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，以太币本身并不符合 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 。

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";

contract WETH is ERC20{
    // 事件：存款和取款
    event  Deposit(address indexed dst, uint wad);
    event  Withdrawal(address indexed src, uint wad);

    // 构造函数，初始化ERC20的名字和代号
    constructor() ERC20("WETH", "WETH"){
    }

    // 回调函数，当用户往WETH合约转ETH时，会触发deposit()函数
    fallback() external payable {
        deposit();
    }
```

```
    // 回调函数，当用户往WETH合约转ETH时，会触发deposit()函数
    receive() external payable {
        deposit();
    }

    // 存款函数，当用户存入ETH时，给他铸造等量的WETH
    function deposit() public payable {
        _mint(msg.sender, msg.value);
        emit Deposit(msg.sender, msg.value);
    }

    // 提款函数，用户销毁WETH，取回等量的ETH
    function withdraw(uint amount) public {
        require(balanceOf(msg.sender) >= amount);
        _burn(msg.sender, amount);
        payable(msg.sender).transfer(amount);
        emit Withdrawal(msg.sender, amount);
    }
}
```

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。

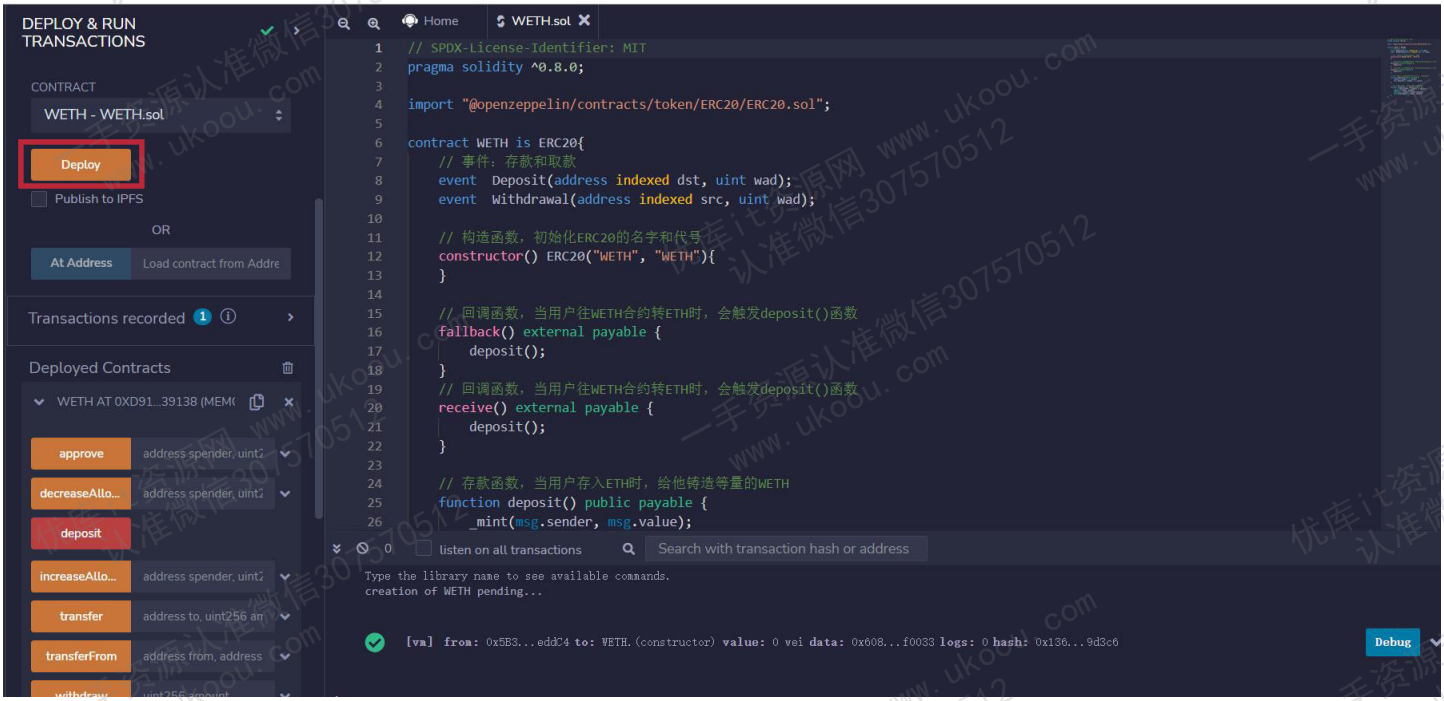
函数

除了 `ERC20` 标准的函数外， `WETH` 合约有 `5` 个函数：

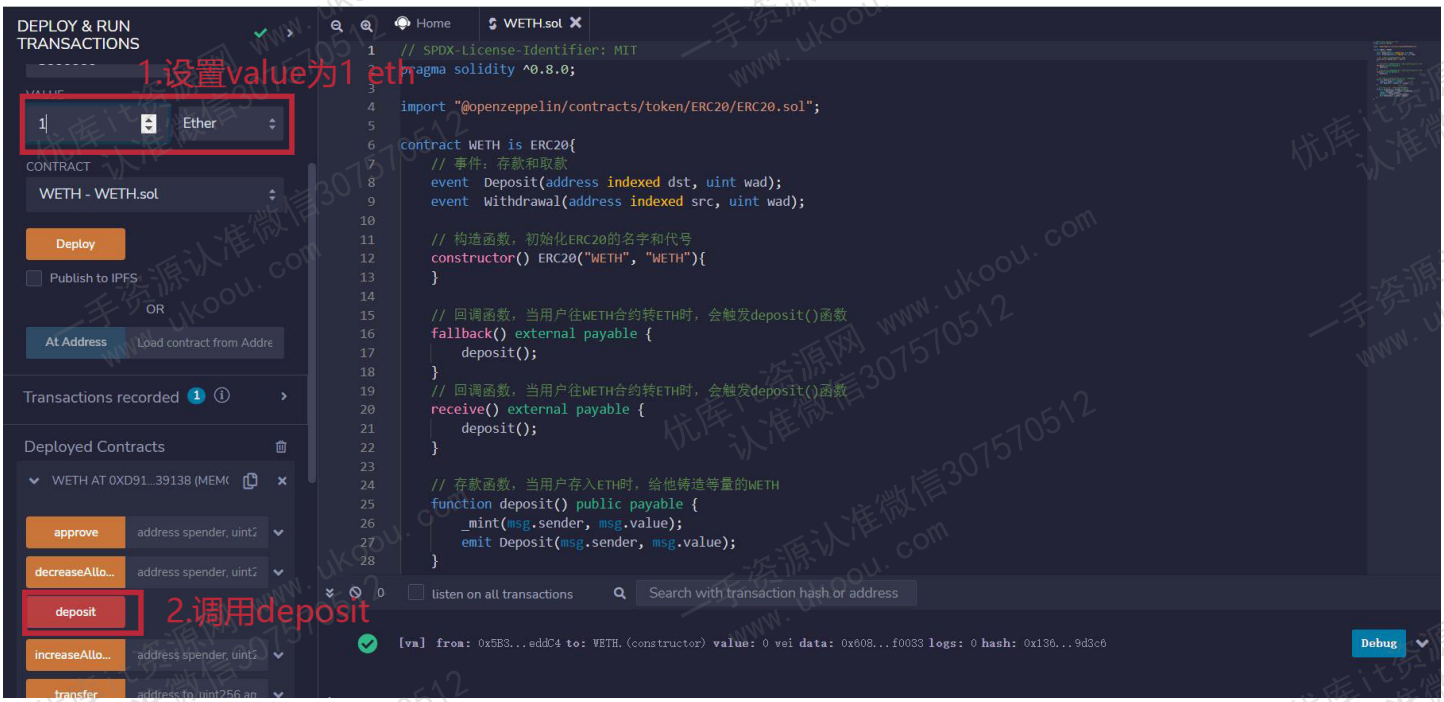
- 构造函数：初始化 `WETH` 的名字和代号。
- 回调函数： `fallback()` 和 `receive()` ，当用户往 `WETH` 合约转 `ETH` 的时候，会自动触发 `deposit()` 存款函数，获得等量的 `WETH` 。
- `deposit()` ：存款函数，当用户存入 `ETH` 时，给他铸造等量的 `WETH` 。
- `withdraw()` ：取款函数，让用户销毁 `WETH` ，并归还等量的 `ETH` 。

`Remix` 演示

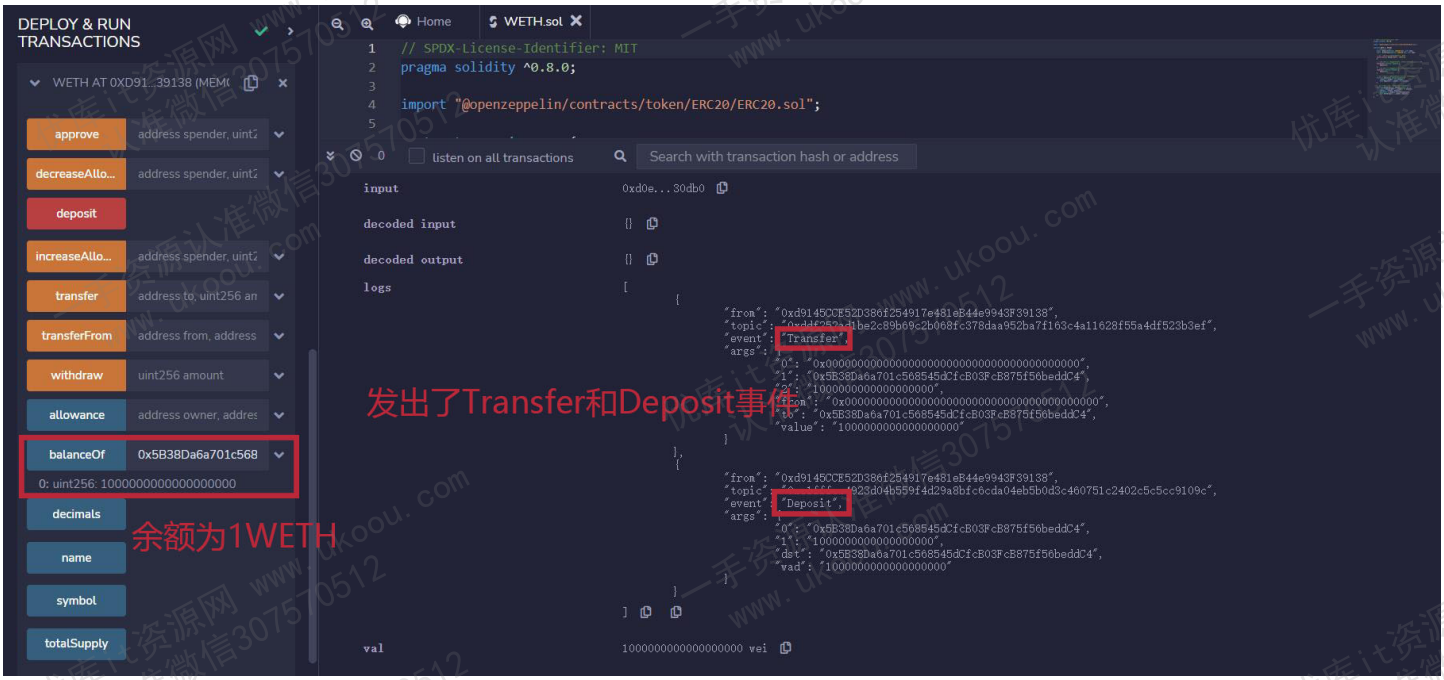
3. 部署 `WETH` 合约



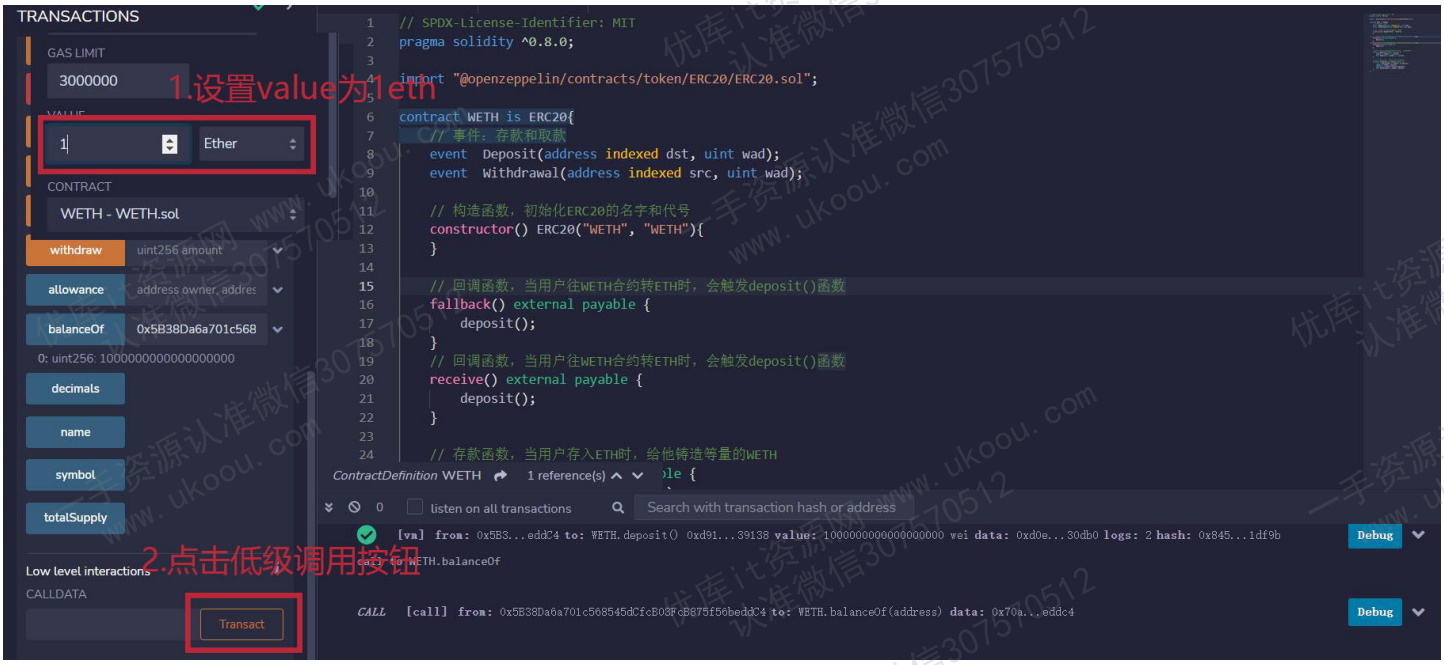
4. 调用 `deposit`，存入 `1 ETH`，并查看 `WETH` 余额



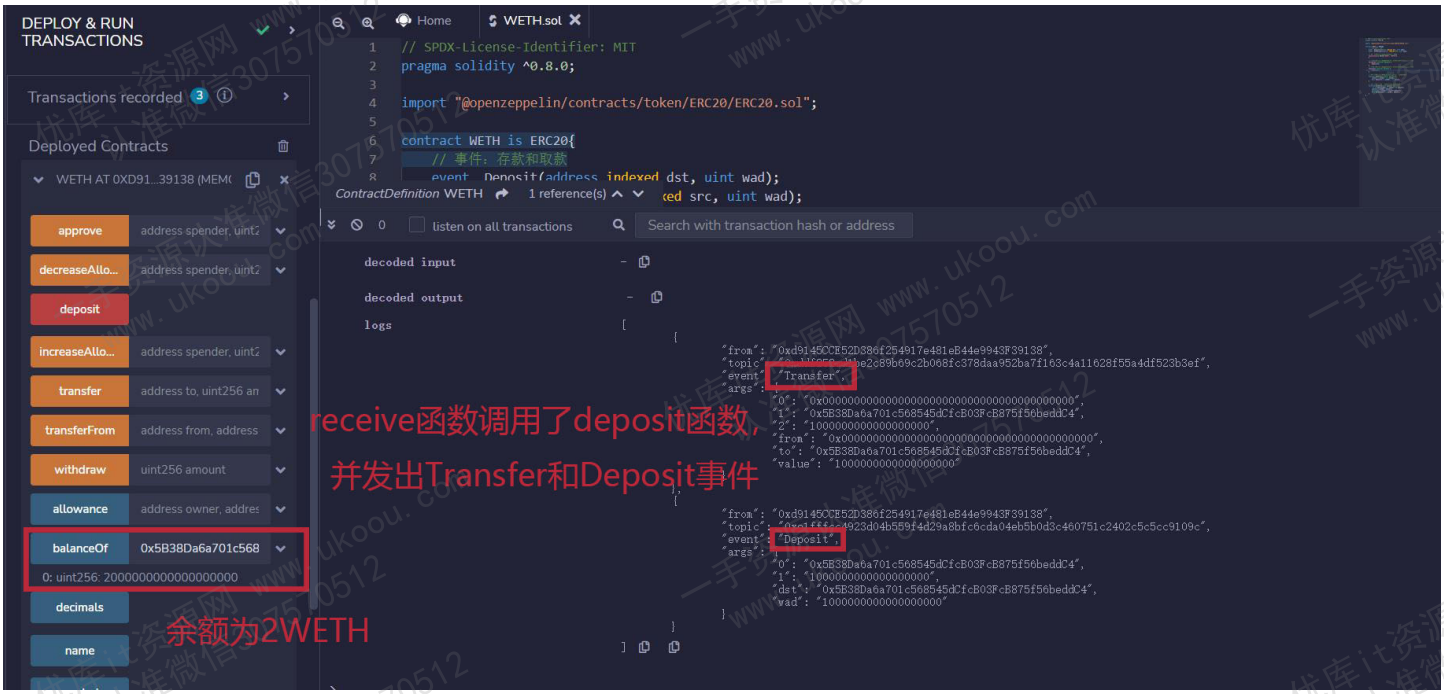
此时 `WETH` 余额为 `1 WETH`。



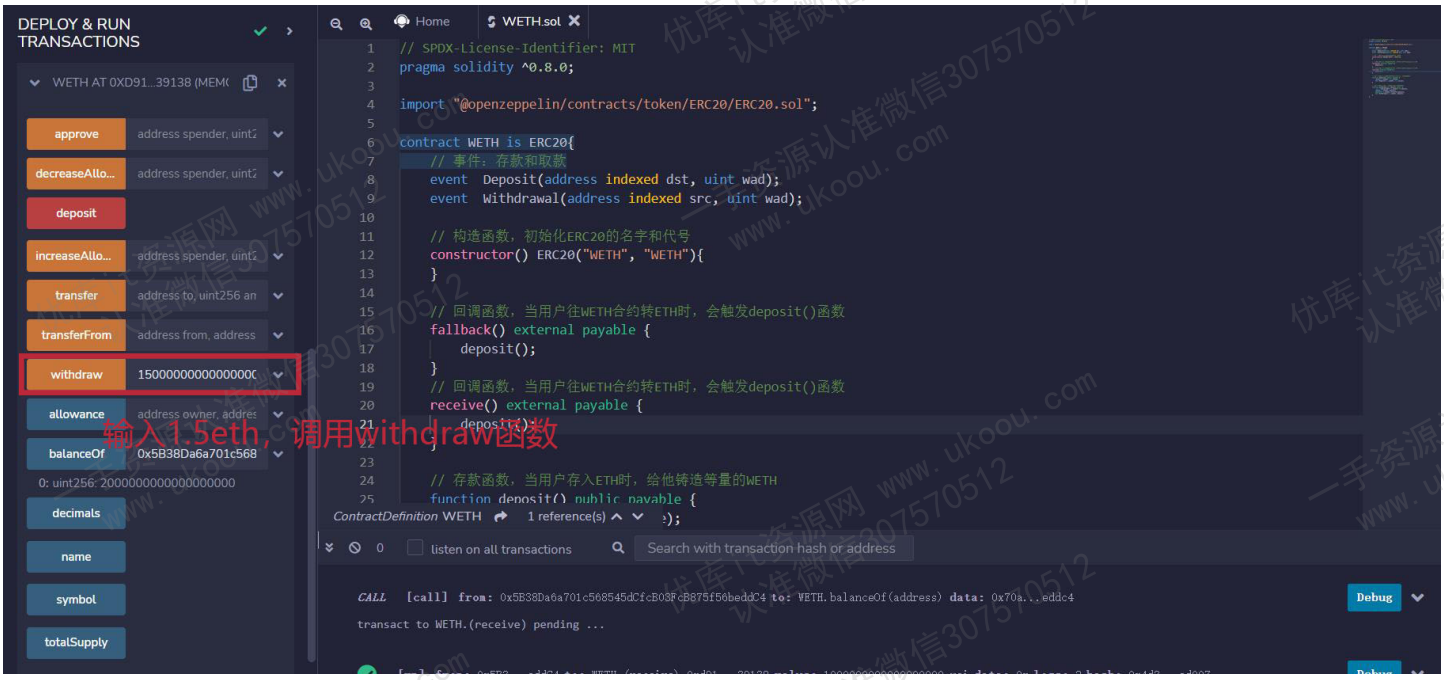
5. 直接向 `WETH` 合约转入 `1 ETH`，并查看 `WETH` 余额



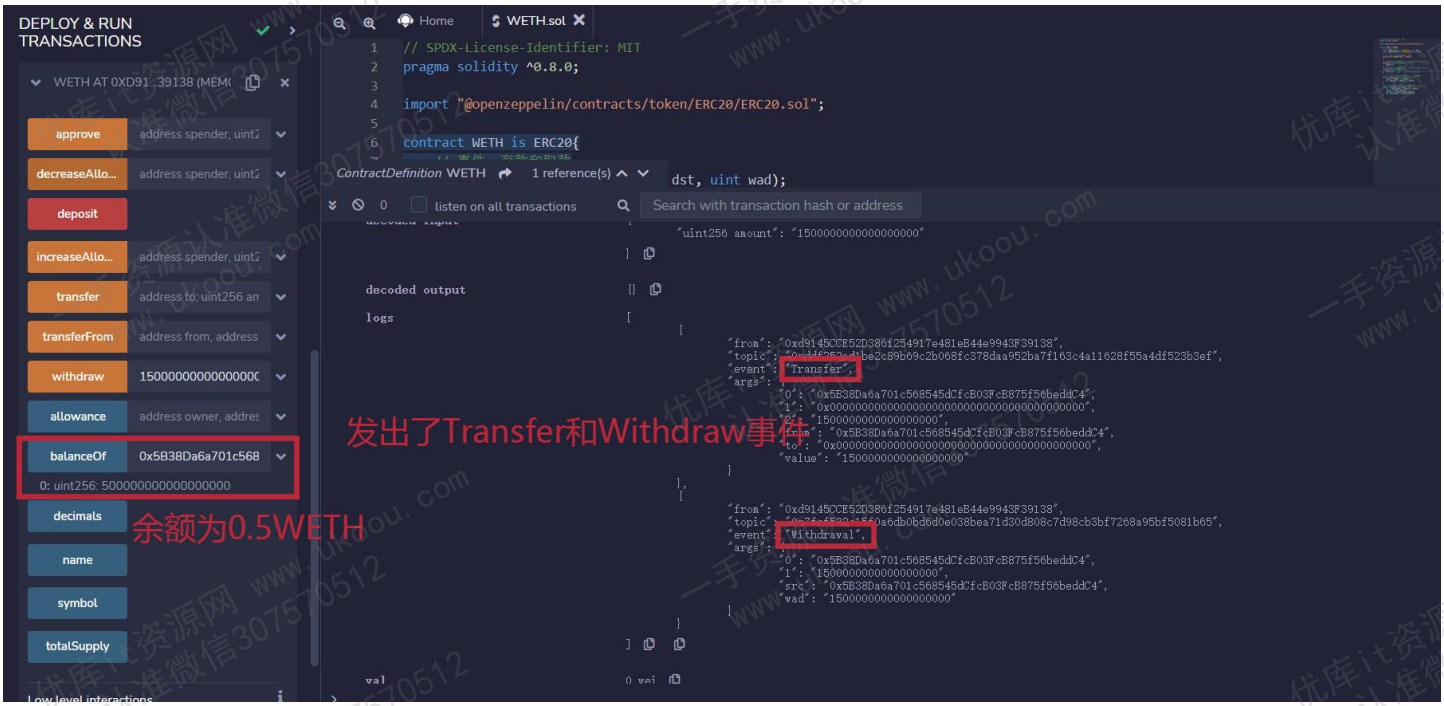
此时 `WETH` 余额为 `2 WETH`。



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



此时 `WETH` 余额为 `0.5 WETH`。



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`。