SYDNEY ETHEREUM

TRUFFLE & TESTRPC

VOJDAN KARDALEV

GOAL

- Up and running with Ethereum
  - Local test node
  - Development framework

TOPICS

- Installing testrpc
- Deploying contracts to testrpc
- Installing Truffle
- Building solidity code using Truffle
- Writing Truffle tests
- Executing Truffle tests

REQUIREMENTS

- Mac OSX (preferred), Linux or Windows
- nodeJS
- npm

TESTRPC

- Fast Ethereum RPC client for testing and development
- Uses ethereumjs to simulate full client behavior
- Includes all popular RPC functions and features (like events) and can be run deterministically

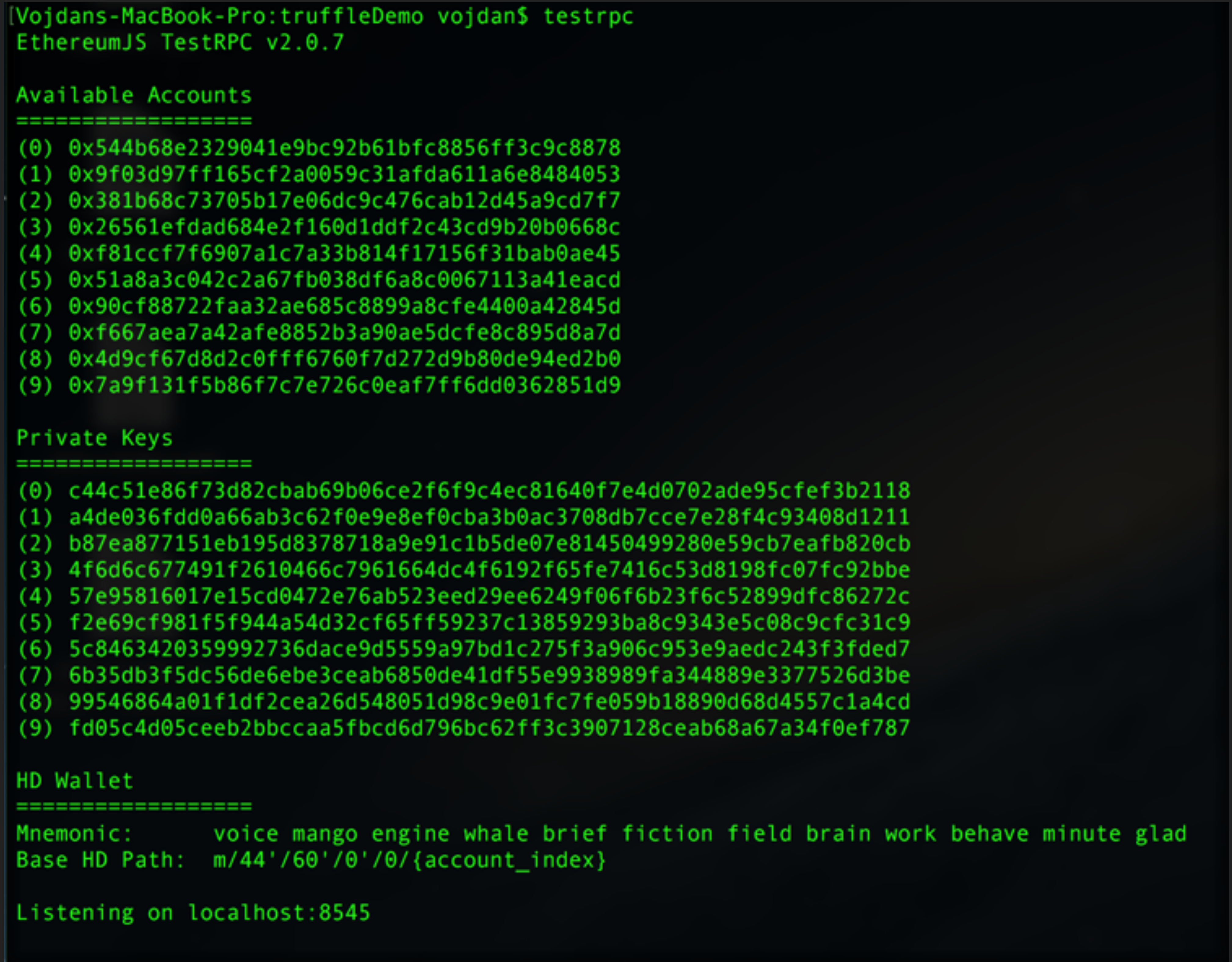
TESTRPC - INSTALLATION

- OSX & Linux
  - $ npm install -g ethereumjs-testrpc
- Windows (not tested)
  - Install Python 2.7 if not already installed, and make sure to add its install location to your PATH.
  - Install OpenSSL from here. Make sure to choose the right package for your architecture, and only install the full package (not the light version). You must install OpenSSL at its recommended location - do not change the install path.
  - $ npm install -g ethereumjs-testrpc

TESTRPC - USAGE

- `$ testrpc <options>`
- -a: Specify the number of accounts to generate at startup.
- -d: Generate deterministic addresses based on a pre-defined mnemonic.
- -m: Use a specific HD wallet mnemonic to generate initial addresses.
- -p: Port number to listen on.
- -s: Use arbitrary data to generate the HD wallet mnemonic to be used.
- --debug: Output VM opcodes for debugging

TESTRPC - USAGE



```
[Vojdans-MacBook-Pro:truffleDemo vojdan$ testrpc
 EthereumJS TestRPC v2.0.7

 Available Accounts
 ==================
 (0) 0x544b68e2329041e9bc92b61bfc8856ff3c9c8878
 (1) 0x9f03d97ff165cf2a0059c31afda611a6e8484053
 (2) 0x381b68c73705b17e06dc9c476cab12d45a9cd7f7
 (3) 0x26561efdad684e2f160d1ddf2c43cd9b20b0668c
 (4) 0xf81ccf7f6907a1c7a33b814f17156f31bab0ae45
 (5) 0x51a8a3c042c2a67fb038df6a8c0067113a41eacd
 (6) 0x90cf88722faa32ae685c8899a8cfe4400a42845d
 (7) 0xf667aea7a42afe8852b3a90ae5dcfe8c895d8a7d
 (8) 0x4d9cf67d8d2c0fff6760f7d272d9b80de94ed2b0
 (9) 0x7a9f131f5b86f7c7e726c0eaf7ff6dd0362851d9

 Private Keys
 ==================
 (0) c44c51e86f73d82cbab69b06ce2f6f9c4ec81640f7e4d0702ade95cfef3b2118
 (1) a4de036fdd0a66ab3c62f0e9e8ef0cba3b0ac3708db7cce7e28f4c93408d1211
 (2) b87ea877151eb195d8378718a9e91c1b5de07e81450499280e59cb7eafb820cb
 (3) 4f6d6c677491f2610466c7961664dc4f6192f65fe7416c53d8198fc07fc92bbe
 (4) 57e95816017e15cd0472e76ab523eed29ee6249f06f6b23f6c52899dfc86272c
 (5) f2e69cf981f5f944a54d32cf65ff59237c13859293ba8c9343e5c08c9cfc31c9
 (6) 5c84634203599927 36dace9d5559a97bd1c275f3a906c953e9aedc243f3fded7
 (7) 6b35db3f5dc56de6ebe3ceab6850de41df55e9938989fa344889e3377526d3be
 (8) 99546864a01f1df2cea26d548051d98c9e01fc7fe059b18890d68d4557c1a4cd
 (9) fd05c4d05ceeb2bbccaa5fbcd6d796bc62ff3c3907128ceab68a67a34f0ef787

 HD Wallet
 ==================
 Mnemonic:      voice mango engine whale brief fiction field brain work behave minute glad
 Base HD Path:  m/44'/60'/0'/0/{account_index}

 Listening on localhost:8545
```

TESTRPC - REMINDER

- When Deploying
  - Geth (go-ethereum): https://github.com/ethereum/go-ethereum
  - WebThree (cpp-ethereum): https://github.com/ethereum/webthree-umbrella

TRUFFLE

- Truffle is a development environment, testing framework and asset pipeline for Ethereum:
  - Smart contract compilation, library linking, deployment
  - Automated contract testing with Mocha and Chai

TRUFFLE - INSTALLATION

- `$ npm install -g truffle`

TRUFFLE - INITIALISATION

- Create project folder
  - $ mkdir myproject

- Initialize Your Project
  - $ cd myproject
  - $ truffle init

TRUFFLE - PROJECT STRUCTURE

- Project structure
  - app/ - application files. This includes recommended folders for Javascript files and stylesheets.
  - contracts/ - solidity contracts.
  - environments/ - environment configuration variables.
  - test/ - test files for your application and contracts.
  - truffle.js - configuration file.

TRUFFLE - COMPILE AND DEPLOY

- Compiling contracts
  - $ truffle compile
- Deploying contracts
  - Note: Ethereum client must be running.
  - $ truffle deploy

TRUFFLE – WATCH AND SERVE

- truffle watch - watch your filesystem for changes and recompile and redeploy your contracts
  - $ truffle watch

- truffle serve - watch your filesystem for changes and recompile, redeploy and rebuild, like truffle watch, and serve the built project on http://localhost:8080.
  - $ truffle serve

TRUFFLE - AUTOMATED TESTING

- Truffle uses the Mocha testing framework for automated testing and Chai for assertions
- Writing Tests
    - Use the contract() function when you're writing tests that interact with your contracts.
    - Before each contract() function is run, your contracts are redeployed to the running Ethereum client so the tests within it run with a clean contract state.
    - The contract() function provides a list of available accounts as a second parameter with which you can write tests against.

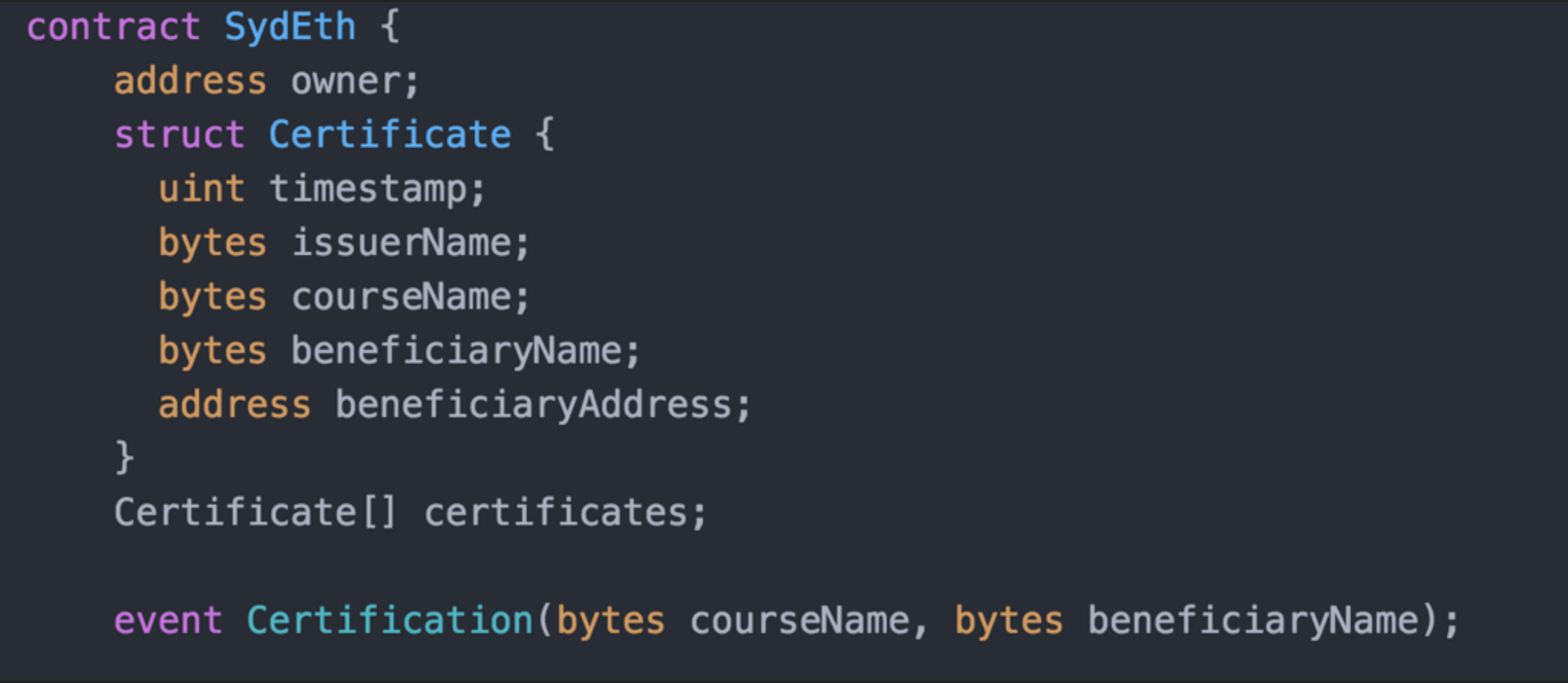
TRUFFLE - AUTOMATED TESTING

- Executing tests
  - Reminder: start testrpc
  - $ truffle test

BENEFITS OF AUTOMATED TESTING

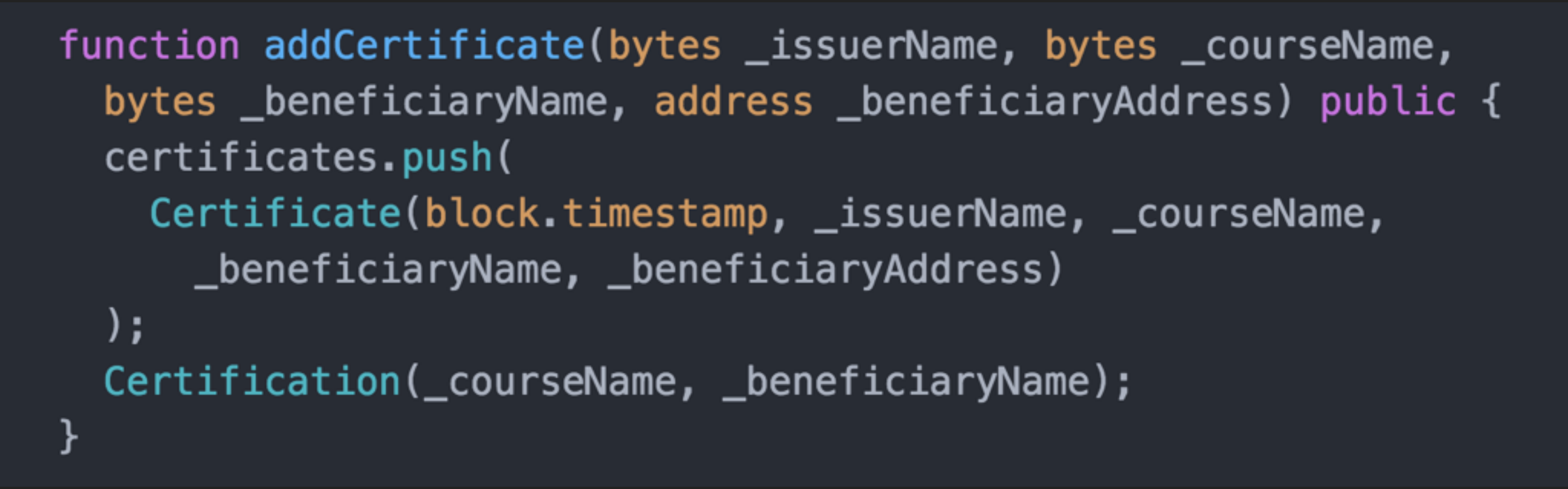
- Developer safety net
  - Less bugs
  - Less effort
- Check compliance with specifications
- Faster!

DEMO PROJECT – SMART CONTRACT



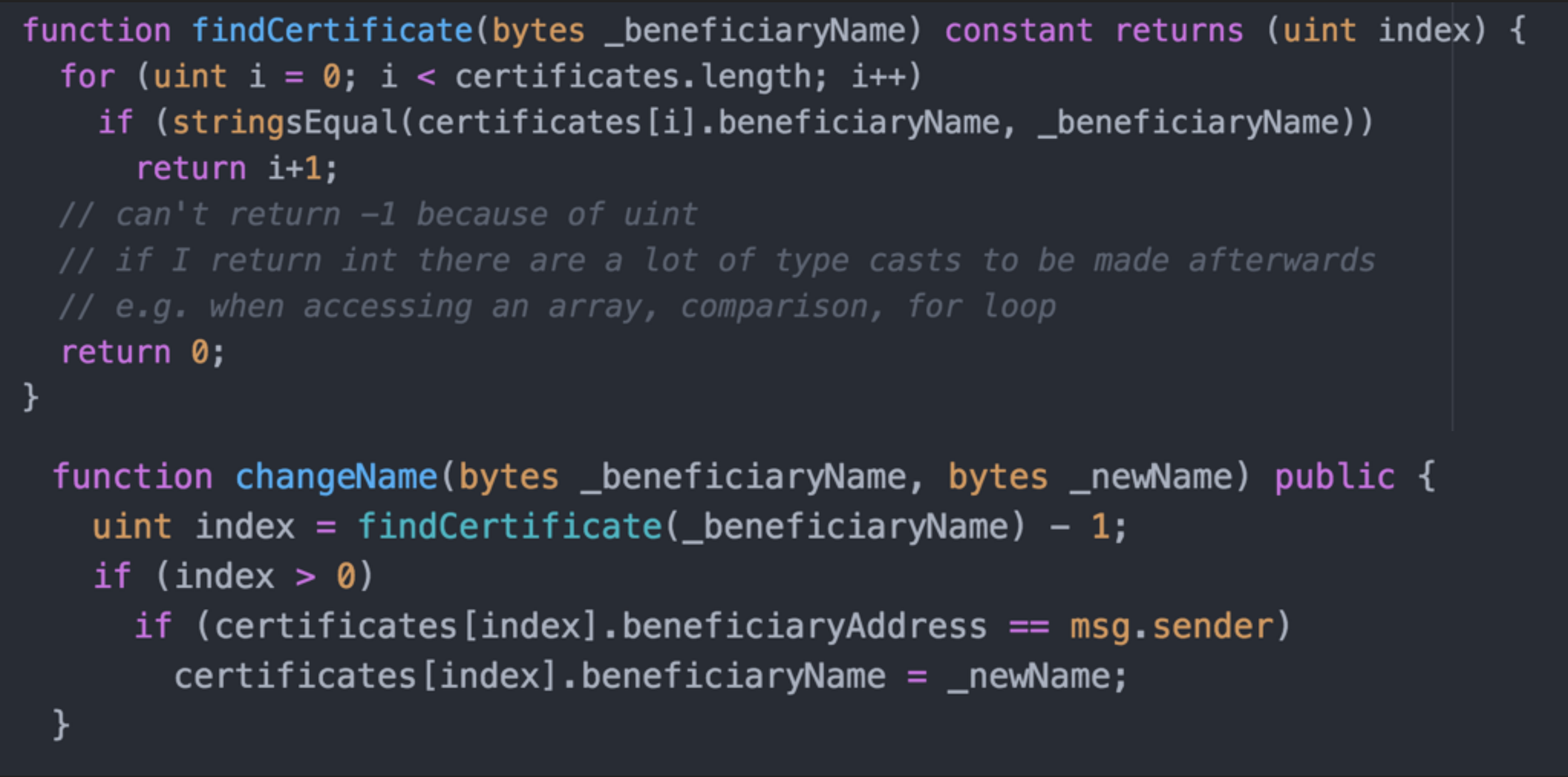
```
contract SydEth {
    address owner;
    struct Certificate {
      uint timestamp;
      bytes issuerName;
      bytes courseName;
      bytes beneficiaryName;
      address beneficiaryAddress;
    }
    Certificate[] certificates;

    event Certification(bytes courseName, bytes beneficiaryName);
```

DEMO PROJECT - SMART CONTRACT



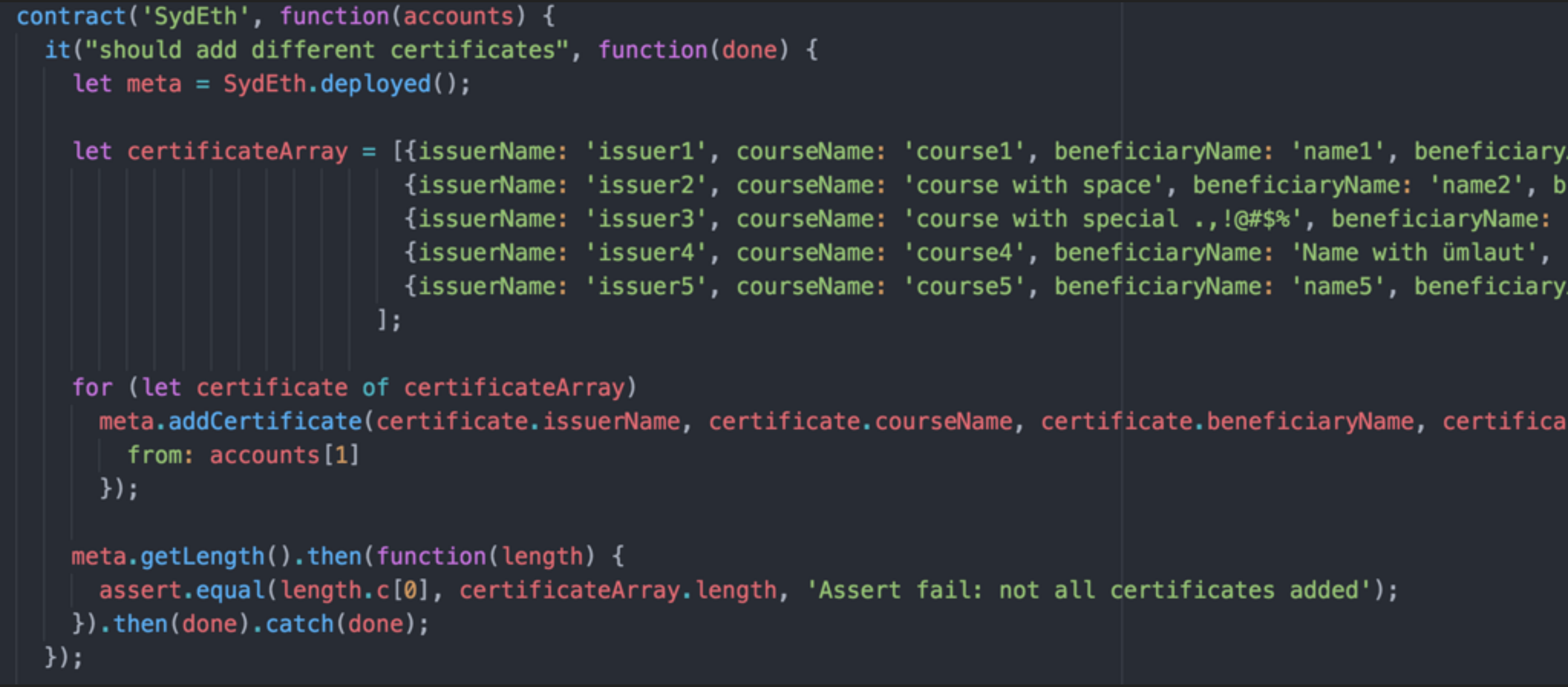
```
function addCertificate(bytes _issuerName, bytes _courseName,
  bytes _beneficiaryName, address _beneficiaryAddress) public {
  certificates.push(
    Certificate(block.timestamp, _issuerName, _courseName,
      _beneficiaryName, _beneficiaryAddress)
  );
  Certification(_courseName, _beneficiaryName);
}
```

DEMO PROJECT - SMART CONTRACT



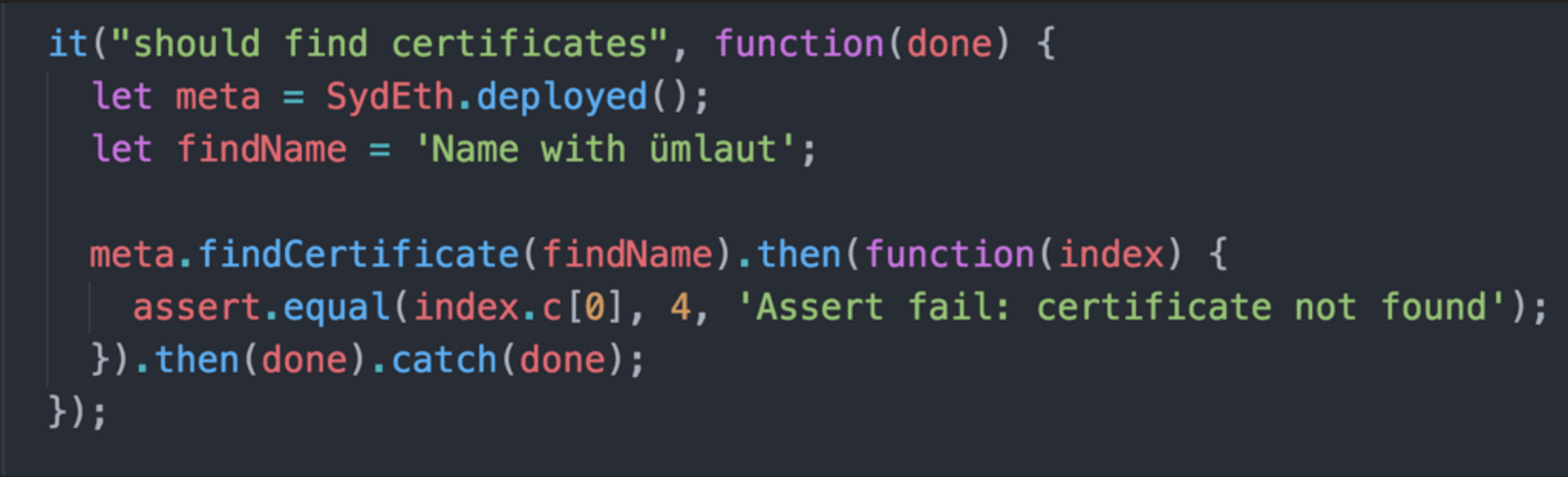
```
function findCertificate(bytes _beneficiaryName) constant returns (uint index) {
  for (uint i = 0; i < certificates.length; i++)
    if (stringsEqual(certificates[i].beneficiaryName, _beneficiaryName))
      return i+1;
  // can't return -1 because of uint
  // if I return int there are a lot of type casts to be made afterwards
  // e.g. when accessing an array, comparison, for loop
  return 0;
}

 function changeName(bytes _beneficiaryName, bytes _newName) public {
   uint index = findCertificate(_beneficiaryName) - 1;
   if (index > 0)
     if (certificates[index].beneficiaryAddress == msg.sender)
       certificates[index].beneficiaryName = _newName;
 }
```

DEMO PROJECT – AUTOMATIC TESTS



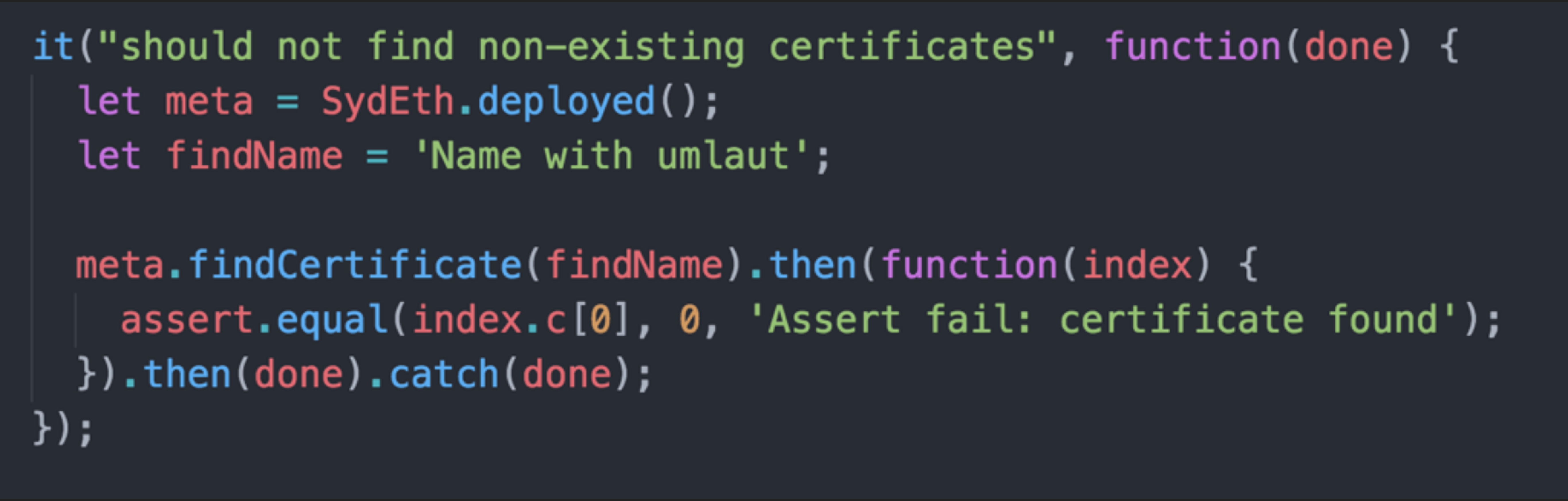
```
contract('SydEth', function(accounts) {
  it("should add different certificates", function(done) {
    let meta = SydEth.deployed();

    let certificateArray = [{issuerName: 'issuer1', courseName: 'course1', beneficiaryName: 'name1', beneficiary
                            {issuerName: 'issuer2', courseName: 'course with space', beneficiaryName: 'name2', b
                            {issuerName: 'issuer3', courseName: 'course with special .,!@#$%', beneficiaryName:
                            {issuerName: 'issuer4', courseName: 'course4', beneficiaryName: 'Name with ümlaut',
                            {issuerName: 'issuer5', courseName: 'course5', beneficiaryName: 'name5', beneficiary
                           ];

    for (let certificate of certificateArray)
      meta.addCertificate(certificate.issuerName, certificate.courseName, certificate.beneficiaryName, certifica
        from: accounts[1]
      });

    meta.getLength().then(function(length) {
      assert.equal(length.c[0], certificateArray.length, 'Assert fail: not all certificates added');
    }).then(done).catch(done);
  });
```

DEMO PROJECT - AUTOMATIC TESTS



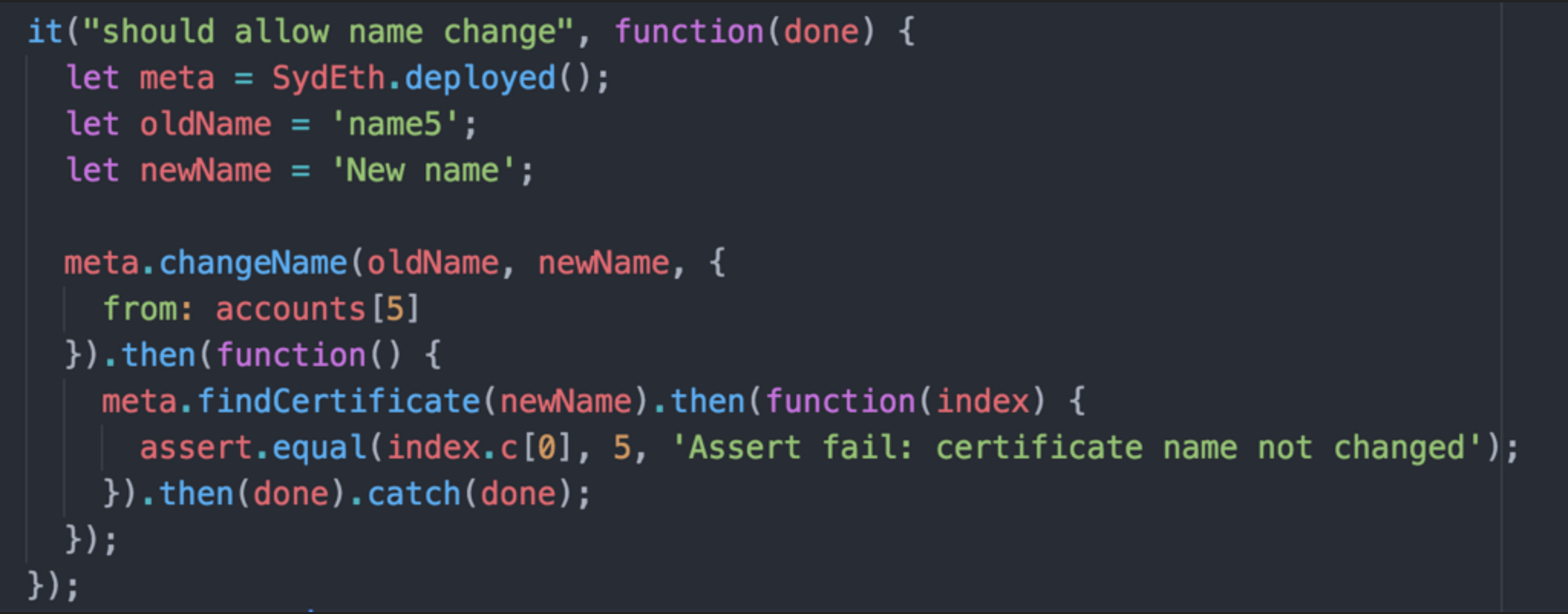
```
it("should find certificates", function(done) {
  let meta = SydEth.deployed();
  let findName = 'Name with ümlaut';

  meta.findCertificate(findName).then(function(index) {
    assert.equal(index.c[0], 4, 'Assert fail: certificate not found');
  }).then(done).catch(done);
});
```

DEMO PROJECT - AUTOMATIC TESTS



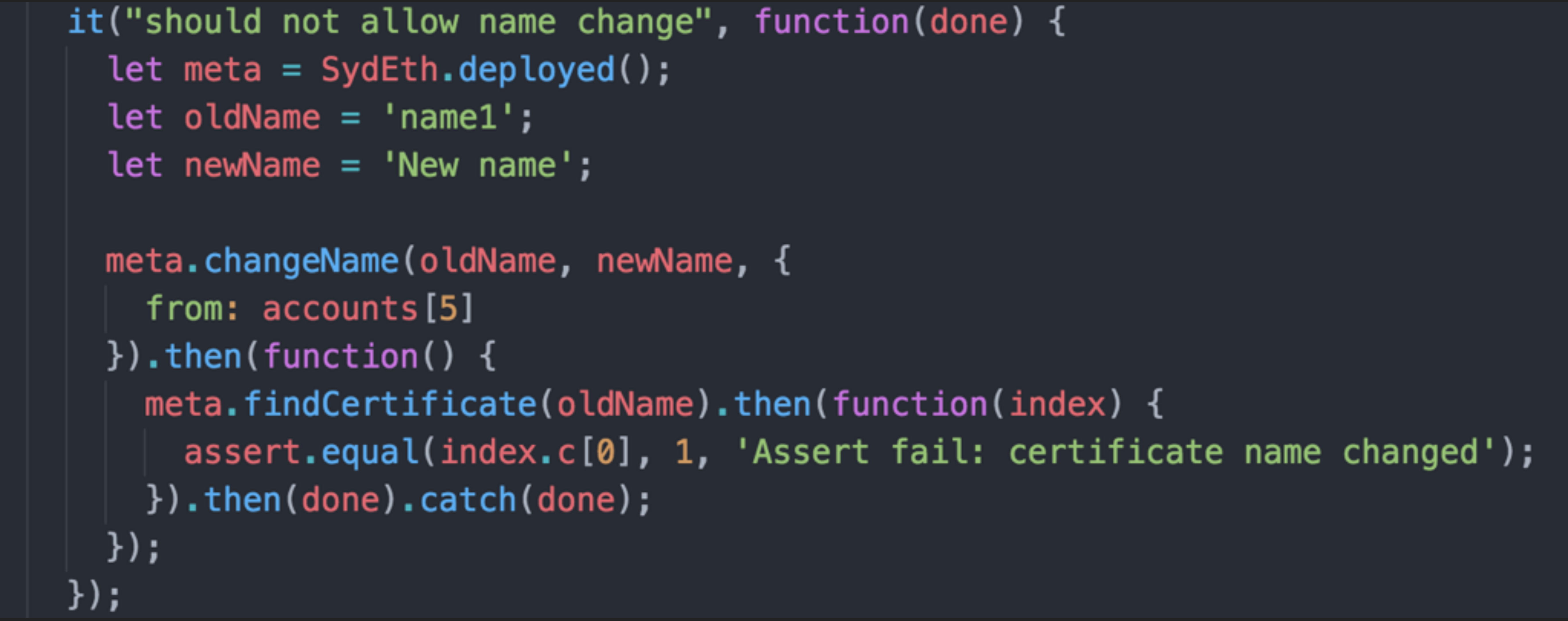
```
it("should not find non-existing certificates", function(done) {
  let meta = SydEth.deployed();
  let findName = 'Name with umlaut';

  meta.findCertificate(findName).then(function(index) {
    assert.equal(index.c[0], 0, 'Assert fail: certificate found');
  }).then(done).catch(done);
});
```

DEMO PROJECT - AUTOMATIC TESTS



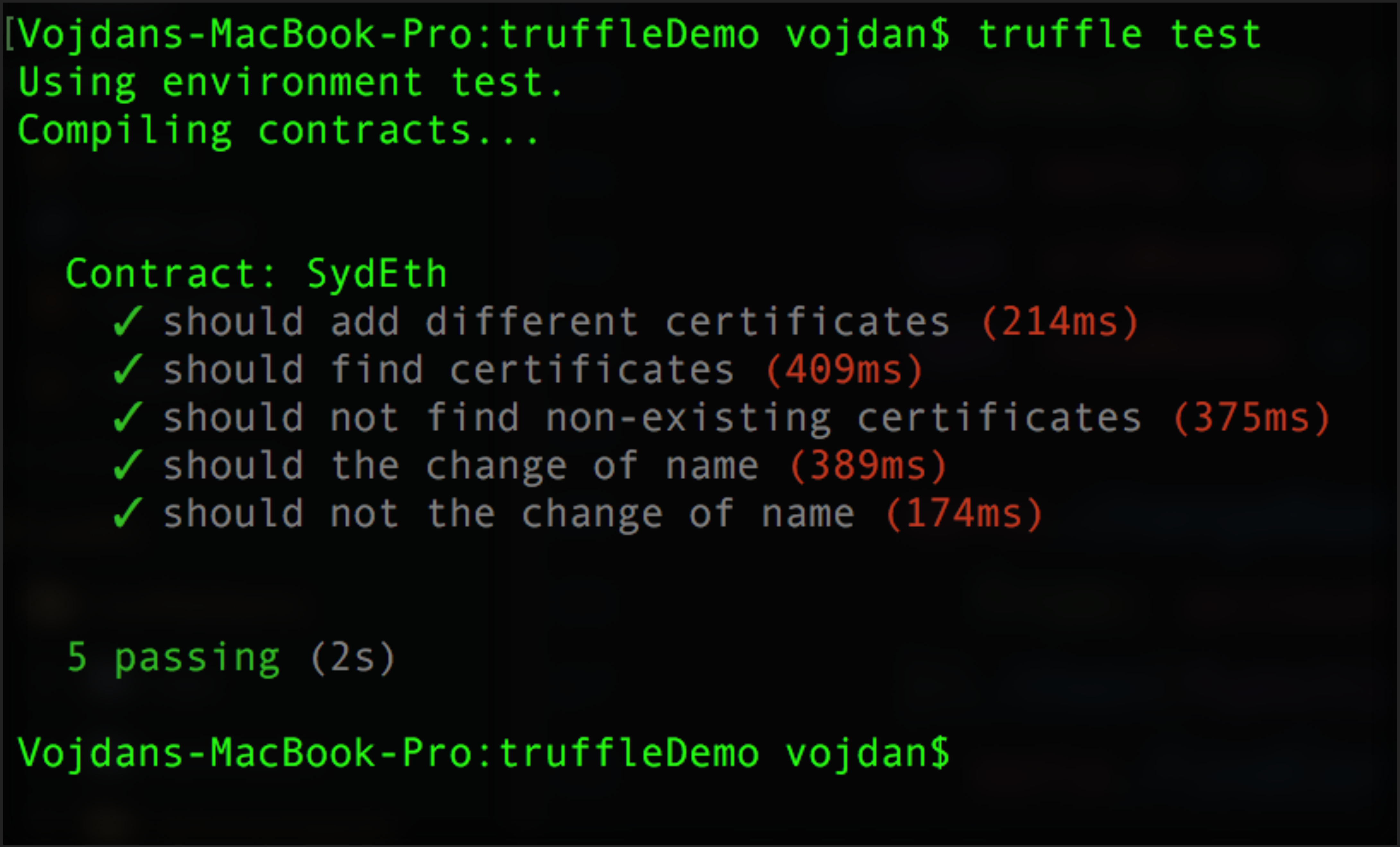
```
it("should allow name change", function(done) {
  let meta = SydEth.deployed();
  let oldName = 'name5';
  let newName = 'New name';

  meta.changeName(oldName, newName, {
    from: accounts[5]
  }).then(function() {
    meta.findCertificate(newName).then(function(index) {
      assert.equal(index.c[0], 5, 'Assert fail: certificate name not changed');
    }).then(done).catch(done);
  });
});
```

DEMO PROJECT - AUTOMATIC TESTS



```
it("should not allow name change", function(done) {
  let meta = SydEth.deployed();
  let oldName = 'name1';
  let newName = 'New name';

  meta.changeName(oldName, newName, {
    from: accounts[5]
  }).then(function() {
    meta.findCertificate(oldName).then(function(index) {
      assert.equal(index.c[0], 1, 'Assert fail: certificate name changed');
    }).then(done).catch(done);
  });
});
```

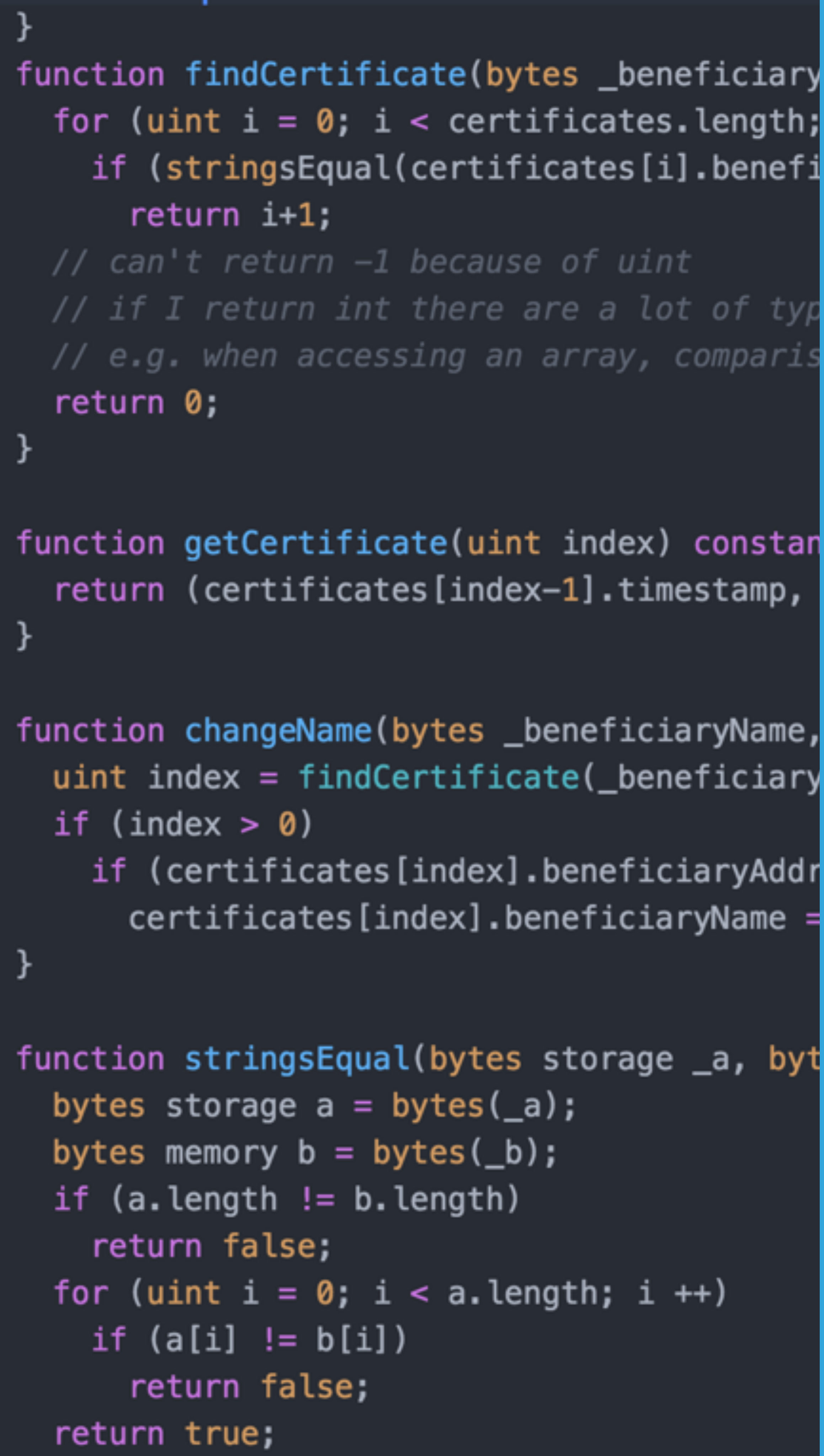
TRUFFLE - AUTOMATED TESTING



```
[Vojdans-MacBook-Pro:truffleDemo vojdan$ truffle test
Using environment test.
Compiling contracts...


  Contract: SydEth
    ✓ should add different certificates (214ms)
    ✓ should find certificates (409ms)
    ✓ should not find non-existing certificates (375ms)
    ✓ should the change of name (389ms)
    ✓ should not the change of name (174ms)


  5 passing (2s)

Vojdans-MacBook-Pro:truffleDemo vojdan$
```

DEVELOPMENT ENVIRONMENT

- Desktop
  - Atom (https://atom.io/)
- Web based
  - Browser solidity (https://ethereum.github.io/browser-solidity/)
  - EtherCamp (https://live.ether.camp/)



```
}
function findCertificate(bytes _beneficiary
  for (uint i = 0; i < certificates.length;
    if (stringsEqual(certificates[i].benefi
      return i+1;
  // can't return -1 because of uint
  // if I return int there are a lot of typ
  // e.g. when accessing an array, comparis
  return 0;
}

function getCertificate(uint index) constan
  return (certificates[index-1].timestamp,
}

function changeName(bytes _beneficiaryName,
  uint index = findCertificate(_beneficiary
  if (index > 0)
    if (certificates[index].beneficiaryAddr
      certificates[index].beneficiaryName =
}

function stringsEqual(bytes storage _a, byt
  bytes storage a = bytes(_a);
  bytes memory b = bytes(_b);
  if (a.length != b.length)
    return false;
  for (uint i = 0; i < a.length; i ++)
    if (a[i] != b[i])
      return false;
  return true;
```

QUESTIONS?

Good luck!