Supercharge your test suite with Scribble

Scribble is a specification language that we built at ConsenSys diligence. The core idea is to add comments to your code that say what the code is supposed to do!

//#if_succeeds old(balanceOf(to)) + value == balanceOf(to); function

transfer (to,

value)

public

returns

(bool)

{} Scribbling your contracts isn't just for documentation purposes! Using the Scribble CLI, we translate those comments into solidity code that checks that the annotation is correct! We call this processinstrumentation .

In this case Scribble will create something which looks like this:

function

transfer (to,

value)

public

returns

(bool)

{

uint

oldBalance

=

balanceOf(to);

...

assert(oldBalance +

value

balanceOf(to));

return

true ; }

Testing¶

What if I told you we could use Scribble to upgrade your regular old unit test suite!

If you look back to the transfer example above, Scribble insertsSolidity assertions to check properties. Fortunately for us, Truffle test cases will report a failure whenever there is a solidity level assertion.

Imagine a scenario where you have a bug, but your test suite (with 100% coverage) doesn't have the assertions to detect it.

This is where Scribble will come in handy! Having written some annotations, you can enhance your test suite with tons of extra checks. The chances are that one of these checks will trigger because of the bug!

Your unit tests are levelled up by Scribble annotations through the tons of assertions that meticulously (and automatically) get added by the CLI! It's often easier to write a Scribble annotation than to do the actual assertion in javascript. Take the following example. Here I add a single annotation that will ensure that all tests check that the total supply is equal to the sum of balances for my Token.

///#invariant unchecked_sum(balances) == totalSupply; contract

MyToken

{

...

} Now any test that modifies thetotalSupply or an entry in thebalances mapping will check that they are in sync!

How to do It¶

There are four simple steps to run your first Scribble enhanced test.

1. Install Scribble
2. Write an annotation
3. Run Scribble instrumentation
4. Run your test suite

Installation¶

First things first, you're going to install Scribble:

npm install -g eth-scribble Pro-tip: If you're using VSCode then install our vscode-scribble extension .

Writing Annotations¶

Next, you're going to write an annotation. The exact annotation that you'd write is particular to your smart contracts, but you can use the earlier example for inspiration!

///#if_succeeds old(balanceOf(to)) + value == balanceOf(to); function

transfer (to ,

value)

public

returns

(bool)

{} Check out the Scribble documentation on writing annotations.

Instrumentation and Testing¶

We're ready to start testing with the Scribble annotation(s) in place. First, you'll use the Scribble CLI to take all the annotations and transform them into solidity code. Next, you'll run your test suite as you usually would; but now with the added benefit of the Scribble instrumentation. Finally, you clean up the Scribble instrumentation.

You can do all this by executing the following three commands(replacing myToken.sol with the filename for the contract you just annotated):

scribble --arm -m files myToken.sol

truffle test

scribble --disarm -m files myToken.sol

Next Steps¶

That's it! You now know all you need to start supercharging your unit tests.

Your next step after levelling up your unit tests is to use fuzzing. Fuzzing finds bugs by automatically generating millions of test cases. The chances are that it will come up with several edge cases that you missed with your manual unit tests!

Read more about fuzzing Scribble annotations here !