Foundry Workshop



2022/11/17

Intro

- wiasliaw
- FiO Blockchain Dev
- <u>Medium</u>
- <u>LunDao</u>



Why Foundry

- a portable, fast and modular toolkit
- rust reimplementation of dapptools

Why Foundry (cont.)

- Tests in Solidity, not Javascript
 - lots of dependencies, configs and node_modules
 - missing typing
- Big Number libs
- Testing with level-1 abstraction
- Fuzz Test

Why Foundry (cont.)

What's the vision?

In Summer 2020, we started with writing ethers-rs, a Rust port of ethers.js, with the goal of helping MEV traders build better bots.

Installation

- CLI: forge, cast and anvil
- windows: install Rust and build from source
- Linux/MacOS
 - build from source
 - use foundryup

foundryup

Need curl & git

```
$ sudo apt -y install curl git
$ curl -L https://foundry.paradigm.xyz | bash
```

\$ foundryup

forge

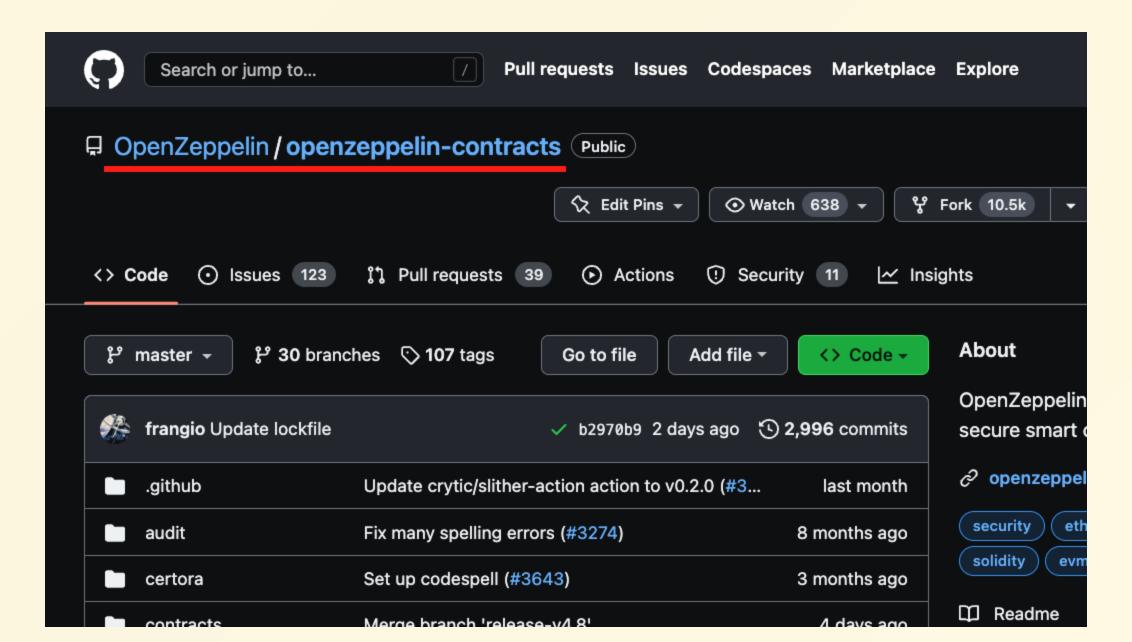
- compile, test, deploy contracts
- dependency management

forge init

• init a repo with <u>default template</u>

```
$ forge init hello
$ npx tree-cli -1 2
/home/parallels/Desktop/hello
    foundry.toml
   lib
   └─ forge-std
   - script
   └─ Counter.s.sol
   src
   L— Counter.sol
    test
   — Counter.t.sol
```

forge install



forge install/update/remove

```
$ forge install OpenZeppelin/openzeppelin-contracts --no-commit
$ forge update ./lib/openzeppelin-contracts
$ forge remove openzeppelin-contracts
```

remappings

How Solidity compiler resolve import files.

remapping hardhat

```
import "@openzeppelin/contracts/token/ERC20/ERC20.sol";
-> @openzeppelin -> node_modules/@openzeppelin/...
```

remapping foundry

```
$ forge remappings > remappings.txt
```

remappings.txt

```
ds-test/=lib/forge-std/lib/ds-test/src/
forge-std/=lib/forge-std/src/
openzeppelin-contracts/=lib/openzeppelin-contracts/contracts/
```

import

```
import "openzeppelin-contracts/token/ERC20/ERC20.sol";
```

forge build & test

```
$ forge build
$ forge test
```

Testing

- cheatcode
 - edit evm status for testing
 - assertion
- need to test
 - getter/setter
 - event/revert

Cheatcode

- warp, roll, deal
- startPrank, stopPrank
- assertEq , expectEmit , expectRevert
- assume
- startBroadcast, stopBroadcast

getter/setter - sample

contract

```
contract Counter {
    uint256 public number;
    function setNumber(uint256 newNumber) public {
        number = newNumber;
    function increment() public {
        number++;
```

```
// SPDX-License-Identifier: UNLICENSED
pragma solidity ^0.8.13;
import "forge-std/Test.sol";
import "../src/Counter.sol";
contract CounterTest is Test {
    Counter public counter;
    function setUp() public { // beforeEach
        counter = new Counter();
        counter.setNumber(∅);
    function testIncrement() public { // prefix testXXXX
        counter.increment();
        assertEq(counter.number(), 1);
```

event - sample

```
import "forge-std/Test.sol";
contract TestERC20 is Test {
    MintableERC20 private _erc20;
    event <u>Transfer</u>(address indexed <u>from</u>, address indexed to, uint256 value);
    function setUp() public {
        _erc20 = new MintableERC20();
    function testMint() public {
        vm.expectEmit(true, true, true, true);
        emit Transfer(address(0), address(this), 1000);
        _erc20.mint(1000);
```

revert - sample

- other 's allowance == 0
- reason string is ERC20: insufficient allowance

```
function testShouldRevert() public {
    _erc20.mint(1000);
    address other = vm.addr(1);
    vm.startPrank(other);
    vm.expectRevert("ERC20: insufficient allowance");
    _erc20.transferFrom(address(this), other, 1000);
    vm.stopPrank();
}
```

Fuzz Test

- Property-based Test
 - 一個測試代表著對該測試對象的證明。因此,一個 property 可視 為該測試對象的「標準」(invariants)或是「規格」 (specification)
 - 測資不應該是手刻,要以自動產生的方式來尋找 edge case

舉例:一個 transfer erc20 的 function,測試的 property 就是「給定一個安全的值,從某個帳號轉帳給另一個帳號」

Fuzz Test - example

sample

```
// SPDX-License-Identifier: UNLICENSED
pragma solidity ^0.8.13;
import "openzeppelin-contracts/token/ERC20/utils/SafeERC20.sol";
contract <u>SafePayment</u> {
    address private _erc20;
    constructor(address erc20) {
        _{erc20} = erc20;
    function payment(address from, address to, uint256 value) external {
        SafeERC20.safeTransferFrom(IERC20(_erc20), from, to, value);
```

```
contract <u>TestPayment</u> is <u>Test</u> {
    MintableERC20 private _token;
    SafePayment private _payment;
    function setUp() public {
        _token = new <u>MintableERC20()</u>;
        _payment = new <u>SafePayment(address(_token));</u>
    function testFuzzPayment(
    ) public {
        vm.startPrank(from);
        _token.mint(amount);
        _token.approve(to, amount);
        vm.stopPrank();
        vm.startPrank(to);
        _payment.payment(from, to, amount);
        vm.stopPrank();
```

fix bugs: clear invalid input

```
function testFuzzPayment(
   address from, address to, uint256 amount
) public {
    vm.assume(from != address(∅));
    vm.assume(to != address(∅));
    vm.startPrank(from);
    _token.mint(amount);
    _token.approve(address(_payment), amount);
    vm.stopPrank();
    vm.startPrank(to);
    _payment.payment(from, to, amount);
    vm.stopPrank();
```

Deployment

- deployment
- verification

forge create

```
$ forge create <PATH>:<CONTRACT_NAME>
```

example

```
forge create \
    --rpc-url <url> \
    --private-key <key> \
    src/Simple.sol:Simple \
    --constructor-args "My Token" "MT" \
    --etherscan-api-key <key> \
    --verify
```

forge verify-contract

```
forge verify-concompiler-version "v0.8.11+commit.d7f03943" \
     <Address> <Path>:<Contract>
```

Script

Using Solidity write script.

example

```
// SPDX-License-Identifier: UNLICENSED
pragma solidity ^0.8.13;
import "forge-std/Script.sol";
import "openzeppelin-contracts/token/ERC721/ERC721.sol";
contract ERC721Deployment is Script {
    // default sig
    // load .env by default
    function run() external {
        uint256 deployerPrivateKey = vm.envUint("PRIVATE_KEY");
        vm.startBroadcast(deployerPrivateKey);
        ERC721 nft = new ERC721("Hello", "World");
        vm.stopBroadcast();
```

```
$ forge script ./script/ERC721Deployment.script.sol \
[--broadcast] \
[--verify]
```

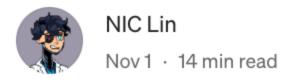
misc

- cast : cli utils like ethers-js
- anvil: local node development

Recommendation



Published in Taipei Ethereum Meetup





Solidity 及 EVM 開發工具介紹

這篇文章將介紹 Ethereum 開發者的一些實用工具: Foundry 除了測試之外的功能及 VSCode 的 Solidity Visual Developer 插件

Reference

- Introducing the Foundry Ethereum development toolbox
- foundry docs
- dapptools github
- cybai introduction-to-property-based-testing-at-coscup-2022