STANDING THE TIME OF TEST

with Truffle and Ganache

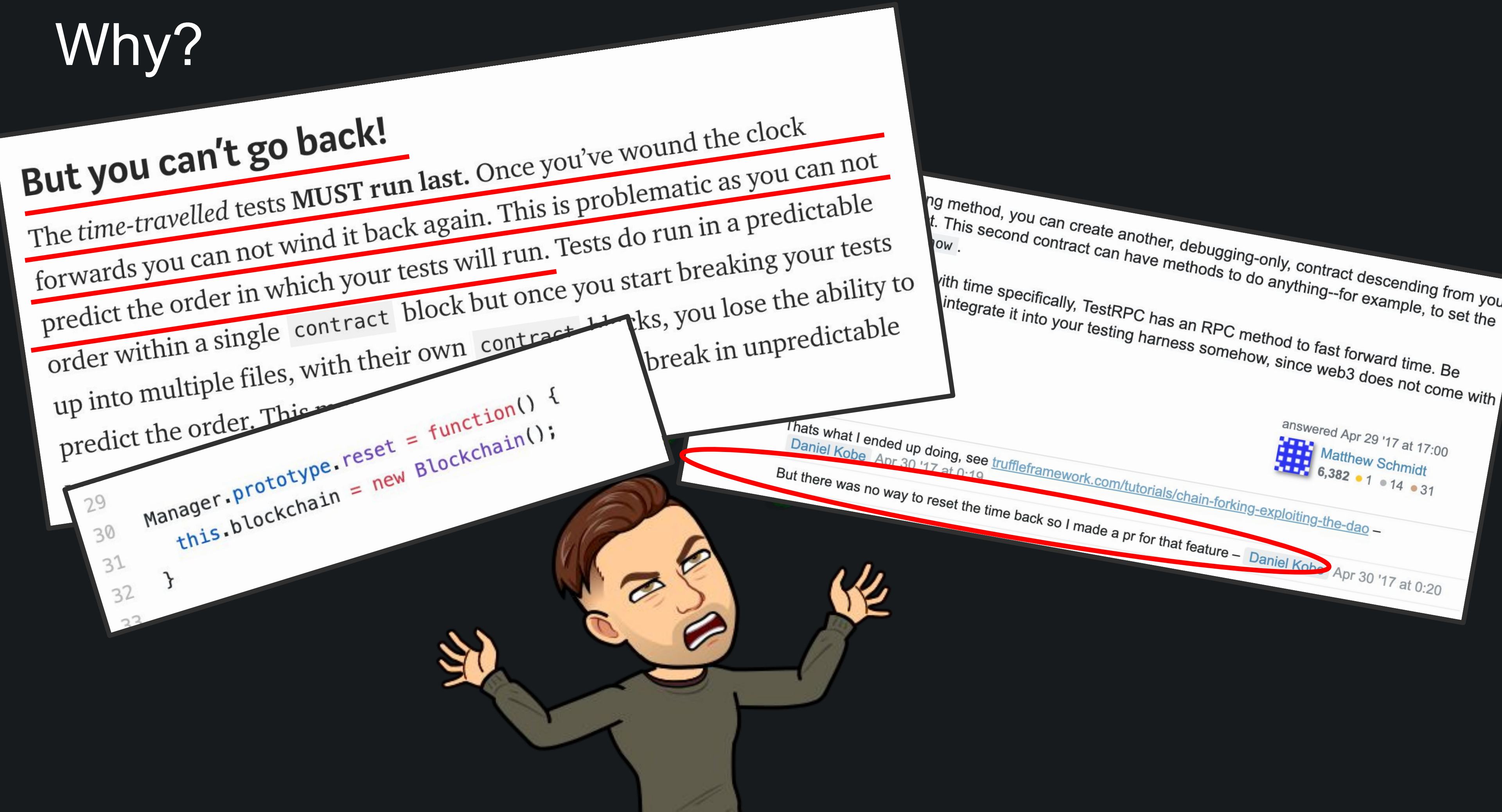
Ethan Wessel

What’s this about?

This presentation explores testing time-dependent code along with methods and tooling that may not be widely known.

The methods demonstrated can empower the developer community to write more isolated tests resulting in safer and more robust contracts.

Why?



But you can't go back!

The *time-travelled* tests **MUST run last.** Once you've wound the clock forwards you can not wind it back again. This is problematic as you can not predict the order in which your tests will run. Tests do run in a predictable order within a single `contract` block but once you start breaking your tests up into multiple files, with their own `contract` blocks, you lose the ability to predict the order. This m break in unpredictable

```
Manager.prototype.reset = function() {
  this.blockchain = new Blockchain();
}
```

ng method, you can create another, debugging-only, contract descending from you t. This second contract can have methods to do anything--for example, to set the ow .

ith time specifically, TestRPC has an RPC method to fast forward time. Be integrate it into your testing harness somehow, since web3 does not come with

answered Apr 29 '17 at 17:00
Matthew Schmidt
6,382 ●1 ●14 ●31

Thats what I ended up doing, see truffleframework.com/tutorials/chain-forking-exploiting-the-dao – Daniel Kobe Apr 30 '17 at 0:19

But there was no way to reset the time back so I made a pr for that feature – Daniel Kobe Apr 30 '17 at 0:20

Disclosure

There is an excellent Medium article by Andy Watt about testing on the Ethereum blockchain which provides useful insights and helper functions

The following illuminates a bit of experience on writing tests and building off Andy's findings (amongst others).

Setting time in `ganache-cli`

```
$ ganache-cli --time '2019-02-15T15:53:00+00:00'
```

Fri Feb 15 15:53:00 UTC 2019

Added in October 2018 and a part of the 6.2.0 release.

Setting time in `ganache-cli` (cont)

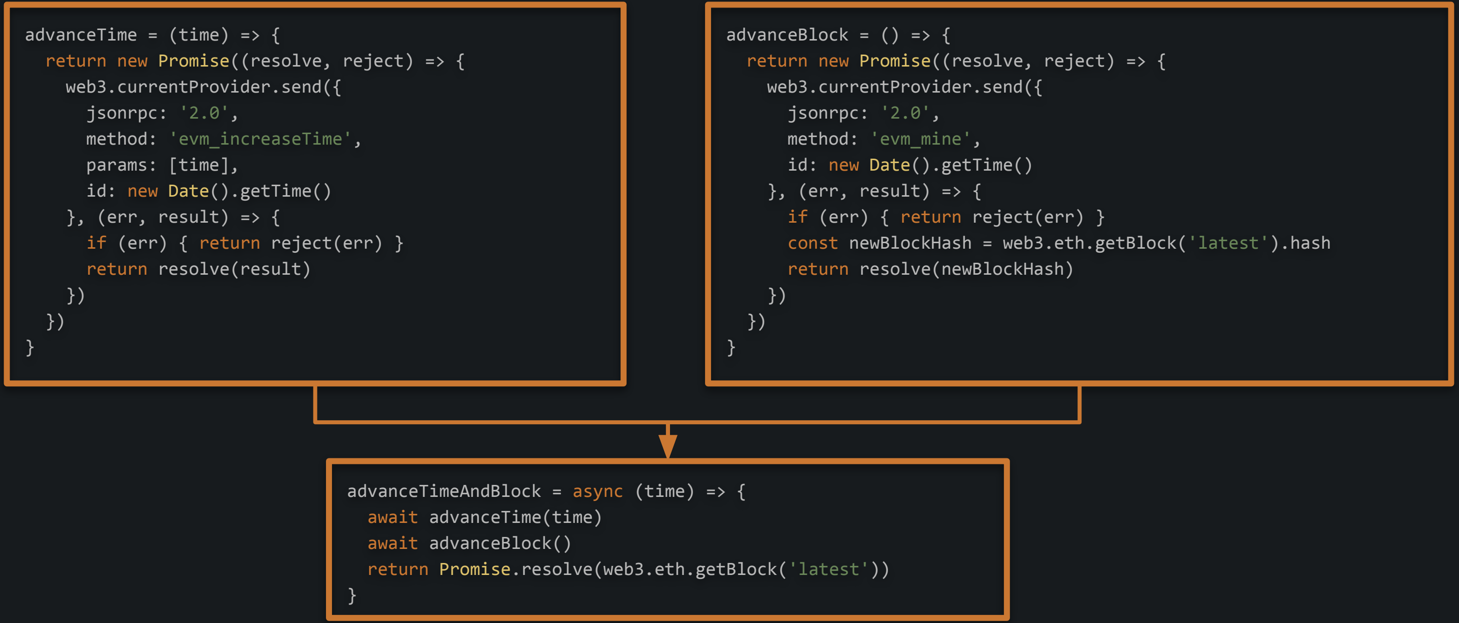
```
$ truffle console
truffle(development)> blockNum = await web3.eth.getBlockNumber()
undefined
truffle(development)> blockNum
0
truffle(development)> block = await web3.eth.getBlock(blockNum)
undefined
truffle(development)> block['timestamp']
1550245980

$ date -u -r "1550245980"
Fri Feb 15 15:53:00 UTC 2019
```

JSON RPC API Methods

```
> +-----------------------------+-----------------------------+
> |           Action            |      Ganache EVM Name       |
> +-----------------------------+-----------------------------+
> | Advance the time            | evm_increaseTime            |
> | Advance the block(s)        | evm_mine                    |
> | Advance both time and block | evm_increaseTime + evm_mine |
> | Save time                   | evm_snapshot                |
> | Revert time                 | evm_revert                  |
> +-----------------------------+-----------------------------+
```

These are now denoted on https://github.com/trufflesuite/ganache-cli

Jumping Forward in time



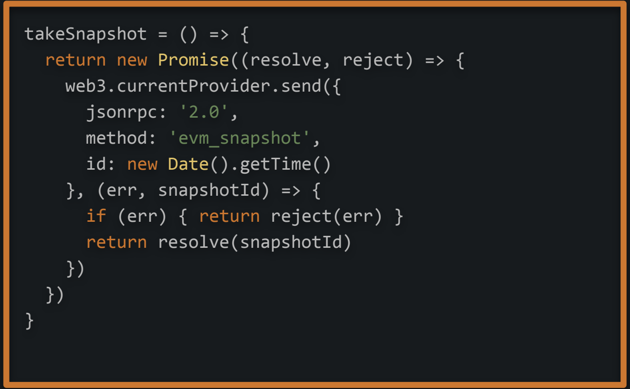
```
advanceTime = (time) => {
  return new Promise((resolve, reject) => {
    web3.currentProvider.send({
      jsonrpc: '2.0',
      method: 'evm_increaseTime',
      params: [time],
      id: new Date().getTime()
    }, (err, result) => {
      if (err) { return reject(err) }
      return resolve(result)
    })
  })
}
```

```
advanceBlock = () => {
  return new Promise((resolve, reject) => {
    web3.currentProvider.send({
      jsonrpc: '2.0',
      method: 'evm_mine',
      id: new Date().getTime()
    }, (err, result) => {
      if (err) { return reject(err) }
      const newBlockHash = web3.eth.getBlock('latest').hash
      return resolve(newBlockHash)
    })
  })
}
```

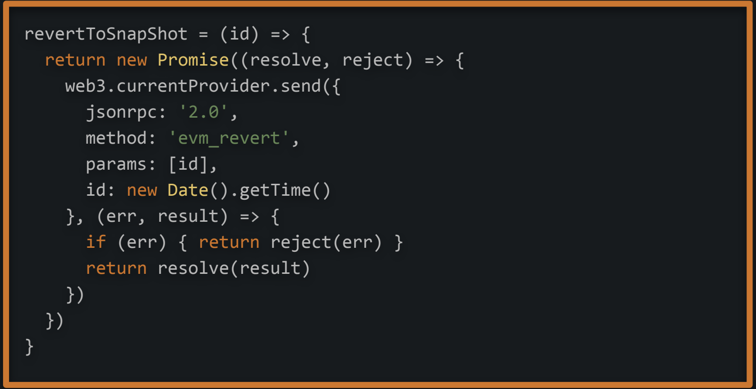
```
advanceTimeAndBlock = async (time) => {
  await advanceTime(time)
  await advanceBlock()
  return Promise.resolve(web3.eth.getBlock('latest'))
}
```

```
await helper.advanceTimeAndBlock(SECONDS_IN_DAY * 100); //advance 100 days
```

Jumping Backward in time



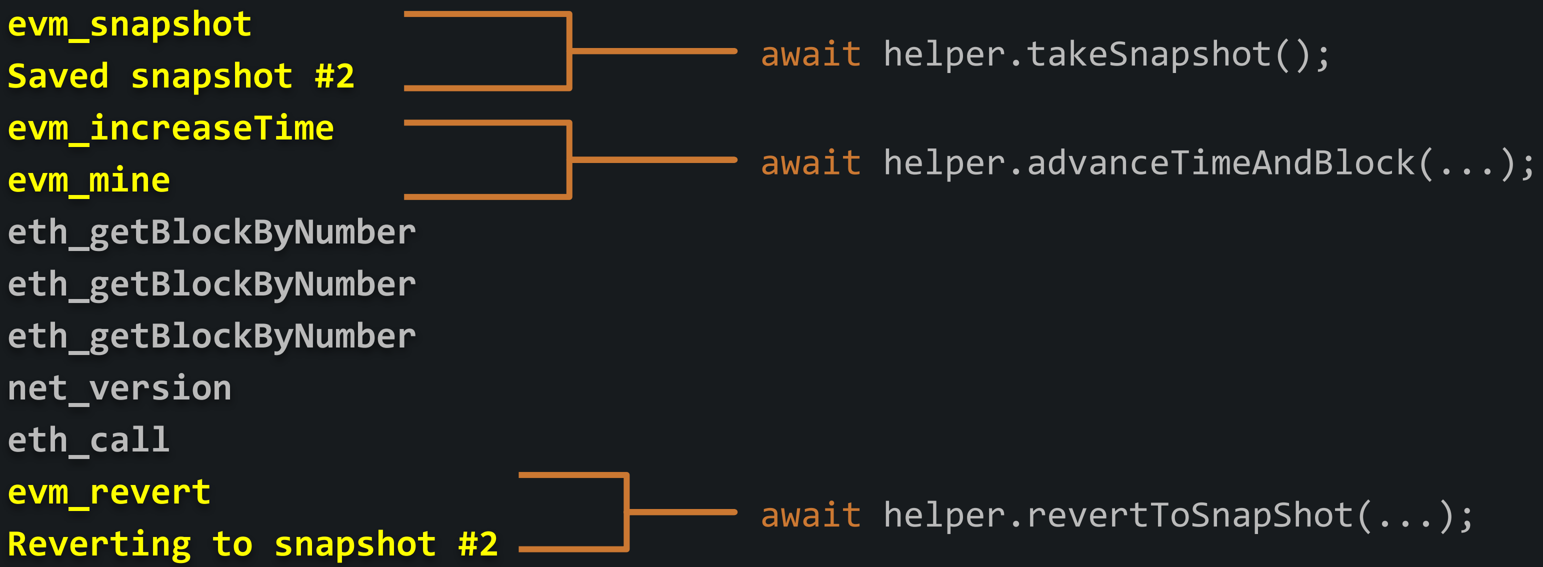
```
takeSnapshot = () => {
  return new Promise((resolve, reject) => {
    web3.currentProvider.send({
      jsonrpc: '2.0',
      method: 'evm_snapshot',
      id: new Date().getTime()
    }, (err, snapshotId) => {
      if (err) { return reject(err) }
      return resolve(snapshotId)
    })
  })
}
```



```
revertToSnapShot = (id) => {
  return new Promise((resolve, reject) => {
    web3.currentProvider.send({
      jsonrpc: '2.0',
      method: 'evm_revert',
      params: [id],
      id: new Date().getTime()
    }, (err, result) => {
      if (err) { return reject(err) }
      return resolve(result)
    })
  })
}
```

```
snapShot = await helper.takeSnapshot();
snapshotId = snapShot['result'];
await helper.revertToSnapShot(snapshotId);
```

ganache-cli Output



```
evm_snapshot
Saved snapshot #2
evm_increaseTime
evm_mine
eth_getBlockByNumber
eth_getBlockByNumber
eth_getBlockByNumber
net_version
eth_call
evm_revert
Reverting to snapshot #2
```

```
await helper.takeSnapshot();
```

```
await helper.advanceTimeAndBlock(...);
```

```
await helper.revertToSnapShot(...);
```

Quick Recap

- We talked about setting time with ganache-cli
- We talked about moving time forward with `evm_increaseTime` and `evm_mine`
- We talked about moving time backward with `evm_snapshot` and `evm_revert`

Let's combine these to make tests that don't depend on one another!!

Note: Truffle does this on a contact level, but not on a test level

Combine Learnings and Bend Time!

1. Add the following to `beforeEach()`

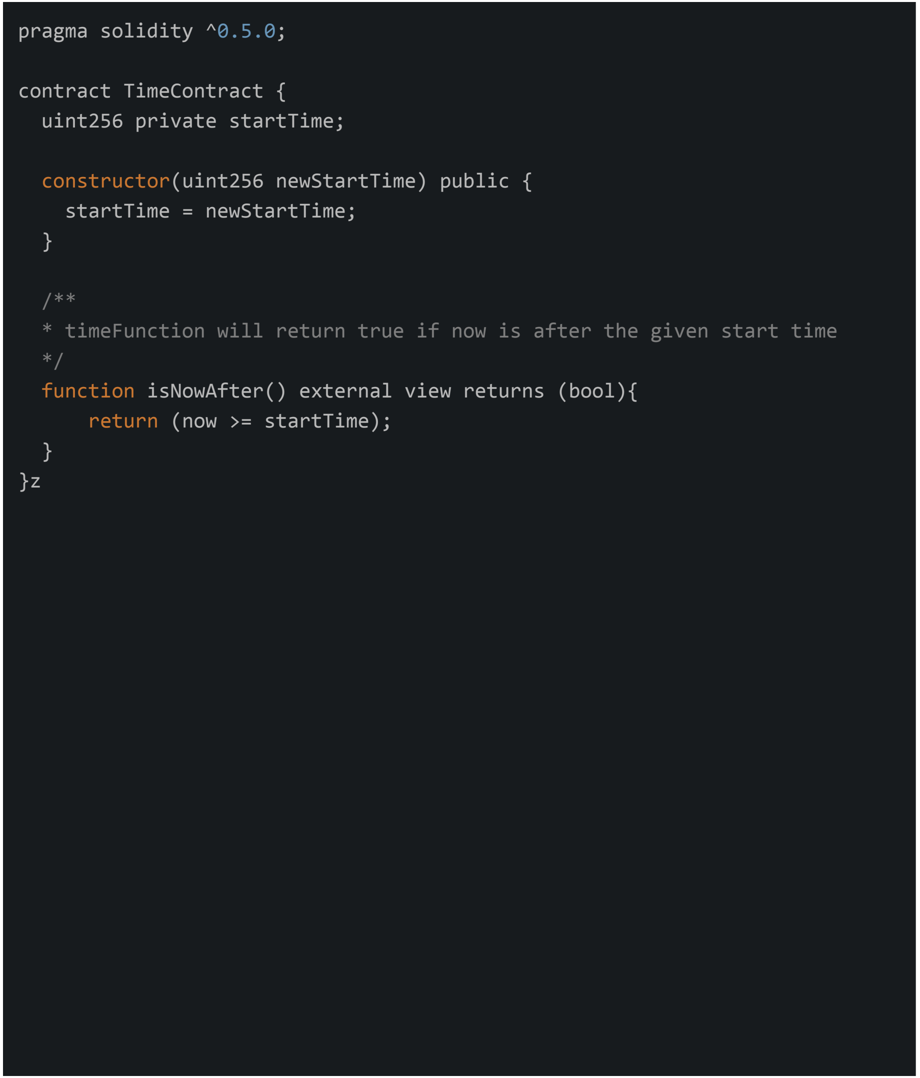
```
snapShot = await helper.takeSnapshot();
snapshotId = snapShot['result'];
```

2. Add the following to `afterEach()`

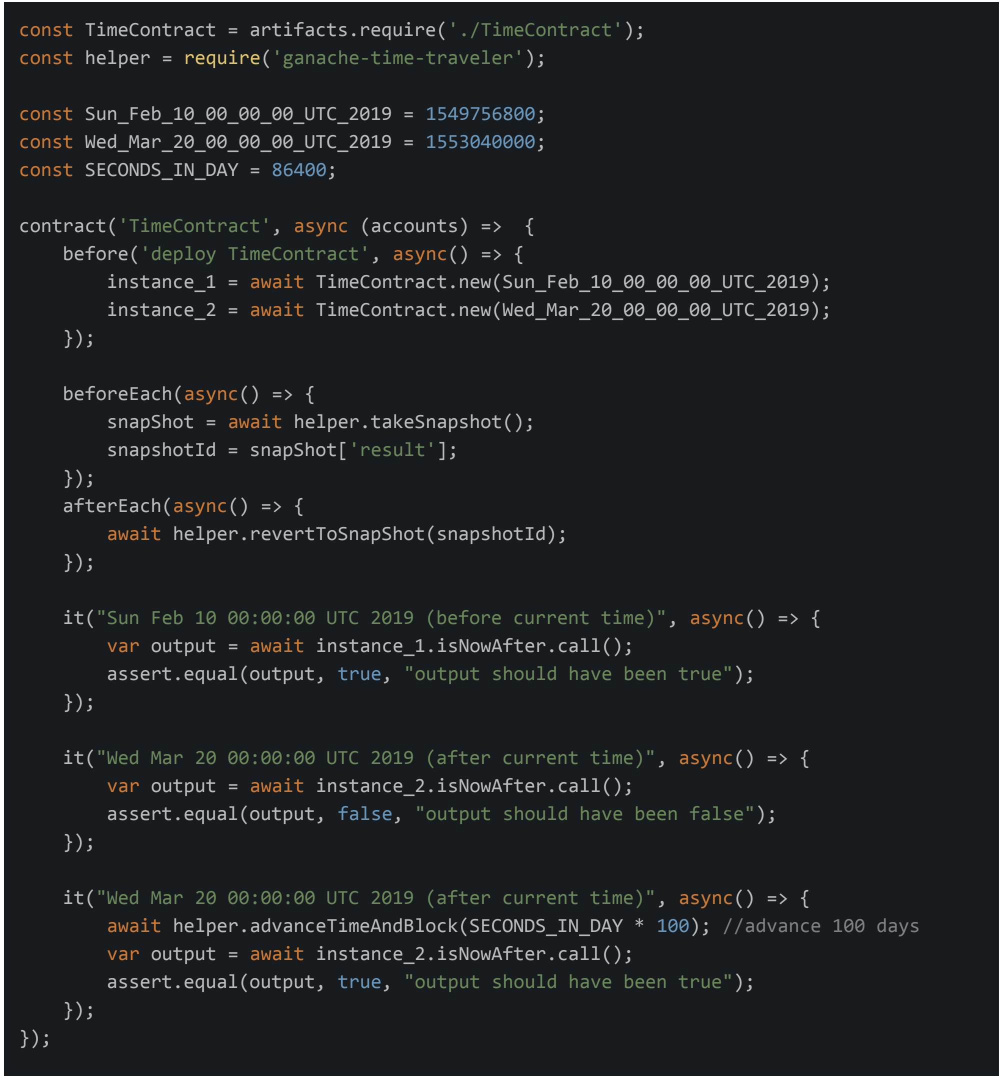
```
await helper.revertToSnapShot(snapshotId);
```

3. Adjust Time from within tests:

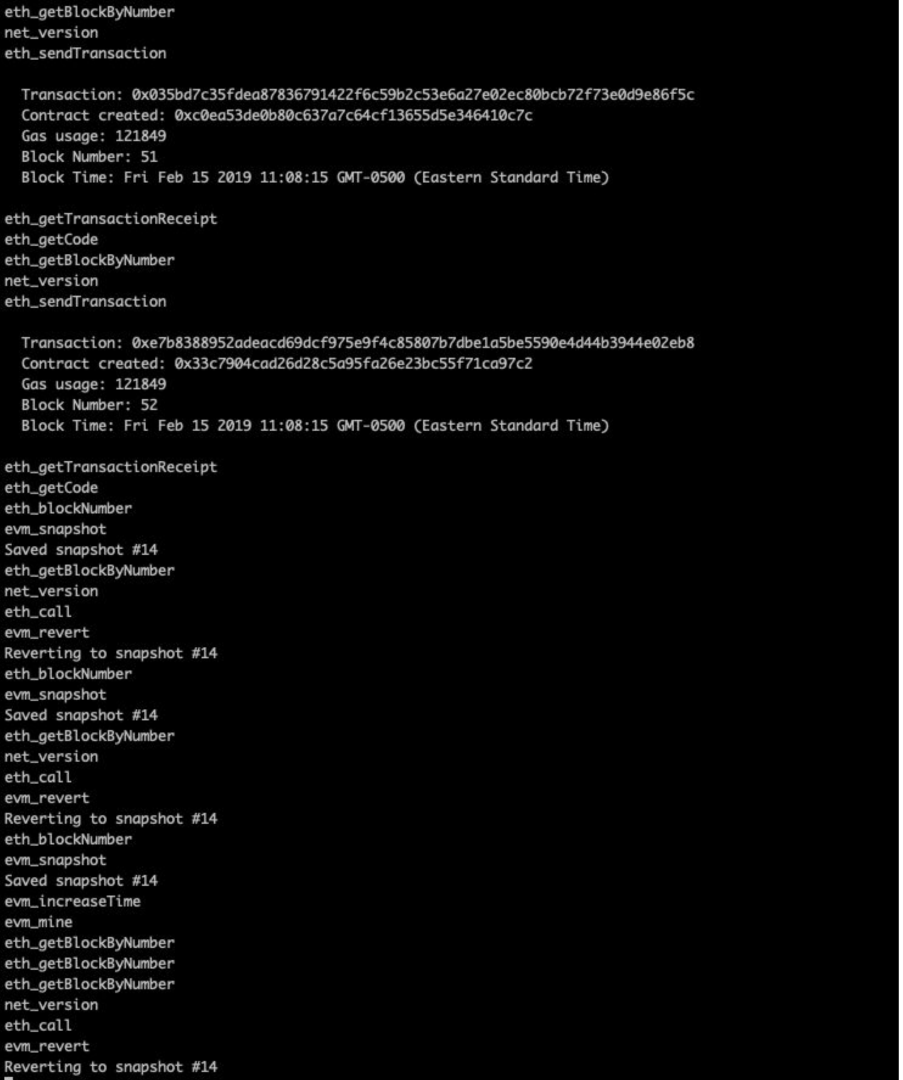
```
await helper.advanceTimeAndBlock(SECONDS_IN_DAY * 100);
```



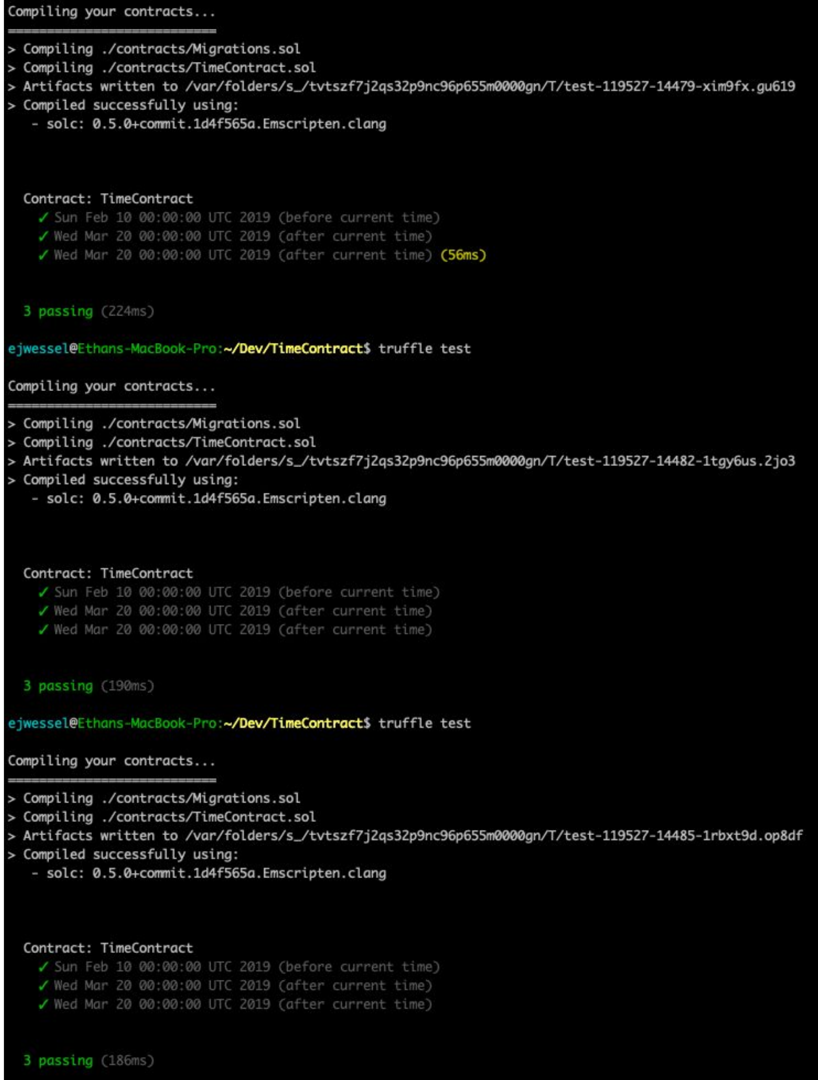
```
pragma solidity ^0.5.0;

contract TimeContract {
  uint256 private startTime;

  constructor(uint256 newStartTime) public {
    startTime = newStartTime;
  }

  /**
  * timeFunction will return true if now is after the given start time
  */
  function isNowAfter() external view returns (bool){
      return (now >= startTime);
  }
}z
```



```
const TimeContract = artifacts.require('./TimeContract');
const helper = require('ganache-time-traveler');

const Sun_Feb_10_00_00_00_UTC_2019 = 1549756800;
const Wed_Mar_20_00_00_00_UTC_2019 = 1553040000;
const SECONDS_IN_DAY = 86400;

contract('TimeContract', async (accounts) =>  {
    before('deploy TimeContract', async() => {
        instance_1 = await TimeContract.new(Sun_Feb_10_00_00_00_UTC_2019);
        instance_2 = await TimeContract.new(Wed_Mar_20_00_00_00_UTC_2019);
    });

    beforeEach(async() => {
        snapShot = await helper.takeSnapshot();
        snapshotId = snapShot['result'];
    });
    afterEach(async() => {
        await helper.revertToSnapShot(snapshotId);
    });

    it("Sun Feb 10 00:00:00 UTC 2019 (before current time)", async() => {
        var output = await instance_1.isNowAfter.call();
        assert.equal(output, true, "output should have been true");
    });

    it("Wed Mar 20 00:00:00 UTC 2019 (after current time)", async() => {
        var output = await instance_2.isNowAfter.call();
        assert.equal(output, false, "output should have been false");
    });

    it("Wed Mar 20 00:00:00 UTC 2019 (after current time)", async() => {
        await helper.advanceTimeAndBlock(SECONDS_IN_DAY * 100); //advance 100 days
        var output = await instance_2.isNowAfter.call();
        assert.equal(output, true, "output should have been true");
    });
});
```



```
eth_getBlockByNumber
net_version
eth_sendTransaction

  Transaction: 0x035bd7c35fdea87836791422f6c59b2c53e6a27e02ec80bcb72f73e0d9e86f5c
  Contract created: 0xc0ea53de0b80c637a7c64cf13655d5e346410c7c
  Gas usage: 121849
  Block Number: 51
  Block Time: Fri Feb 15 2019 11:08:15 GMT-0500 (Eastern Standard Time)

eth_getTransactionReceipt
eth_getCode
eth_getBlockByNumber
net_version
eth_sendTransaction

  Transaction: 0xe7b8388952adeacd69dcf975e9f4c85807b7dbe1a5be5590e4d44b3944e02eb8
  Contract created: 0x33c7904cad26d28c5a95fa26e23bc55f71ca97c2
  Gas usage: 121849
  Block Number: 52
  Block Time: Fri Feb 15 2019 11:08:15 GMT-0500 (Eastern Standard Time)

eth_getTransactionReceipt
eth_getCode
eth_blockNumber
evm_snapshot
Saved snapshot #14
eth_getBlockByNumber
net_version
eth_call
evm_revert
Reverting to snapshot #14
eth_blockNumber
evm_snapshot
Saved snapshot #14
eth_getBlockByNumber
net_version
eth_call
evm_revert
Reverting to snapshot #14
eth_blockNumber
evm_snapshot
Saved snapshot #14
evm_increaseTime
evm_mine
eth_getBlockByNumber
eth_getBlockByNumber
eth_getBlockByNumber
net_version
eth_call
evm_revert
Reverting to snapshot #14
```



```
Compiling your contracts...
===========================
> Compiling ./contracts/Migrations.sol
> Compiling ./contracts/TimeContract.sol
> Artifacts written to /var/folders/s_/tvtszf7j2qs32p9nc96p655m0000gn/T/test-119527-14479-xim9fx.gu619
> Compiled successfully using:
   - solc: 0.5.0+commit.1d4f565a.Emscripten.clang



  Contract: TimeContract
    ✓ Sun Feb 10 00:00:00 UTC 2019 (before current time)
    ✓ Wed Mar 20 00:00:00 UTC 2019 (after current time)
    ✓ Wed Mar 20 00:00:00 UTC 2019 (after current time) (56ms)


  3 passing (224ms)

ejwessel@Ethans-MacBook-Pro:~/Dev/TimeContract$ truffle test

Compiling your contracts...
===========================
> Compiling ./contracts/Migrations.sol
> Compiling ./contracts/TimeContract.sol
> Artifacts written to /var/folders/s_/tvtszf7j2qs32p9nc96p655m0000gn/T/test-119527-14482-1tgy6us.2jo3
> Compiled successfully using:
   - solc: 0.5.0+commit.1d4f565a.Emscripten.clang



  Contract: TimeContract
    ✓ Sun Feb 10 00:00:00 UTC 2019 (before current time)
    ✓ Wed Mar 20 00:00:00 UTC 2019 (after current time)
    ✓ Wed Mar 20 00:00:00 UTC 2019 (after current time)


  3 passing (190ms)

ejwessel@Ethans-MacBook-Pro:~/Dev/TimeContract$ truffle test

Compiling your contracts...
===========================
> Compiling ./contracts/Migrations.sol
> Compiling ./contracts/TimeContract.sol
> Artifacts written to /var/folders/s_/tvtszf7j2qs32p9nc96p655m0000gn/T/test-119527-14485-1rbxt9d.op8df
> Compiled successfully using:
   - solc: 0.5.0+commit.1d4f565a.Emscripten.clang



  Contract: TimeContract
    ✓ Sun Feb 10 00:00:00 UTC 2019 (before current time)
    ✓ Wed Mar 20 00:00:00 UTC 2019 (after current time)
    ✓ Wed Mar 20 00:00:00 UTC 2019 (after current time)


  3 passing (186ms)
```

Do you want to use it? (shameless plug)

- https://medium.com/fluidity/standing-the-time-of-test-b906fcc374a9
- https://github.com/ejwessel/GanacheTimeTraveler (code)
- https://github.com/ejwessel/TimeContract (example usage)

```
Add it to your package.json dependencies
     "ganache-time-traveler": "github:ejwessel/GanacheTimeTraveler"
Add to tests:
     const helper = require('ganache-time-traveler');
```

Versions

```
Truffle v5.0.5 (core: 5.0.5)
Solidity v0.5.0 (solc-js)
Node v11.10.0
Ganache CLI v6.3.0 (ganache-core: 2.4.0)
```

Resources

- Icon fast forward by Sidiq Fathurochman from the Noun Project
- https://medium.com/coinmonks/testing-time-dependent-logic-in-ethereum-smart-contracts-1b24845c7f72
- https://ethereum.stackexchange.com/questions/15596/how-can-i-mock-the-time-for-solidity-tests
- https://michalzalecki.com/ethereum-test-driven-introduction-to-solidity-part-2/
- https://github.com/iurimatias/EtherSim/blob/cb40a93374ca212eace6124310ef52d99bdcf68a/lib/manager.js

Thank you