Write and deploy my first dApp part.1

Deploy a local blockchain based on the Ethereum Virtual Machine

Prerequisites

Node.js

Installation

Ganache-cli (new testrpc) simulates a full node behaviour.

```
npm install -g ganache-cli
```

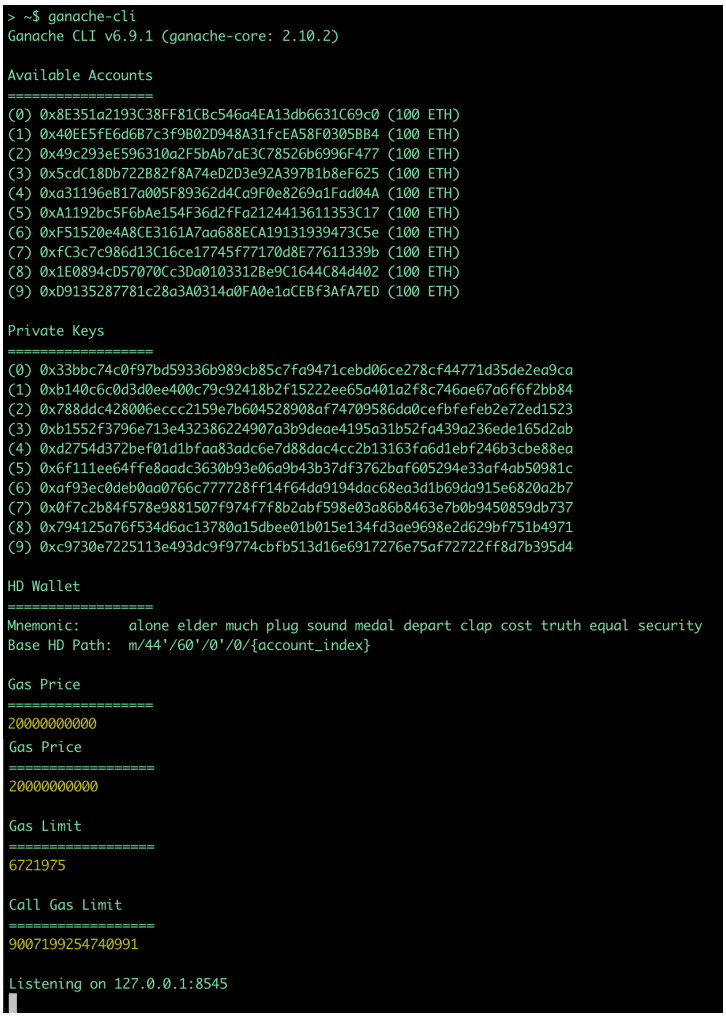
Web3 is a collection of libraries to interact with a local or remote Ethereum node.

```
npm install web3
```

First run

By starting ganache-cli without any option, 10 accounts are created with 100ETH each.

ganache-cli listens on the local port 8585

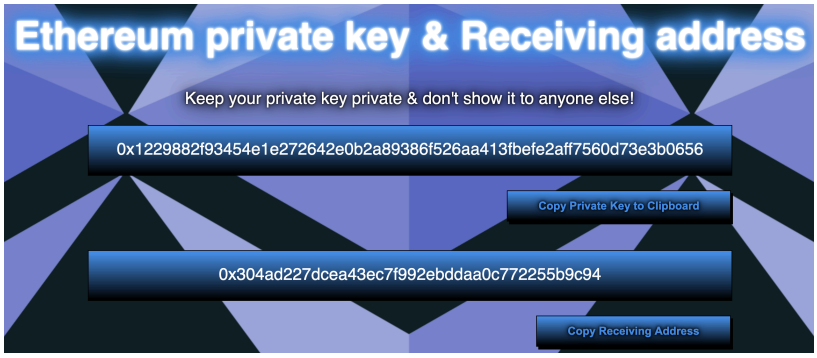


Account generation

Create two pairs of keys.

https://www.myetherwallet.com/

https://generatepaperwallet.com/ethereum/



Second run with two accounts

Run ganache-cli with the account option followed by a private key.

The second number is the amount of ETH in this account, expressed in WEI where 1ETH=1000000000000000000 WEI (« 1 » and 18 « 0 »).

```
ganache-cli
--account=
"0xbe4953a541f006c4bb2297a6cbf5f1fd1c38ce158d6c11c68bd1135e1150016
1,9000000000000000000"
--account=
« 0xaaeee2a36f453b0efb61b28f85a760ad4dbfa6fb61d4f6a8c126c7f11f7807
14,90000000000000000000"
```

```
> ~$ ganache-cli --account="0xbe4953a541f006c4bb2297a6cbf5f1fd1c38ce158d6c11c68bd1135
e11500161,9000000000000000000" --account="0xaaeee2a36f453b0efb61b28f85a760ad4dbfa6fb6
1d4f6a8c126c7f11f780714,90000000000000000000"
Ganache CLI v6.9.1 (ganache-core: 2.10.2)

Available Accounts
==================
(0) 0x43c97D3063d4ecAf6139EbAE1ac6c2F332e835a5 (9 ETH)
(1) 0xF8cC988b81308F6F2d21b18B10772bc5fCE30EEf (9000 ETH)

Private Keys
==================
(0) 0xbe4953a541f006c4bb2297a6cbf5f1fd1c38ce158d6c11c68bd1135e11500161
(1) 0xaaeee2a36f453b0efb61b28f85a760ad4dbfa6fb61d4f6a8c126c7f11f780714

Gas Price
==================
20000000000

Gas Limit
==================
6721975

Call Gas Limit
==================
9007199254740991

Listening on 127.0.0.1:8545
```

Interactions with ganache-cli

Keep the terminal with ganache-cli running open.

Open a second terminal and run node. Initialise web3 with the following
`var Web3=require('web3');`
`var web3= new Web3('http://localhost:8545');`

And display the accounts
`web3.eth.getAccounts(console.log);`

```
> ~$ node
Welcome to Node.js v12.12.0.
Type ".help" for more information.
> var Web3=require('web3');
undefined
> var web3= new Web3('http://localhost:8545');
undefined
> web3.eth.getAccounts(console.log);
Promise { <pending> }
> null [
  '0x43c97D3063d4ecAf6139EbAE1ac6c2F332e835a5',
  '0xF8cC988b81308F6F2d21b18B10772bc5fCE30EEf'
]
```

While displaying the accounts with getAccounts, in the other terminal you should see the following

```
Call Gas Limit
==================
9007199254740991

Listening on 127.0.0.1:8545
eth_accounts
```

Get the balance of an account
`web3.eth.getBalance("0x00ae1858ea41f5667cda17c7915c2f289c4ee819").then(console.log);`

```
> web3.eth.getBalance("0xF8cC988b81308F6F2d21b18B10772bc5fCE30EEf").then(console.log);
Promise { <pending> }
> 90000000000000000000
```

In the other terminal you should see the following

```
Call Gas Limit
==================
9007199254740991

Listening on 127.0.0.1:8545
eth_accounts
eth_getBalance
```

We create a transaction from one account to the other of 10000000000000000000 WEI with
`web3.eth.sendTransaction({from : '0xF8cC988b81308F6F2d21b18B10772bc5fCE30EEf', to :'0x43c97d3063d4ecaf6139ebae1ac6c2f332e835a5', value:'10000000000000000000'});`

```
> web3.eth.sendTransaction({from : '0xF8cC988b81308F6F2d21b18B10772bc5fCE30EEf',
to :'0x43c97d3063d4ecaf6139ebae1ac6c2f332e835a5', value:'10000000000000000000'});
Promise {
  <pending>,
  _events: Events <[Object: null prototype] {}> {},
  emit: [Function: emit],
  on: [Function: on],
  once: [Function: once],
  off: [Function: removeListener],
  listeners: [Function: listeners],
  addListener: [Function: on],
  removeListener: [Function: removeListener],
  removeAllListeners: [Function: removeAllListeners]
}
>
```

And we check at which block we are with
`web3.eth.getBlockNumber().then(console.log);`

```
> web3.eth.getBlockNumber().then(console.log);
Promise { <pending> }
> 1
```

Meanwhile, in the other terminal

```
eth_sendTransaction

  Transaction: 0x05a1ae391a7c6400b2696beeb7e477e
  Gas usage: 21000
  Block Number: 1
  Block Time: Mon Jan 01 1970 00:00:01 GMT+0000

eth_getTransactionReceipt
eth_blockNumber
```

We can check the new amount in our account

```
> web3.eth.getBalance("0xF8cC988b81308F6F2d21b18B10772bc5fCE30EEf").then(console.log);
Promise { <pending> }
> 898999958000000000000
```

It is easier for the different tests to put all the commands in a file and run it with
`node file.js`

Try different tests, for example:
— empty an account with a while loop;
— is there is a delay to respect between two transactions authored by a single account;
— with a several accounts, can we slow down ganache by performing many transactions simultaneously.

Truffle

Download the truffle environment to deploy smart-contracts written in Solidity.
Ganache-cli must be running besides.
`npm install -g truffle`

Create an empty directory and run
`truffle init`

Truffle creates a project, with configuration files and folders for the different contracts written in Solidity.

```
> truffle-smartcontract$ ls
build               migrations          truffle-config.js
contracts           test                truffle.js
> truffle-smartcontract$ ls contracts
Migrations.sol Token.sol
```

Make sure to use the last Solidity version in your .sol files, with the header
`pragma solidity ^0.5.8;`

The commands

`truffle compile`
compiles your code

`truffle migrate`
deploys the contracts to the local blockchain.