



# Solidity 智能合约的 开发流程

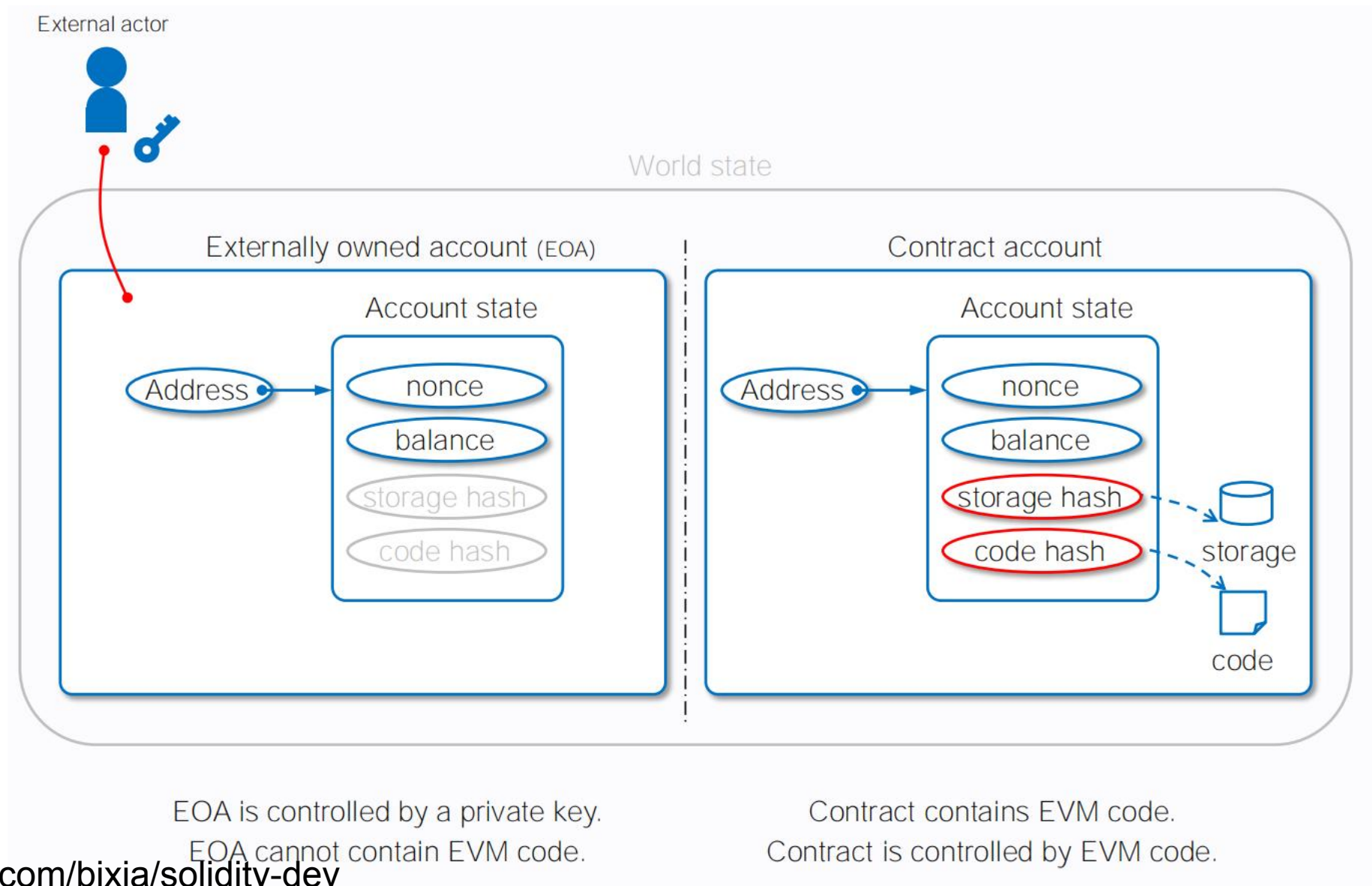
开发，测试与部署

bugWriter 20220326

Ref: <https://github.com/bixia/solidity-dev>

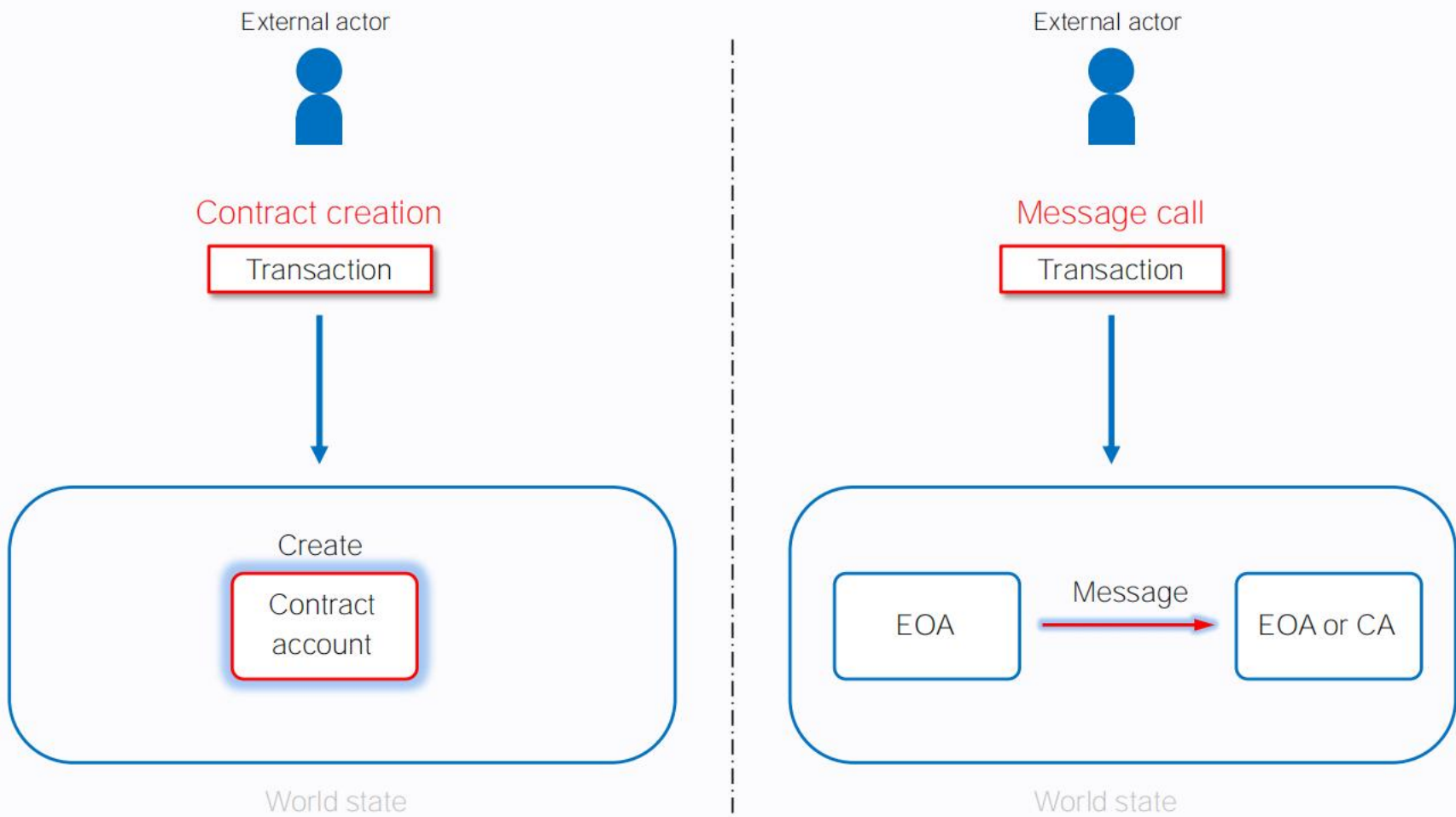


# 以太坊的账户模型





# 以太坊的交易



There are two practical types of transaction, contract creation and message call.

Ref: <https://github.com/bixia/solidity-dev>





# 认识Solidity智能合约

- 在EVM兼容的链上执行
- 先编译后执行
- Solidity与opcode关系





# 智能合约组成部分

- SPDX–License
- pragma solidity 0.8.0;
- contract 关键字
- constructor 关键字
- public,private,external,internal
- 全局变量
- 只读函数view
- 状态更改函数

官方文档链接:

<https://docs.soliditylang.org/en/v0.8.13/>

中文文档链接:

<https://learnblockchain.cn/docs/solidity/>

//SPDX-License-Identifier: MIT

pragma solidity 0.8.12;

UnitTest stub | dependencies | uml | draw.io

contract Dev {

address public owner;

uint256 private key;

ftrace

constructor() payable {

owner = address(0xdeadbeef);

key = 32;

}

modifier onlyOwner() {

require(msg.sender == owner, "!owner");

=;

}

ftrace | funcSig

function setKey(uint256 \_key) public onlyOwner {

key = \_key;

}

ftrace | funcSig

function getOwner() public view returns (address) {

return owner;

}

ftrace | funcSig

function hack(uint256 \_key) public {

require(key == \_key, "Wrong key");

payable(address(msg.sender)).transfer(address(this).balance);

}

ftrace

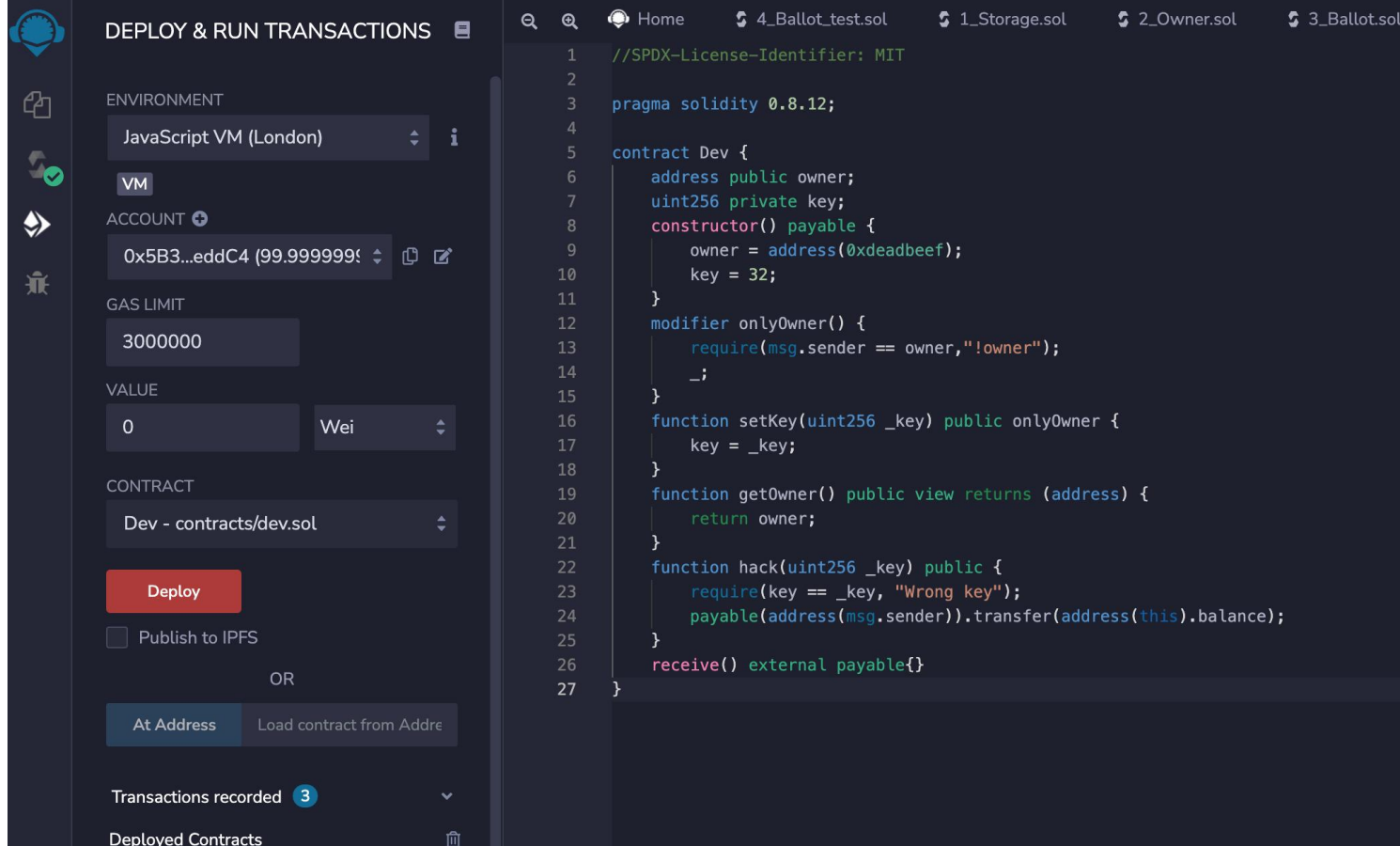
receive() external payable{

}



# Remix

- 网页版IDE，无需环境配置
- 一键编译
- Debug模式
- 部署到本地网络
- 部署到测试网络



官方链接: <https://remix.ethereum.org/>







# 测试网络

- 获取测试币
- 开源源码
- 使用etherscan浏览器调用
  - 调用view方法
  - 调用非view方法

rinkeby.etherscan.io

Etherscan

Home Blockchain Tokens Misc Rinkeby

Rinkeby Testnet Explorer

All Filters Search by Address / Txn Hash / Block / Token / Ens

Advertise your brand here! Start Today

Latest Blocks			Latest Transactions		
Bk	10391791 16 secs ago	Miner 0xf10326c1c6884b094e... 23 txns in 15 secs 0.00866 Eth	Tx	0x8170ed1ea1ea... 16 secs ago	From 0xc6903bff3452a7adbffc... To 0xa9917d31d30048dcf2... 0 Eth
Bk	10391790 31 secs ago	Miner 0xd6ae8250b8348c9484... 21 txns in 15 secs 0.01056 Eth	Tx	0xa6fa0af2e4955... 16 secs ago	From 0xc42ec859e58d58ae02... To 0x1174cd75c386a538cf4... 0 Eth
Bk	10391789 46 secs ago	Miner 0x7ffc57839b00206d1ad... 34 txns in 15 secs 0.01013 Eth	Tx	0x608fab47b83d... 16 secs ago	From 0xc42ec859e58d58ae02... To 0x1174cd75c386a538cf4... 0 Eth
Bk	10391788 1 min ago	Miner 0x6dc0c0be4c8b2dfe750... 26 txns in 15 secs 0.01404 Eth	Tx	0xc46ac7993f4f8... 16 secs ago	From 0x338fe4dee63031e61a... To 0x7d58560196dc564d63... 0 Eth
Bk	10391787 1 min ago	Miner 0x42eb768f2244c8811c6... 17 txns in 15 secs 0.01077 Eth	Tx	0x5c60f0299e08f... 16 secs ago	From 0x338fe4dee63031e61a... To 0x7d58560196dc564d63... 0 Eth
Bk	10391786 1 min ago	Miner 0x6635f83421bf059cd81... 22 txns in 15 secs 0.0023 Eth	Tx	0xd9bf255f9efb3... 16 secs ago	From 0x7cb4f0d468806aba9c... To 0x25726251659d9455... 0 Eth

View all blocks View all transactions

官方链接：

获取测试币：<https://faucet.rinkeby.io/>

测试网浏览器：<https://rinkeby.etherscan.io/>





# Foundry

官方链接:

foundry的官方仓库: <https://github.com/gakonst/foundry>

foundry的官方文档: <https://onbjerg.github.io/foundry-book/config/vscode.html>

- 现代化Solidity开发框架

- 下载

```
curl -L https://foundry.paradigm.xyz | bash
```

- 安装

```
foundryup
```

- 编译

```
$ forge init hello_foundry
```

- 测试

```
$ forge build  
compiling...  
success.
```

```
$ forge test  
compiling...  
no files changed, compilation skipped.  
Running 1 test for ContractTest.json:ContractTest  
[PASS] testExample() (gas: 254)
```







# Foundry fork主网

- RPC节点
- 区块高度
- 在foundry中重放一笔交易

```
//SPDX-License-Identifier: MIT  
pragma solidity 0.8.12;
```

```
import "ds-test/test.sol";  
import "forge-std/Vm.sol";
```

```
//forge test --match-contract ForkTest --fork-url https://eth-mainnet.alchemyapi.io/v2/7Brn0mxZn  
/// https://etherscan.io/tx/0x5cfed0b8664dfbc7bc74d83e4d425ce47ebf13e964c50b51e928f4a88c0b7577
```

UnitTest stub | dependencies | uml | draw.io

```
contract ForkTest is DSTest {  
    Vm public vm = Vm(HEVM_ADDRESS);  
    address EOA = 0x108F0eC0D7F05490b7Fa284de5F1D4b1dfb64Dbf;  
    address uniswapV3Router = 0x68b3465833fb72A70ecDF485E0e4C7bD8665Fc45;  
    address weth = 0xC02aaA39b223FE8D0A0e5C4F27eAD9083C756Cc2;  
    address token = 0xF1cA9cb74685755965c7458528A36934Df52A3EF;  
    address pool = 0x27a9ff745cf1Dd366d94267Cb4aDE2350588a187;
```

ftrace | funcSig

```
function setUp() public {  
    vm.label(EOA, "EOA");  
    vm.label(uniswapV3Router, "uniswapV3Router");  
    vm.label(weth, "weth");  
    vm.label(token, "token");  
    vm.label(pool, "pool");  
}
```

ftrace | funcSig

```
function test_Reply() public {  
    vm.startPrank(EOA);  
    address(uniswapV3Router).call(hex"5ae401dc00000000000000000000000000000000000000000000000000000000");  
    vm.stopPrank();  
}
```