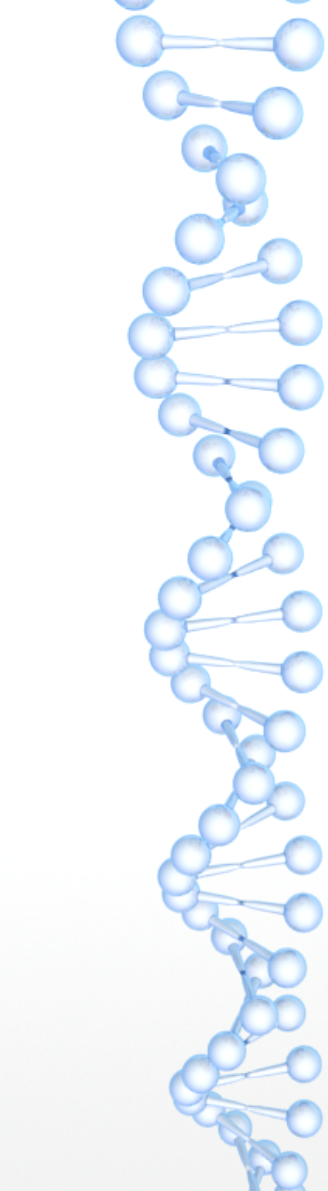


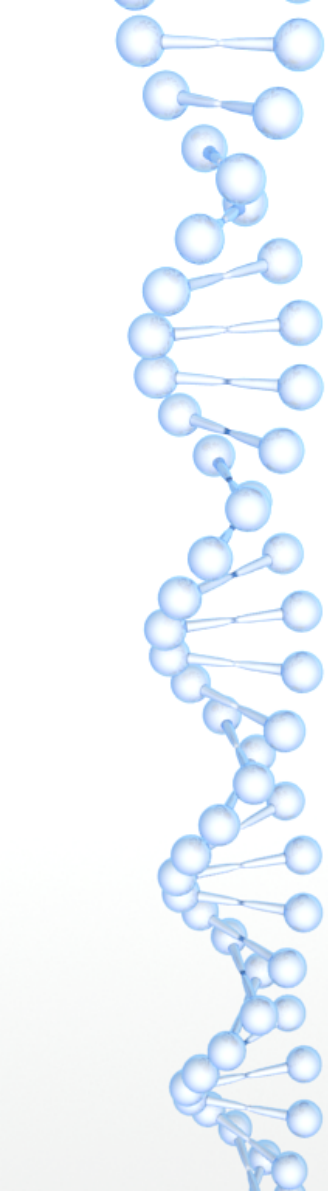
Running a Private Blockchain

Austin Hester
University of Missouri – St. Louis
01/28/18

Software Requirements

- Linux OS
- go-1.8.x+
- go-ethereum-1.7.3-stable
- Nodejs-1.8.x+
  - npm
    - truffle
    - testrpc
- Solidity compiler
- [opt] Mist browser
- [opt] VSCode w/ Solidity ext.

```
// add-apt-repository ppa:gophers/archive
// add-apt-repository ppa:ethereum/ethereum
```



go-ethereum (geth)

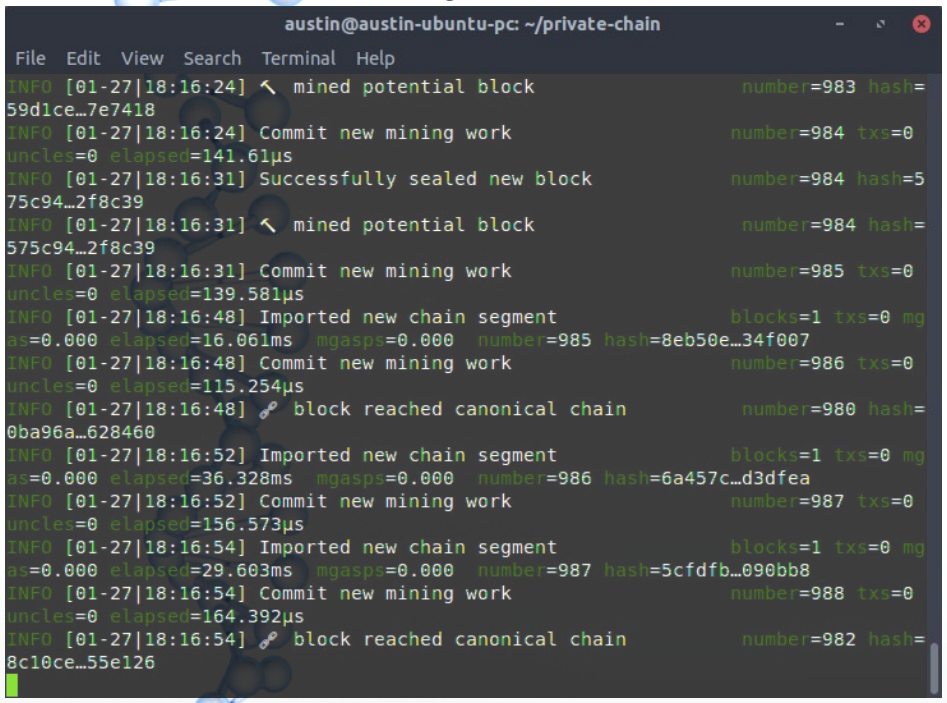
- What is it?
  - Geth is a command-line interface for running nodes on the Ethereum network.
- What is it used for?
  - Geth is widely used for development of **Decentralized applications** (Dapps).
  - Also used for deploying development / testing blockchains.
  - It’s a decent CPU miner.
- How can we use it?
  - Running a private blockchain with custom genesis block and testing Smart Contracts.

https://github.com/ethereum/go-ethereum

go-ethereum

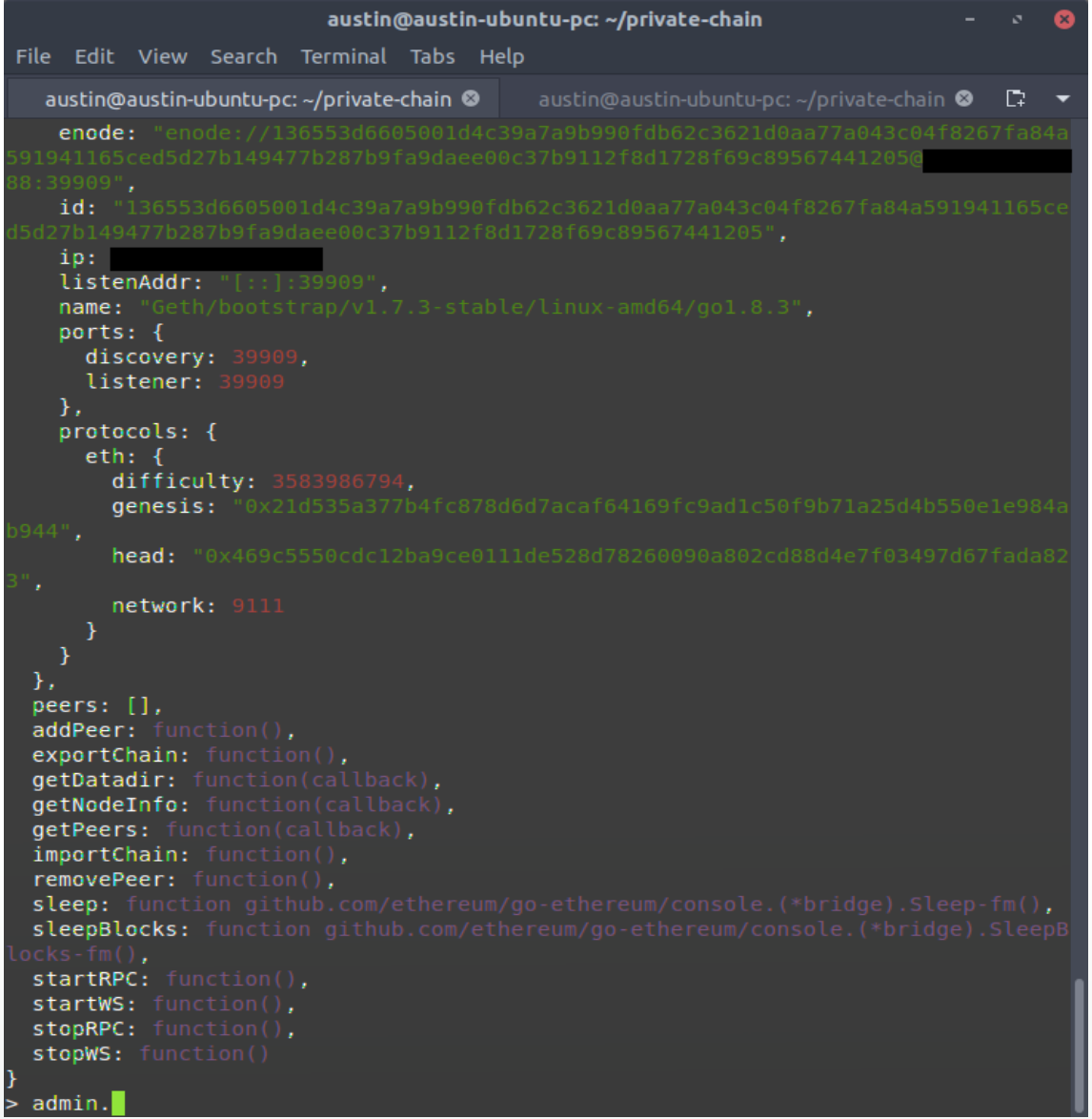
- What does it look like?

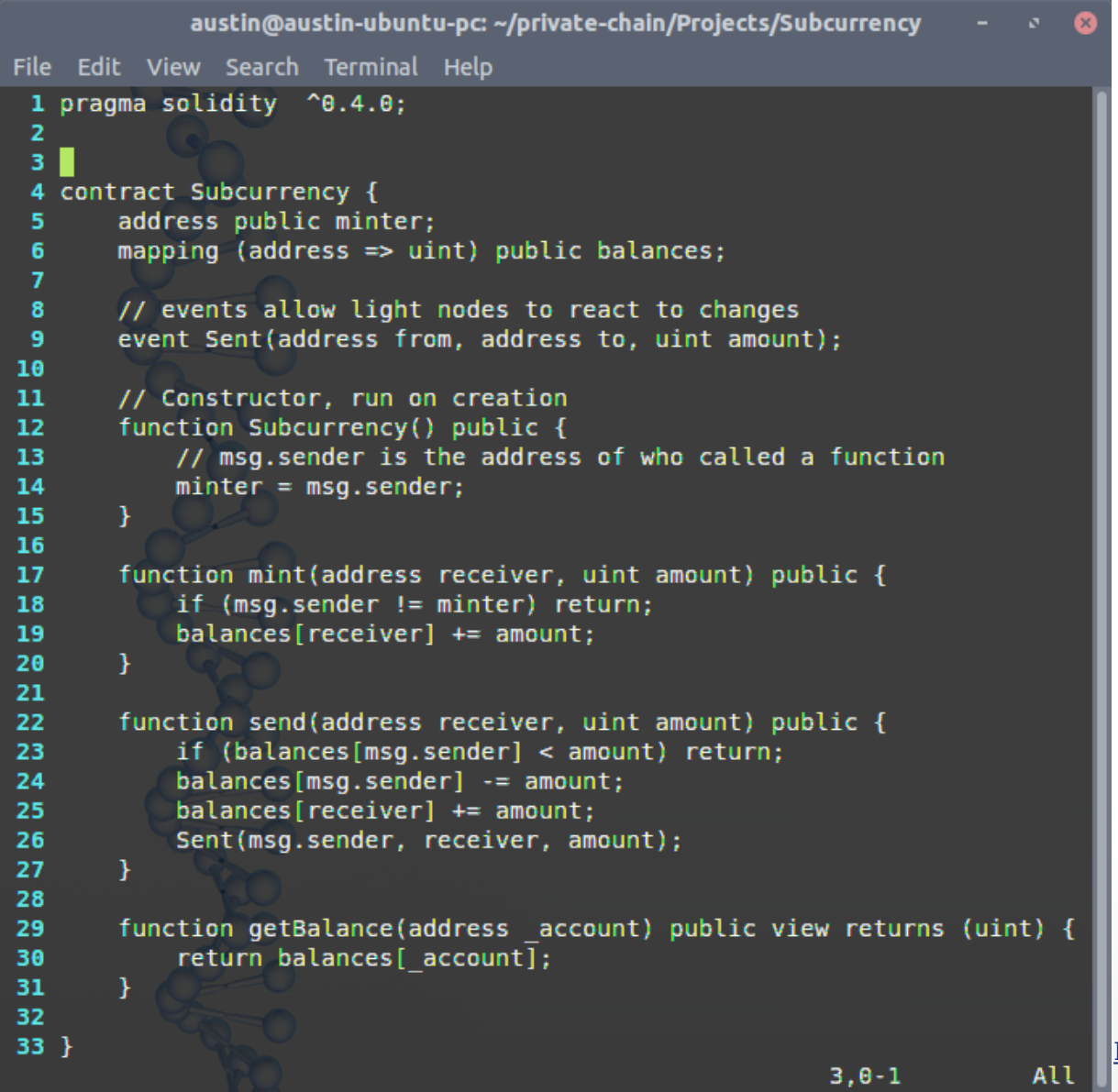
Mining



Administration

/go-ethereum/Management-API



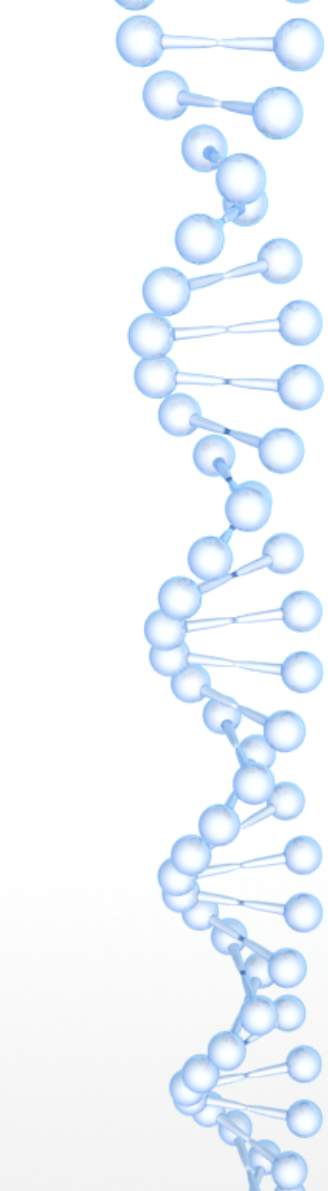


Solidity

> "Solidity is a contract-oriented, high-level language for implementing smart contracts. It was influenced by C++, Python and JavaScript and is designed to target the Ethereum Virtual Machine."

- Contracts exist in **contract accounts**, while user accounts are called **external accounts**.
- Contract accounts are controlled by **code**, while external accounts are controlled by public-private **key pairs**.
- Ex: Simple contract which maps a `balance` to an `address`.
  - Event `Sent` for watching contract.
  - Minter can `mint` tokens to a `receiver`
  - Users can `send` tokens to others.
  - You can also `getBalance` of an address.

https://solidity.readthedocs.io/en/develop/



Truffle

- What is it?
  - Truffle is a development environment for Ethereum contracts.
- What is it used for?
  - Smart contract compilation / testing
  - Contract deployment
  - Interactive console for using the contract
  - Install packages via Ethereum package manager (EthPM).
- Using `testrpc` we can deploy contracts on a temporary blockchain.

https://github.com/trufflesuite/truffle

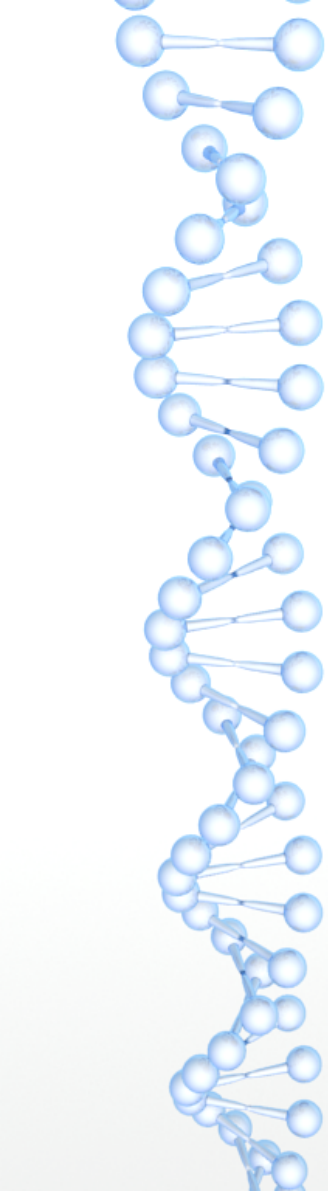
Truffle

- How can we use it?
  - `$ mkdir SmartToken && cd SmartToken`
  - `$ truffle init` *// initialize*
  - `$ truffle compile`
  - `$ truffle migrate --reset` *// deploy contract on test network*
  - `$ truffle console` *// start the console*

  - `truffle> SmartToken.address`
  - `truffle> JSON.stringify(SmartToken.abi)`

  - The above two lines retrieve the information needed to interact with your contract on the Mist browser.

https://github.com/trufflesuite/truffle



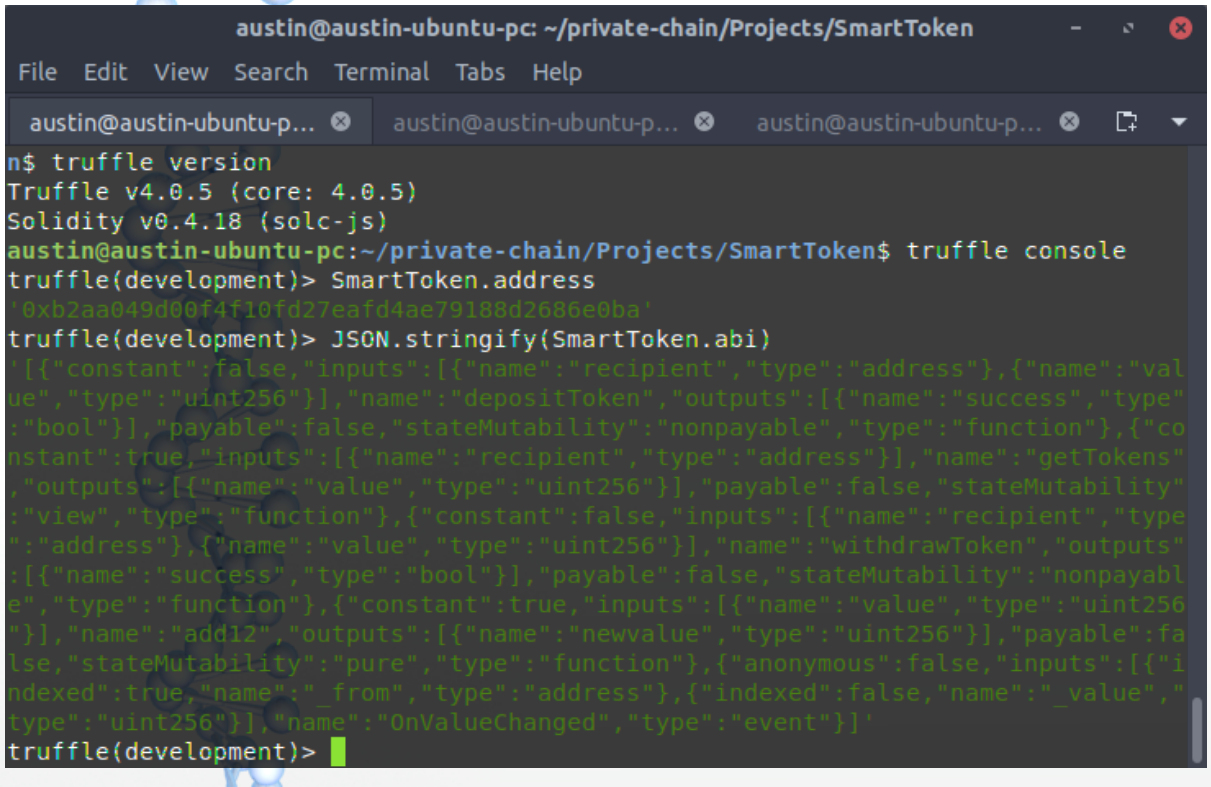
Mist Browser

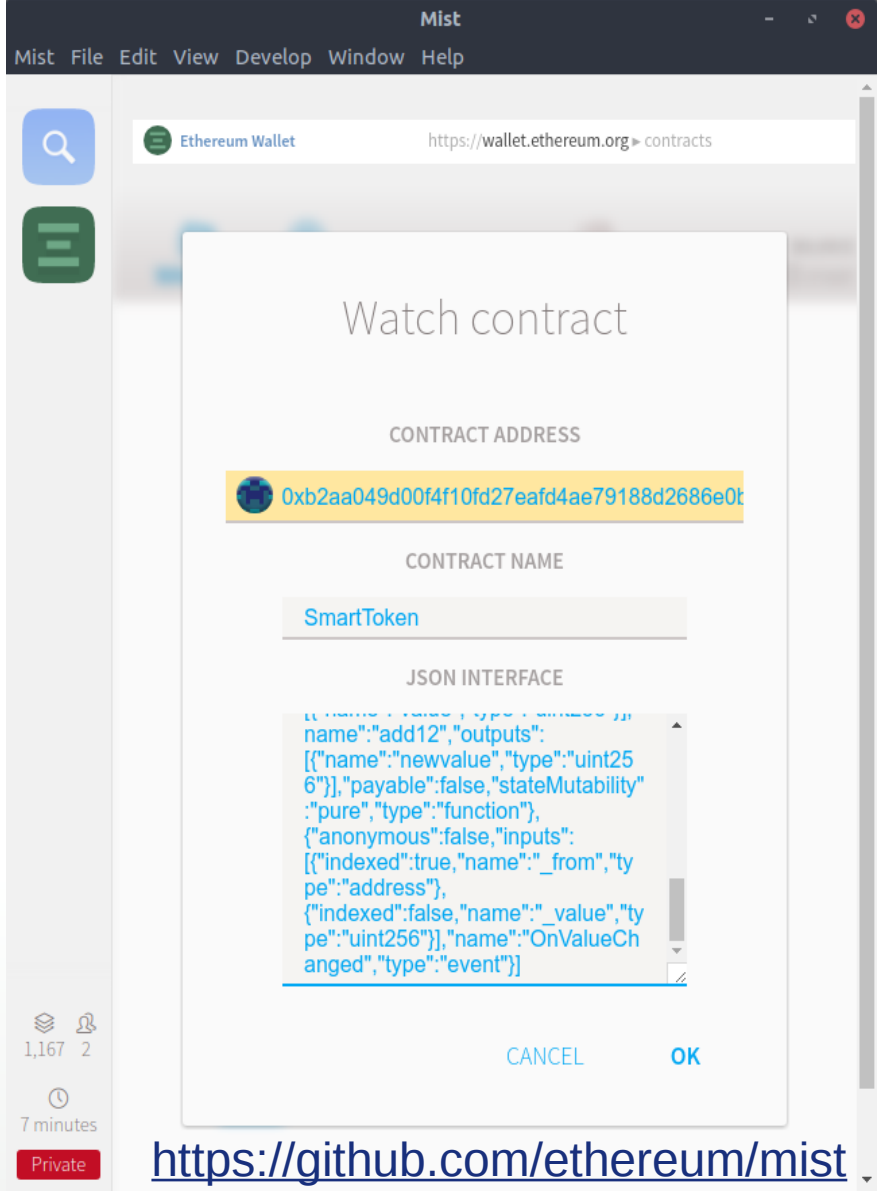
- Mist browser is a GUI for monitoring and manipulating your private blockchain.
- You can link to a node by opening Mist using its rpc port.
- Mist browser doubles as an Ethereum wallet.
- You can:
  - Send / receive transactions.
  - Watch smart contract activity.

https://github.com/ethereum/mist

Mist Browser

You can watch a contract for events by giving it the `address` and `JSON interface` of the contract.

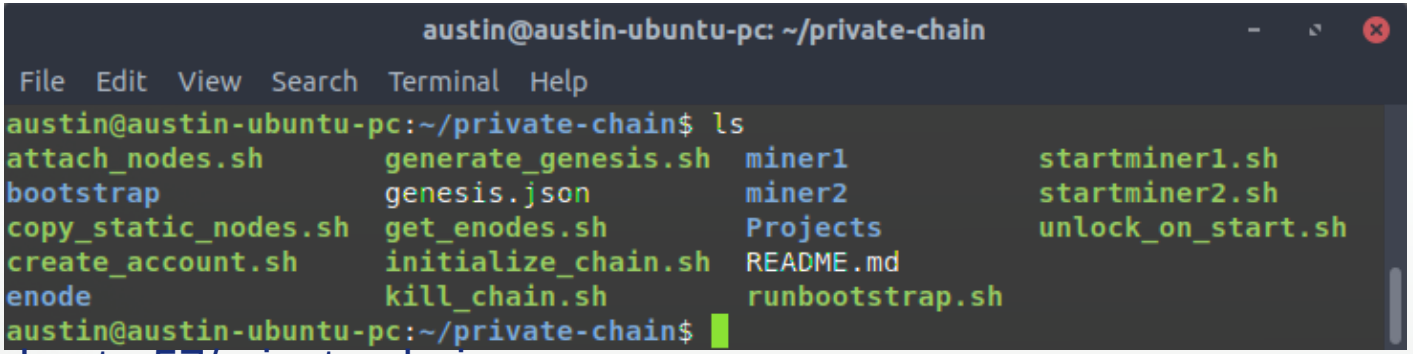




https://github.com/ethereum/mist

My Scripts for geth

- **`generate_genesis.sh`** *// generates `genesis.json` for initialization (recommend changes)*
- **`initialize_chain.sh`** *// initialize blockchain with generated genesis block*
- **`create_account.sh`** *// create 2 accounts for each node*
- **`unlock_on_start.sh`** *// create `passwd.sec` in datadirs*
- **`get_enodes.sh`** & **`copy_static_nodes.sh`** *// retrieve `enodes`, required for syncing nodes*
- **`runbootstrap.sh`** & **`startminer[1,2].sh`** *// for starting each*
- **`attach_nodes.sh`** *// attach to node and open console*
- **`kill_chain.sh`** *// kill all instances of `geth`*



https://github.com/ahester57/private-chain

runbootstrap.sh

austin@austin-ubuntu-pc: ~/private-chain

File Edit View Search Terminal Help

```
 1 # !/bin/bash
 2
 3 CURR=`dirname $0`
 4 if [ -d $CURR/bootstrap/geth ]; then
 5 geth --identity "bootstrap" --networkid 9111 \
 6       --datadir $CURR/bootstrap  \
 7       --ipcpath /home/$USER/.ethereum/geth.ipc \
 8       --nodiscover --port "39909" --rpc --rpcport "42024" \
 9       --rpcapi "db,eth,web3,net,personal,miner" \
10       --unlock 0 --password $CURR/bootstrap/passwd.sec
11 else
12     echo "Initialize first"
13 fi
~
"runbootstrap.sh" 13L, 394C                                1,1           All
```

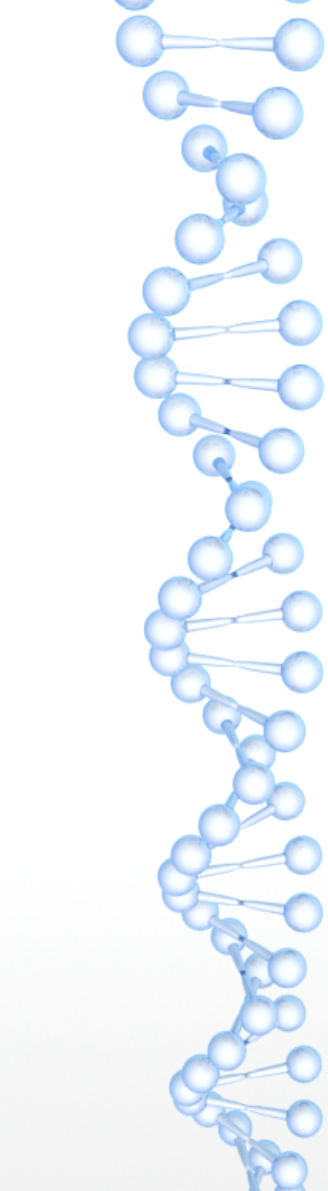
This script starts the `bootstrap` node. Avoids typing all this every time.

Raspberry Pi as Node

- I was successfully able to use a Raspberry Pi 3 as a light node, which syncs with the miners running on my PC.
- On the Pi:
  - Clone the repo: https://github.com/ahester57/private-chain.git
  - `$ sftp <to PC>; > get genesis.json`
  - `$ geth --datadir ./pinode0 init genesis.json`
  - Create accounts and run to get its "`enode://...`" address
  - Save "`enode://...`" to `<file>`
  - `$ sftp <to PC>; > put <file>`
  - Restart the Pi node, it will now wait until synced with our blockchain.

Raspberry Pi as Node

- On PC:
  - `./attach_nodes.sh` *// choose 'b'*
  - `> admin.addPeer("enode://..@<pi_ip>:<pi_port>");`
    - Should return `true`
- Wait a short while, and check it is connected using
  - `> admin.peers` while attached to `bootstrap` node.
  - The `geth` instance running on the Pi should start importing **chain segments** once connected to our blockchain.
  - We can add its `enode` address to `static-nodes.json` to do this automatically on restart.



Resources

- https://github.com/ethereum/wiki/wiki
- https://github.com/ethereum/go-ethereum
- https://github.com/ahester57/private-chain
- https://ethereum.org/token
- https://solidity.readthedocs.io/en/develop/
- http://truffleframework.com/docs/getting_started/project
- https://github.com/ethereum/mist/wiki