Continuous Integration

This walktrough tutorial covers the basics of using remix libraries in your Continuous Integration

Continuous Integration

Remix libraries are located in

https://github.com/ethereum/remix

They can be used for several purposes:

- Compiling solidity file
- Running static Analysis
- Running unit testing
- Debugging transaction

Continuous Integration

We will cover in these slides how to:

- Compile solidity files
- Run static analysis
- Run Solidity unit testing

… in your Continuous Integration.

Continuous Integration

Remix libraries are JavaScript libraries thus they need to be run in a JavaScript runtime (e.g node).

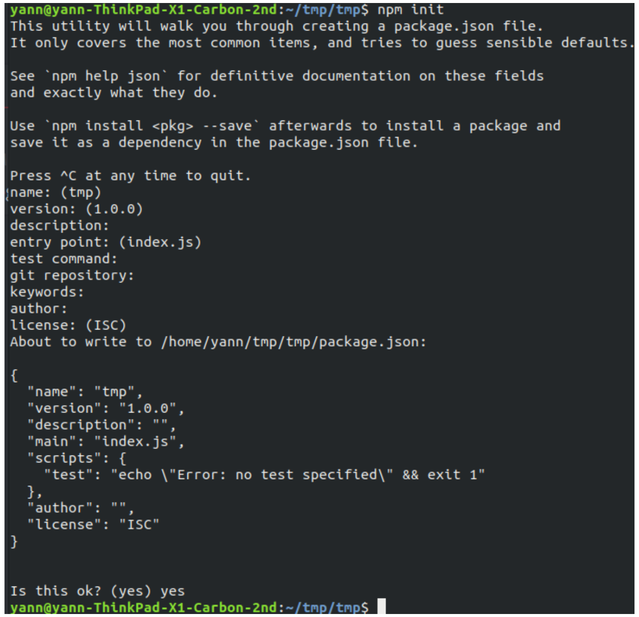
All the Remix libraries are deployed to NPM:

- https://www.npmjs.com/package/remix-solidity
- https://www.npmjs.com/package/remix-analyzer
- https://www.npmjs.com/package/remix-tests

Continuous Integration

For the purpose of this tutorial, we will create a new NPM package.

Run `npm init` in the console, and validate all the default parameters.



```
yann@yann-ThinkPad-X1-Carbon-2nd:~/tmp/tmp$ npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.

See `npm help json` for definitive documentation on these fields
and exactly what they do.

Use `npm install <pkg> --save` afterwards to install a package and
save it as a dependency in the package.json file.

Press ^C at any time to quit.
name: (tmp)
version: (1.0.0)
description:
entry point: (index.js)
test command:
git repository:
keywords:
author:
license: (ISC)
About to write to /home/yann/tmp/tmp/package.json:

{
  "name": "tmp",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "",
  "license": "ISC"
}


Is this ok? (yes) yes
yann@yann-ThinkPad-X1-Carbon-2nd:~/tmp/tmp$
```

Continuous Integration

We will import 3 files:

- `Sample.sol` – source contract.
- `Sample_test.sol` – solidity unit testing.
- `main.js` – Continous Integration script.

Continuous Integration

Please download all the files from

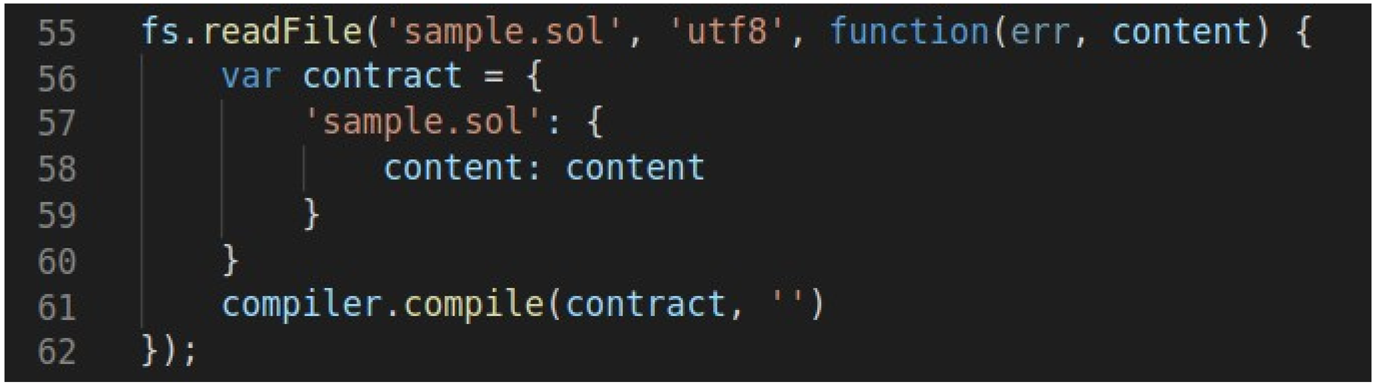
https://github.com/ethereum/remix-workshops/tree/master/continuous_integration

and put them in the working folder.

Continuous Integration

Let’s look at main.js. It include 3 steps:

– Compiling the main solidity file – line 61

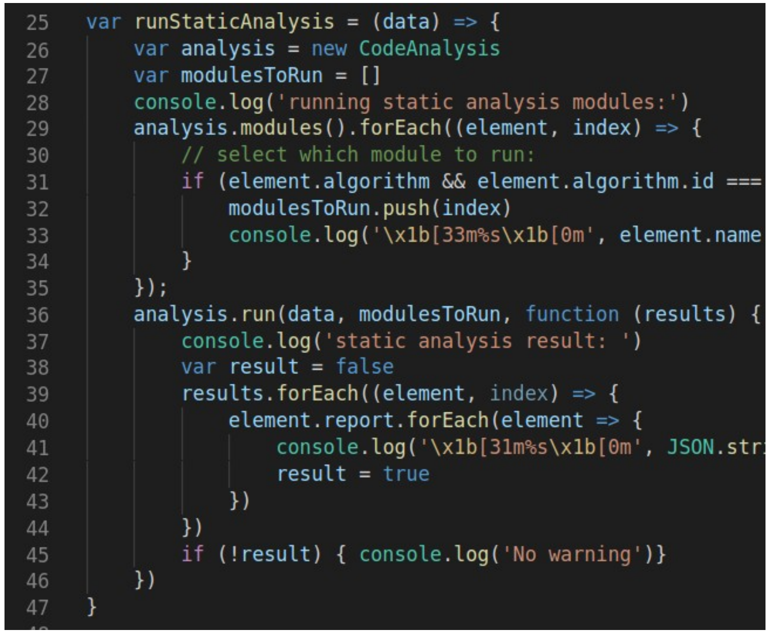


```
55  fs.readFile('sample.sol', 'utf8', function(err, content) {
56      var contract = {
57          'sample.sol': {
58              content: content
59          }
60      }
61      compiler.compile(contract, '')
62  });
```

Continuous Integration

- Running static analysis against some selected modules – line 36

  that way it is possible to trigger a failure of integration process if some specific reports are returned.



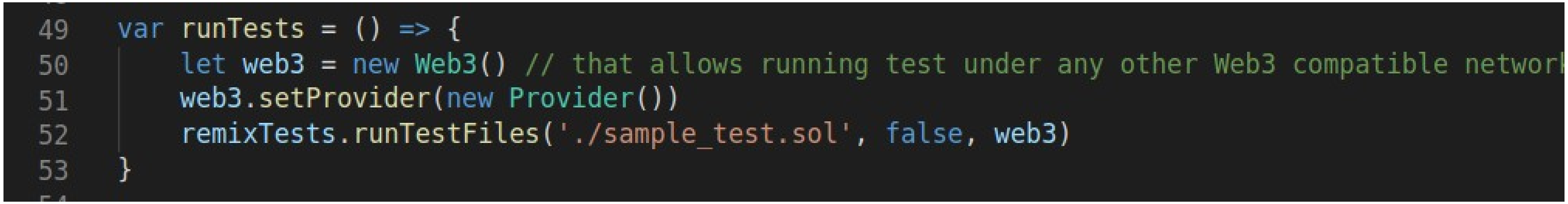
```
25  var runStaticAnalysis = (data) => {
26      var analysis = new CodeAnalysis
27      var modulesToRun = []
28      console.log('running static analysis modules:')
29      analysis.modules().forEach((element, index) => {
30          // select which module to run:
31          if (element.algorithm && element.algorithm.id ===
32              modulesToRun.push(index)
33              console.log('\x1b[33m%s\x1b[0m', element.name
34          }
35      });
36      analysis.run(data, modulesToRun, function (results) {
37          console.log('static analysis result: ')
38          var result = false
39          results.forEach((element, index) => {
40              element.report.forEach(element => {
41                  console.log('\x1b[31m%s\x1b[0m', JSON.str
42                  result = true
43              })
44          })
45          if (!result) { console.log('No warning')}
46      })
47  }
```

Continuous Integration

– Running unit tests – line 52

  Please see sample_test.sol

  Note that the ‘testing’ tab of Remix IDE (remix.ethereum.org) execute the exact same library.



```
49  var runTests = () => {
50      let web3 = new Web3() // that allows running test under any other Web3 compatible networ
51      web3.setProvider(new Provider())
52      remixTests.runTestFiles('./sample_test.sol', false, web3)
53  }
```

For the Remix IDE related documentation:
https://remix.readthedocs.io/en/latest/unittesting_tab.html

Continuous Integration

For simplicity of this tutorial, `main.js` does not include a linter.

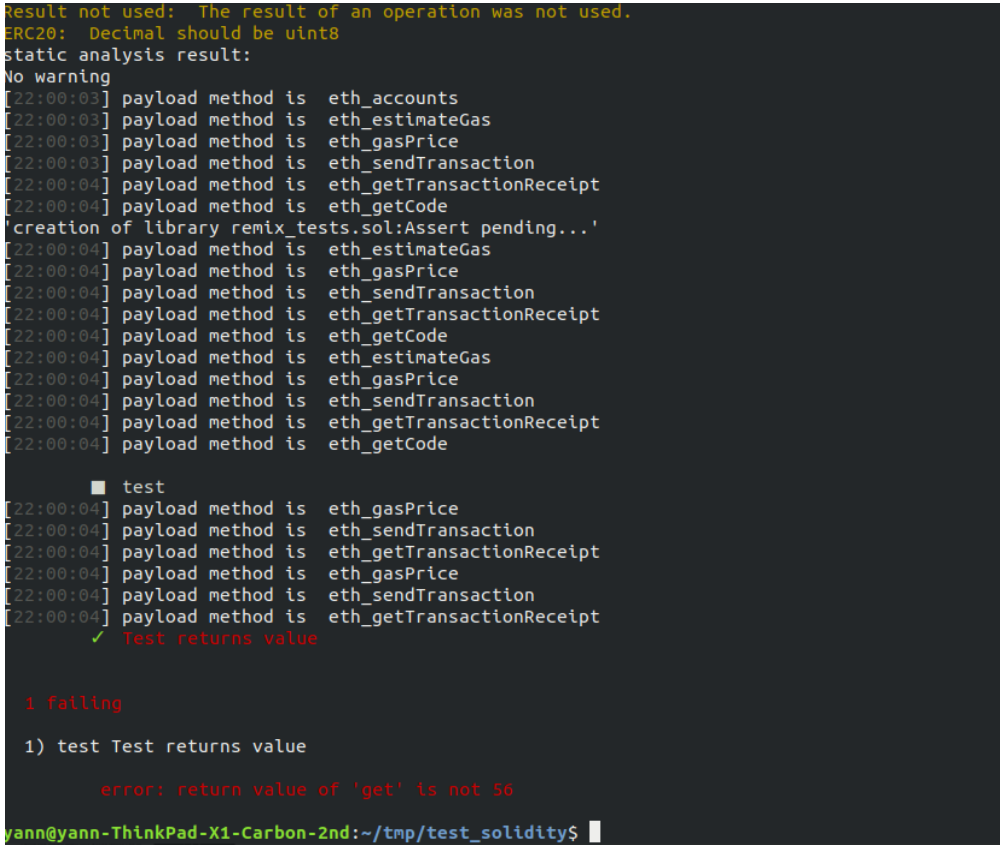
Please check out `Solium`:

`https://www.npmjs.com/package/solium`

Continuous Integration

Finally, let’s run node main.js

Fix the error in unit testing
And re execute main.js



```
Result not used:  The result of an operation was not used.
ERC20:  Decimal should be uint8
static analysis result:
No warning
[22:00:03] payload method is  eth_accounts
[22:00:03] payload method is  eth_estimateGas
[22:00:03] payload method is  eth_gasPrice
[22:00:03] payload method is  eth_sendTransaction
[22:00:04] payload method is  eth_getTransactionReceipt
[22:00:04] payload method is  eth_getCode
'creation of library remix_tests.sol:Assert pending...'
[22:00:04] payload method is  eth_estimateGas
[22:00:04] payload method is  eth_gasPrice
[22:00:04] payload method is  eth_sendTransaction
[22:00:04] payload method is  eth_getTransactionReceipt
[22:00:04] payload method is  eth_getCode
[22:00:04] payload method is  eth_estimateGas
[22:00:04] payload method is  eth_gasPrice
[22:00:04] payload method is  eth_sendTransaction
[22:00:04] payload method is  eth_getTransactionReceipt
[22:00:04] payload method is  eth_getCode

        ■  test
[22:00:04] payload method is  eth_gasPrice
[22:00:04] payload method is  eth_sendTransaction
[22:00:04] payload method is  eth_getTransactionReceipt
[22:00:04] payload method is  eth_gasPrice
[22:00:04] payload method is  eth_sendTransaction
[22:00:04] payload method is  eth_getTransactionReceipt
        ✓  Test returns value


  1 failing

  1) test Test returns value

         error: return value of 'get' is not 56

yann@yann-ThinkPad-X1-Carbon-2nd:~/tmp/test_solidity$ 
```