



ScaleBit

OpenBuild

Tact 合约测试和交互

Jason



大纲

- 合约测试
 - 测试框架介绍
 - 编写测试代码
- 合约交互
 - Typescript SDK和FT交互调用
 - Typescript SDK和和NFT交互调用
- 总结
- 作业

测试框架介绍

- TON有一个sandbox的区块链模拟执行环境
- Tact的TypeScript SDK已经默认集成了sandbox, 提供该环境
 - <https://github.com/tact-lang/tact-template>

tact-template / package.json

anton-trunov Update to @ton/core deps (#9)

Code Blame 32 lines (32 loc) · 853 Bytes

```
1 {
2   "private": true,
3   "scripts": {
4     "build": "tact --config ./tact.config.json",
5     "test": "jest",
6     "deploy": "ts-node ./sources/contract.deploy.ts",
7     "read": "ts-node ./sources/contract.read.ts"
8   },
9   "dependencies": {
10    "@tact-lang/compiler": "^1.1.5",
11    "@tact-lang/coverage": "^0.0.8",
12    "@tact-lang/deployer": "^0.2.0",
13    "@tact-lang/emulator": "^4.2.3",
14    "@ton/core": "0.49.2",
15    "@ton/crypto": "^3.2.0",
16    "@types/jest": "^29.2.4",
17    "@types/node": "^18.11.14",
18    "@types/qs": "^6.9.7",
19    "base64url": "^3.0.1",
20    "enquirer": "^2.3.6",
21    "jest": "^29.3.1",
22    "open": "^8.4.0",
23    "prando": "^6.0.1",
24    "prettier": "^2.5.1",
25    "qs": "^6.11.0",
26    "@tact-lang/ton-abi": "^0.0.3",
27    "@tact-lang/ton-jest": "^0.0.4",
28    "ts-jest": "^29.0.3",
29    "ts-node": "^10.9.1",
30    "typescript": "^4.9.4"
31   }
32 }
```



智能合约测试的一般流程

智能合约测试可以覆盖安全性、优化 Gas 支出以及检查边缘情况。

- 创建测试账户
- 创建合约实例
- 发送特定消息
- 检查消息执行结果
- 检查状态

```
1 import { toNano } from "@ton/core";
2 import { ContractSystem } from "@tact-lang/emulator";
3 import { SampleTactContract } from "../output/sample_SampleTactContract";
4
5 describe("contract", () => {
6     it("should deploy correctly", async () => {
7         // Create ContractSystem and deploy contract
8         let system = await ContractSystem.create();
9         let owner = system.treasure("owner");
10        let nonOwner = system.treasure("non-owner");
11        let contract = system.open(await SampleTactContract.fromInit(owner.address));
12        system.name(contract.address, "main");
13        let track = system.track(contract);
14        await contract.send(owner, { value: toNano(1) }, { $$type: "Deploy", queryId: 0n });
15        await system.run();
16 > expect(track.collect()).toMatchInlineSnapshot(` ...
17 `);
18
19        // Check counter
20        expect(await contract.getCounter()).toEqual(0n);
21
22        // Increment counter
23        await contract.send(owner, { value: toNano(1) }, "increment");
24        await system.run();
25 > expect(track.collect()).toMatchInlineSnapshot(` ...
117 `);
```



执行测试

- Ton Emulator 允许您在 Nodejs 代码中拥有一个小型虚拟区块链, 该库是专门为在单元测试中测试智能合约而构建的。
- 支持jest的静态测试。
 - yarn jest 生成快照或测试
 - yarn jest -u 更新快照



jest生成快照

```
describe("contract", () => {
  it("should deploy correctly", async () => {

    // Init test
    const system = await ContractSystem.create();
    const treasure = await system.treasure('my treasure');
    const contract = system.open(sample_Contract.fromInit(
    const tracker = system.track(contract.address);

    // Send a message
    await contract.send(treasure, { value: toNano(1) }, {
    await system.run();

    // Testing output
    expect(tracker.collect()).toMatchInlineSnapshot();

  });
});
```

```
expect(tracker.collect()).toMatchInlineSnapshot(`
  [
    {
      "$seq": 0,
      "events": [
        {
          "$type": "deploy",
        },
        {
          "$type": "received",
          "message": {
            "body": {
              "cell": "x{946A98B6000000000000000000}",
              "type": "cell",
            },
            "bounce": true,
            "from": "kQAI-3FJVc_ywSuY4vq0bYrzR7S40ch4y7bTU_i5yL0B3A6P",
            "to": "kQBrSAP2y7QIUw4_1q0qciHfqdFm0YR9CC1oinn7kyWWRuov",
            "type": "internal",
            "value": 1000000000n,
          },
        },
      ],
    },
  ],
`);
```



在合约中启用dump

```
{ } tact.config.json > ...
```

You, 35 minutes ago | 2 authors (Steve Korshakov and others)

```
1  {
2    "projects": [{
3      "name": "sample",
4      "path": "./sources/contract.tact",
5      "output": "./sources/output",
6      "options": {
7        "debug": true
8      }
9    }
10  ]
11 }
```

Steve Korshakov, 13 months ago • feat:

```
31 receive("increment") {
32   dump("received increment");
33
34   self.add(1);
35   self.reply("incremented".asComment());
36 }
```




FT调用案例

合约代码如下, 通过Tact Typescript SDK进行交互

<https://github.com/howardpen9/jetton-implementation-in-tact>



NFT调用案例

合约代码如下, 通过Tact Typescript SDK进行交互

<https://github.com/howardpen9/nft-template-in-tact/blob/tutorial/sources/contract.tact>



总结

- 智能合约测试能极大程度保障合约代码的正确性
- Tact Typescript SDK功能强大, 能实现和链的各种交互



作业

做一个TODO合约应用, 用户可以录入每天的任务, 用户将当天的所有任务完成后, 可以获得一个完成任务的勋章NFT。



参考

本章参考代码

<https://github.com/0xOutOfGas/tact-template-jetton>



ScaleBit

Thanks

Contact us:

- Twitter: @scalebit_
- Email: contact@scalebit.xyz

More information : www.scalebit.xyz