 WETH, 符合 ERC20 同质化代币标准, 可以跨链, 可以用于 dApp; 脱下衣服, 它可1:1兑换 ETH。

WETH 合约

目前在用的主网WETH合约写于2015年, 非常老, 那时候solidity是0.4版本。我们用0.8版本重新写一个 WETH。

WETH 符合 ERC20 标准, 它比普通的 ERC20 多了两个功能:

1. 存款: 包装, 用户将 ETH 存入 WETH 合约, 并获得等量的 WETH。
2. 取款: 拆包装, 用户销毁 WETH, 并获得等量的 ETH。

```
1 // SPDX-License-Identifier: MIT
2 pragma solidity ^0.8.0;
3
4 import "@openzeppelin/contracts/token/ERC20/ERC20.sol";
5
6 contract WETH is ERC20{
7     // 事件: 存款和取款
8     event Deposit(address indexed dst, uint wad);
9     event Withdrawal(address indexed src, uint wad);
10
11     // 构造函数, 初始化ERC20的名字和代号
12     constructor() ERC20("WETH", "WETH"){
13     }
14
15     // 回调函数, 当用户往WETH合约转ETH时, 会触发deposit()函数
16     fallback() external payable {
17         deposit();
18     }
```

```
19 // 回调函数，当用户往WETH合约转ETH时，会触发deposit()函数
20 receive() external payable {
21     deposit();
22 }
23
24 // 存款函数，当用户存入ETH时，给他铸造等量的WETH
25 function deposit() public payable {
26     _mint(msg.sender, msg.value);
27     emit Deposit(msg.sender, msg.value);
28 }
29
30 // 提款函数，用户销毁WETH，取回等量的ETH
31 function withdraw(uint amount) public {
32     require(balanceOf(msg.sender) >= amount);
33     _burn(msg.sender, amount);
34     payable(msg.sender).transfer(amount);
35     emit Withdrawal(msg.sender, amount);
36 }
37 }
```

继承

WETH 符合 ERC20 代币标准，因此 WETH 合约继承了 ERC20 合约。

事件

WETH 合约共有 2 个事件：

1. Deposit：存款事件，在存款的时候释放。
2. Withdraw：取款事件，在取款的时候释放。

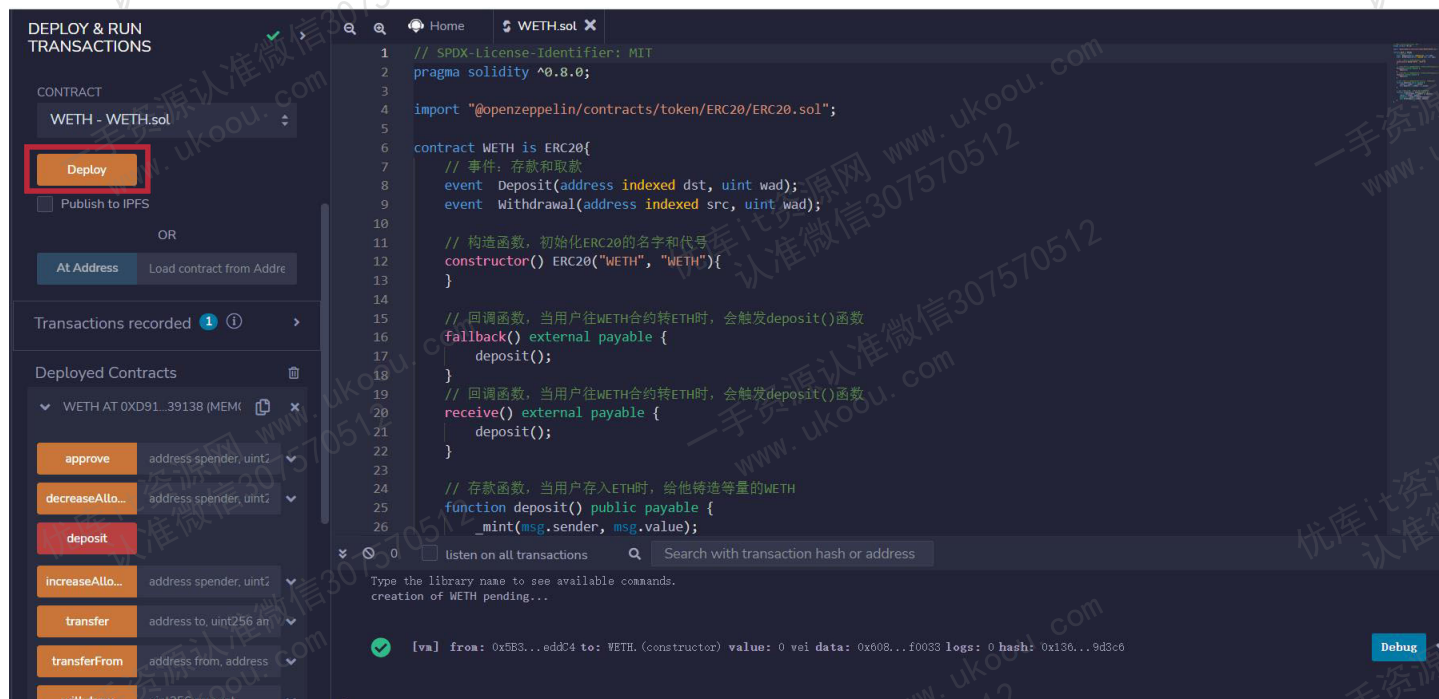
函数

除了 ERC20 标准的函数外，WETH 合约有 5 个函数：

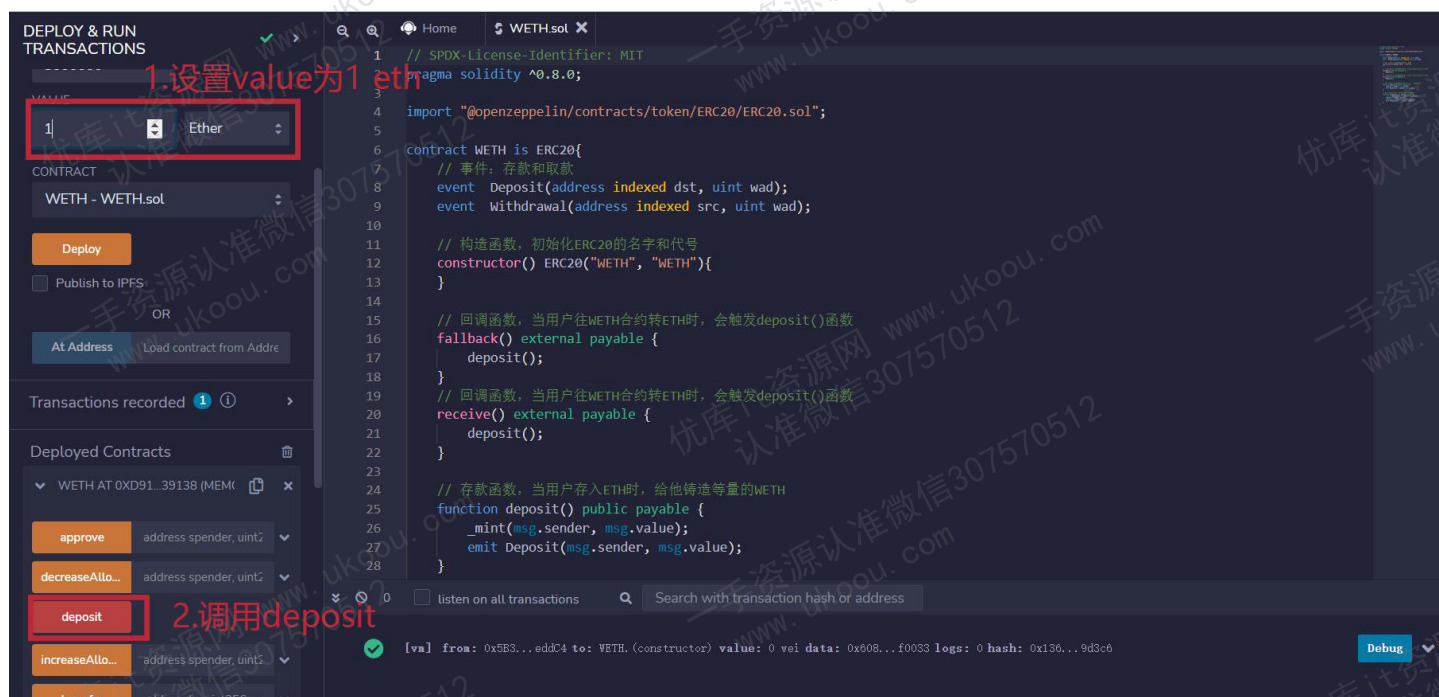
- 构造函数：初始化 WETH 的名字和代号。
- 回调函数：fallback() 和 receive()，当用户往 WETH 合约转 ETH 的时候，会自动触发 deposit() 存款函数，获得等量的 WETH。
- deposit()：存款函数，当用户存入 ETH 时，给他铸造等量的 WETH。
- withdraw()：取款函数，让用户销毁 WETH，并归还等量的 ETH。

Remix 演示

3. 部署 WETH 合约



4. 调用 deposit, 存入 1 ETH, 并查看 WETH 余额



此时 WETH 余额为 1 WETH。

The screenshot shows the WETH.sol interface. On the left, the 'balanceOf' function is selected, showing a balance of 0x5B38Da6a701c568. A red box highlights this balance, with a red arrow pointing to the text '余额为1WETH'. The main area displays the Solidity code for the WETH contract, with the 'Transfer' and 'Deposit' events highlighted in red. The transaction logs at the bottom show the execution of these events.

发出了Transfer和Deposit事件

5. 直接向 WETH 合约转入 1 ETH，并查看 WETH 余额

The screenshot shows the WETH.sol interface. On the left, the 'deposit' function is selected, and the 'value' field is set to 1. A red box highlights this value, with a red arrow pointing to the text '1.设置value为1eth'. The main area displays the Solidity code for the WETH contract, with the 'deposit' function highlighted in red. The transaction logs at the bottom show the execution of the deposit function, with a red box highlighting the 'value' field.

1.设置value为1eth

2.点击低级调用按钮

此时 WETH 余额为 2 WETH。

The screenshot shows the WETH.sol contract interface. On the left, the 'balanceOf' function is highlighted with a red box, showing a balance of 0x5B38Da6a701c568. Below this, a red box contains the text '余额为2WETH'. In the center, a red box highlights the 'deposit' function in the Solidity code, with a red arrow pointing to it from the text 'receive函数调用了deposit函数, 并发出Transfer和Deposit事件'. The right side of the interface shows the transaction logs, with a red box highlighting the 'Deposit' event.

receive函数调用了deposit函数, 并发出Transfer和Deposit事件

余额为2WETH

6. 调用 `withdraw`，取出 1.5 ETH，并查看 WETH 余额

The screenshot shows the WETH.sol contract interface. On the left, the 'withdraw' function is highlighted with a red box, showing a value of 1500000000000000000. Below this, a red box contains the text '输入1.5eth, 调用withdraw函数'. The right side of the interface shows the transaction logs, with a red box highlighting the 'CALL' transaction.

输入1.5eth, 调用withdraw函数

此时 WETH 余额为 0.5 WETH。

DEPLOY & RUN TRANSACTIONS

WETH AT 0xD91...39138 (MEM)

approve address spender, uint2

decreaseAllo... address spender, uint2

deposit

increaseAllo... address spender, uint2

transfer address to, uint256 an

transferFrom address from, address

withdraw 15000000000000000000

allowance address owner, address

balanceOf 0x5B38Da6a701c568

0: uint256: 5000000000000000000

decimals

name

symbol

totalSupply

Low level interactions

```
1 // SPDX-License-Identifier: MIT
2 pragma solidity ^0.8.0;
3
4 import "@openzeppelin/contracts/token/ERC20/ERC20.sol";
5
6 contract WETH is ERC20 {
7     ContractDefinition WETH 1 reference(s) dst, uint wad);
8
9     "uint256 amount": "15000000000000000000"
10
11     decoded output
12
13     logs
14
15     "from": "0xd9145CE52D386f254917e481eB44e9943F39138",
16     "topic": "0xddf25b9da1e2c8969c2b008f378daa952ba7f103c4a11028f55a4df523b3ef",
17     "event": "Transfer",
18     "args": {
19         "0": "0x5B38Da6a701c568545dCfC803FcB875f56beddC4",
20         "1": "0x0000000000000000000000000000000000000000000000000000000000000000",
21         "2": "150000000000000000000000000000000000000000000000000000000000000000",
22         "3": "0x5B38Da6a701c568545dCfC803FcB875f56beddC4",
23         "4": "0x0000000000000000000000000000000000000000000000000000000000000000",
24         "value": "150000000000000000000000000000000000000000000000000000000000000000"
25     }
26 }
27
28 "from": "0xd9145CE52D386f254917e481eB44e9943F39138",
29 "topic": "0x7f555257680a5d5e58bba7133d888bc7d98cb3f7268a95bf5081b05",
30 "event": "Withdrawal",
31 "args": {
32     "0": "0x5B38Da6a701c568545dCfC803FcB875f56beddC4",
33     "1": "150000000000000000000000000000000000000000000000000000000000000000",
34     "src": "0x5B38Da6a701c568545dCfC803FcB875f56beddC4",
35     "wad": "150000000000000000000000000000000000000000000000000000000000000000"
36 }
37
38 }
39
40 val
41 0 val
```

发出了Transfer和Withdraw事件

余额为0.5WETH

总结

这一讲，我们介绍了 WETH 并实现了 WETH 合约。它就像是给原生 ETH 穿了一件智能合约做的衣服：穿上衣服的时候，就变成了 WETH，符合 ERC20 同质化代币标准，可以跨链，可以用于 dApp；脱下衣服，它可以1:1兑换 ETH。