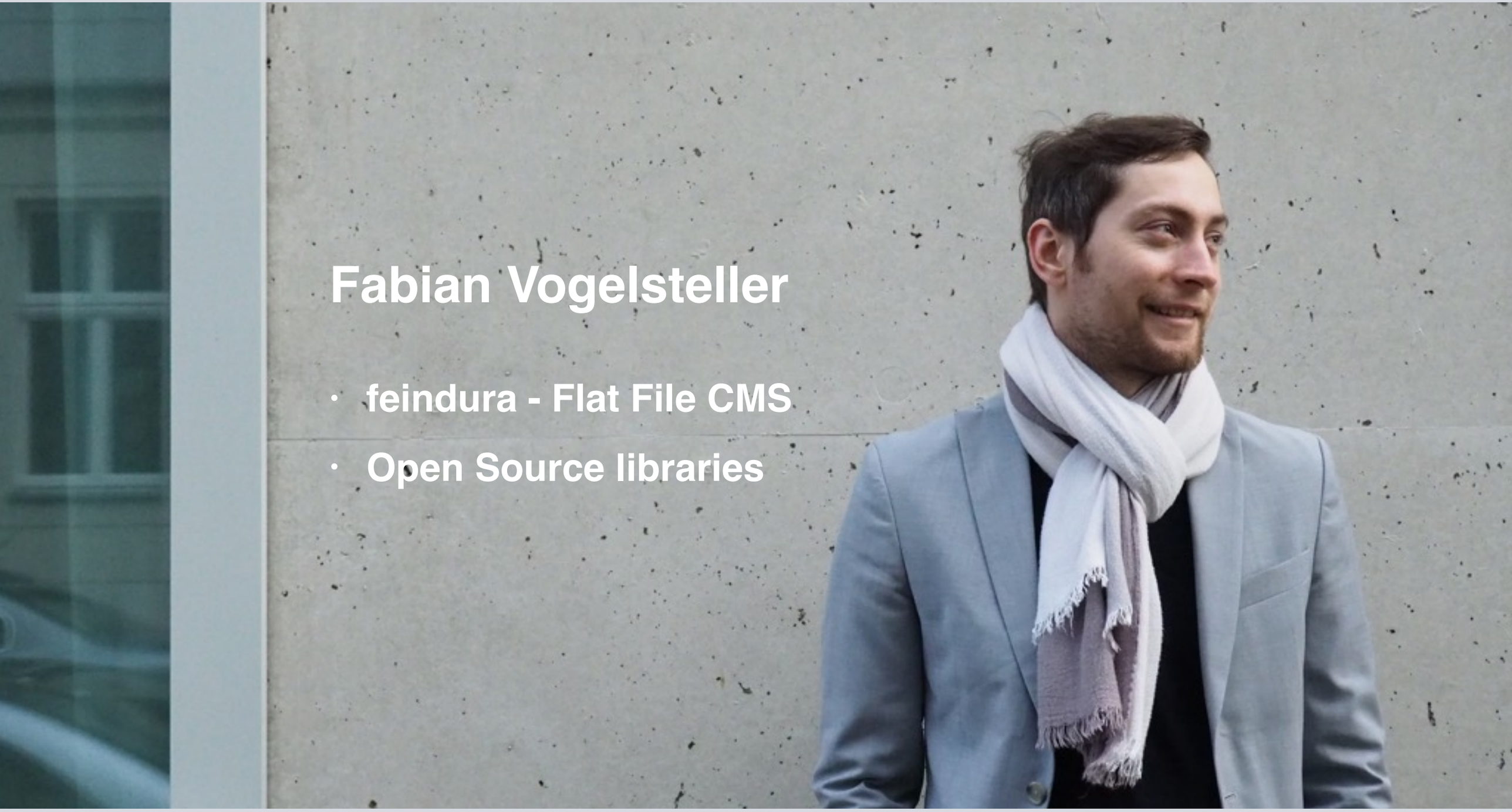
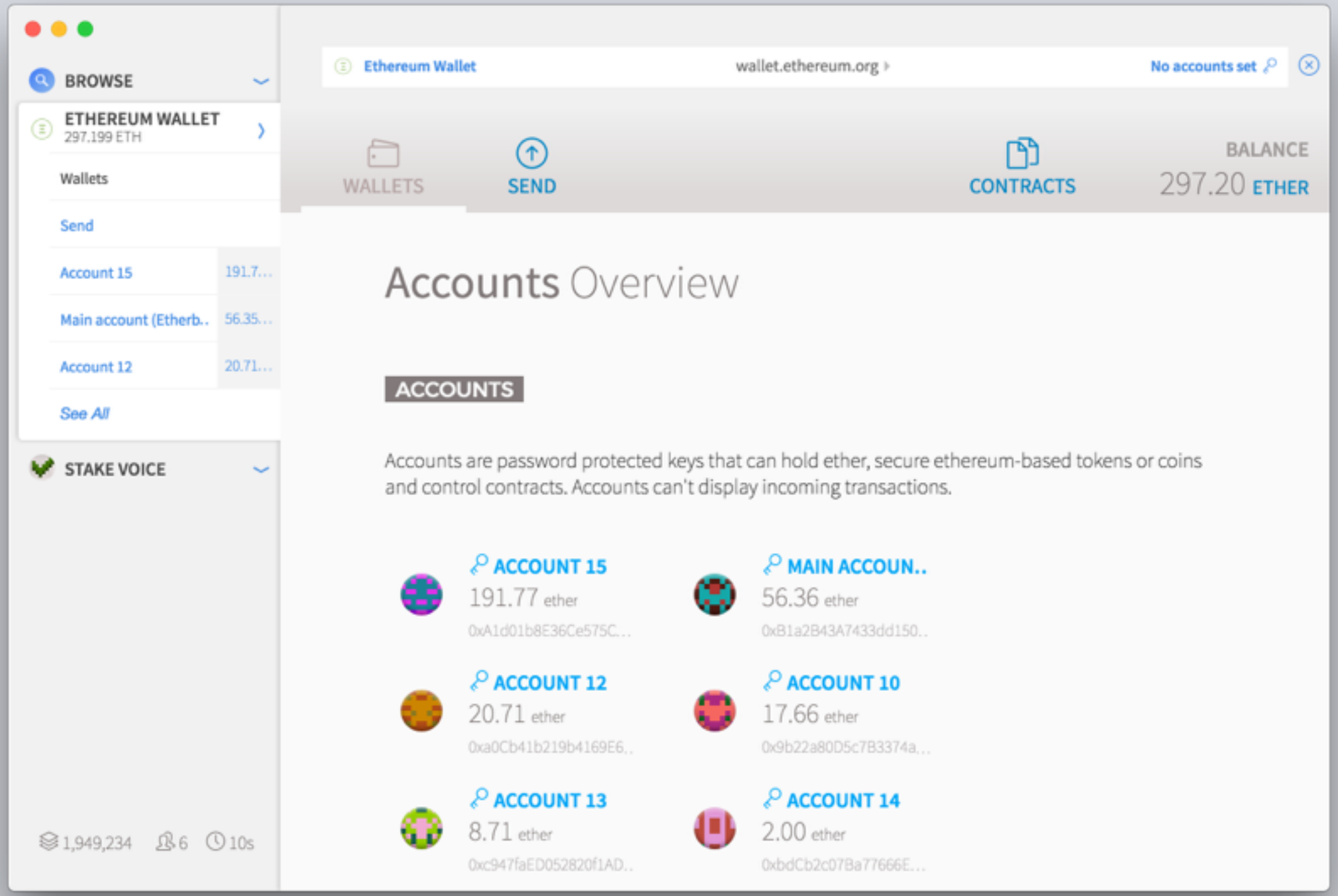
web3.js

Fabian Vogelsteller, web3.js and Mist Developer Ethereum
@feindura
http://frozeman.de/blog



Fabian Vogelsteller, web3.js and Mist Developer Ethereum
@feindura
http://frozeman.de/blog

Fabian Vogelsteller, web3.js and Mist Developer Ethereum
@feindura
http://frozeman.de/blog



Fabian Vogelsteller, web3.js and Mist Developer Ethereum
@feindura
http://frozeman.de/blog

Ethereum JavaScript API

gitter join chat

This is the Ethereum compatible JavaScript API which implements the Generic JSON RPC spec. It's available on npm as a node module, for bower and component as an embeddable js and as a meteor.js package.

npm package 0.17.0-alpha | build error | dependencies up to date | devDependencies up to date | coverage 91% | Ready 0

You need to run a local Ethereum node to use this library.

Documentation

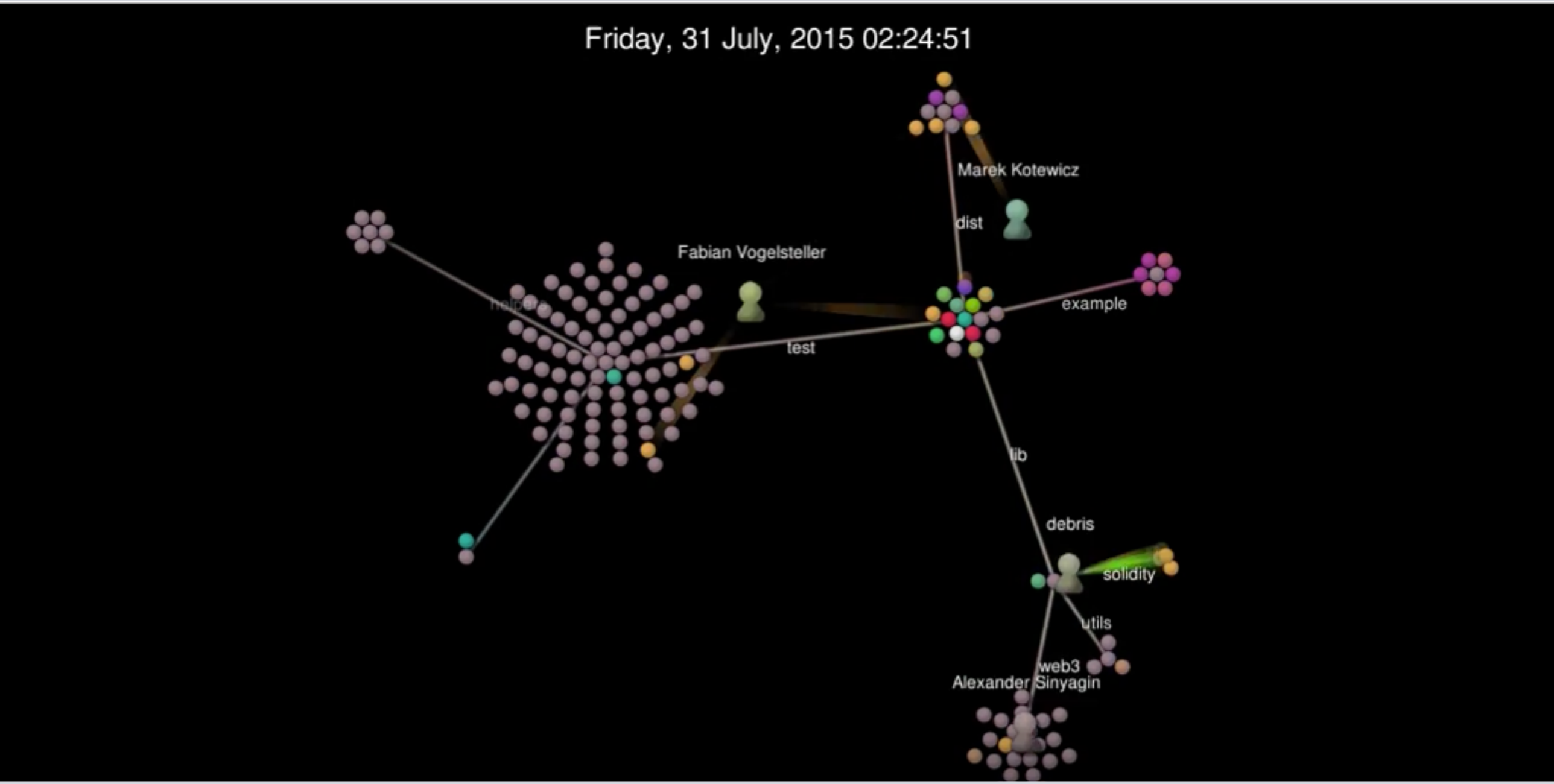
Installation

Node.js

```
npm install web3
```

Meteor.js

```
meteor add ethereum:web3
```

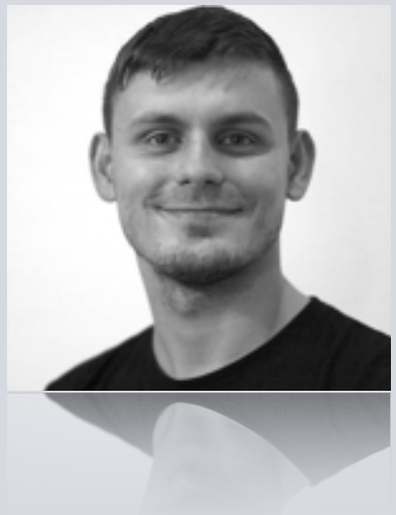


Fabian Vogelsteller, web3.js and Mist Developer Ethereum
@feindura
http://frozeman.de/blog

Jeffrey Wilcke

Fabian Vogelsteller, web3.js and Mist Developer Ethereum
@feindura
http://frozeman.de/blog

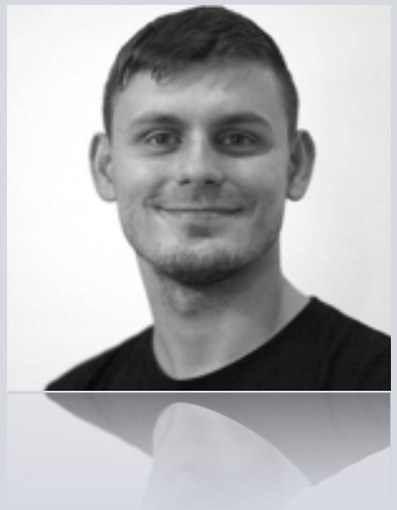
Jeffrey Wilcke



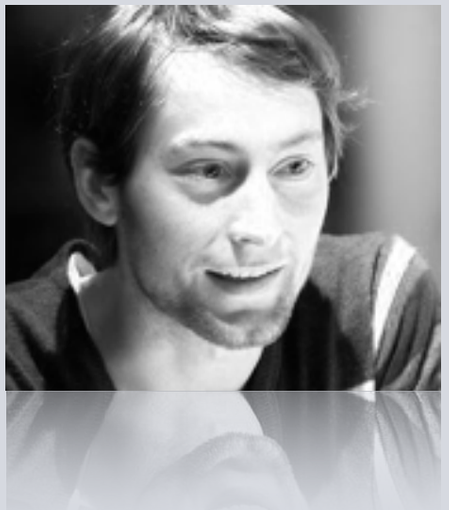
Marek Kotewicz

Fabian Vogelsteller, web3.js and Mist Developer Ethereum
@feindura
http://frozeman.de/blog

Jeffrey Wilcke



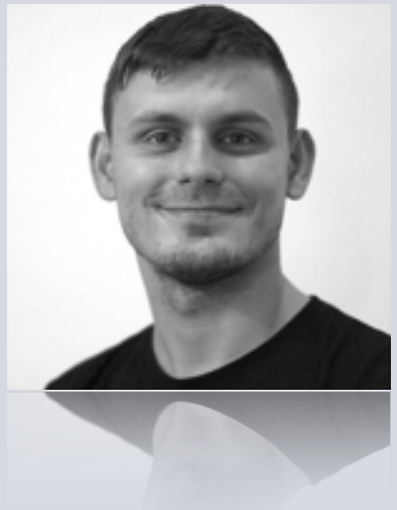
Marek Kotewicz



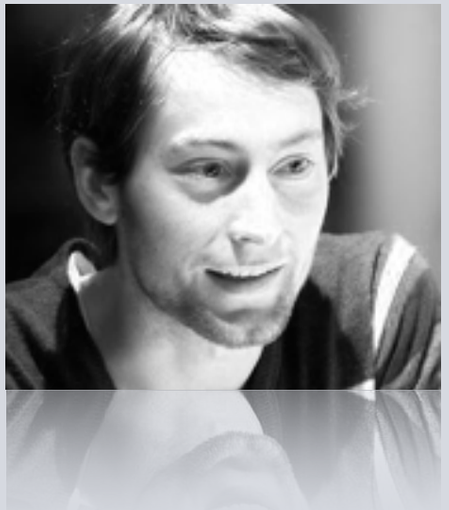
Fabian Vogelsteller

Fabian Vogelsteller, web3.js and Mist Developer Ethereum
@feindura
http://frozeman.de/blog

Jeffrey Wilcke



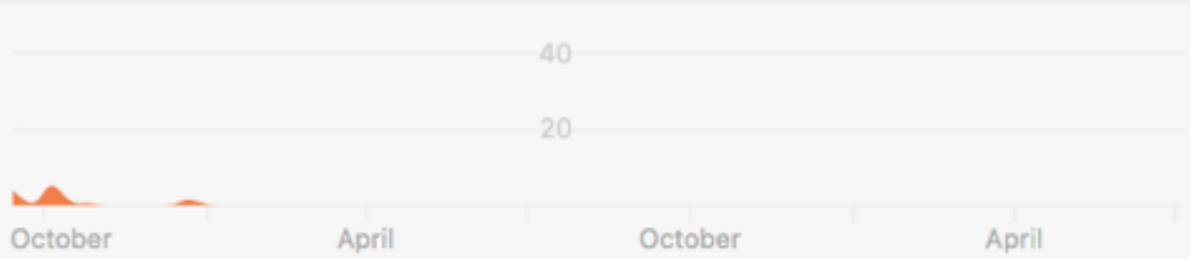
Marek Kotewicz



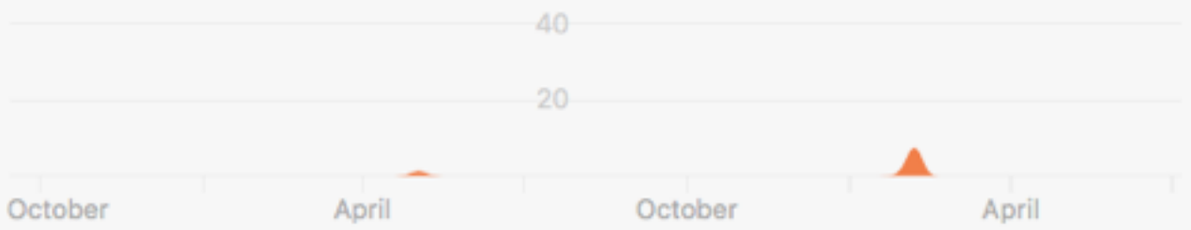
Fabian Vogelsteller

cubedro #3
28 commits / 3,089 ++ / 2,624 --

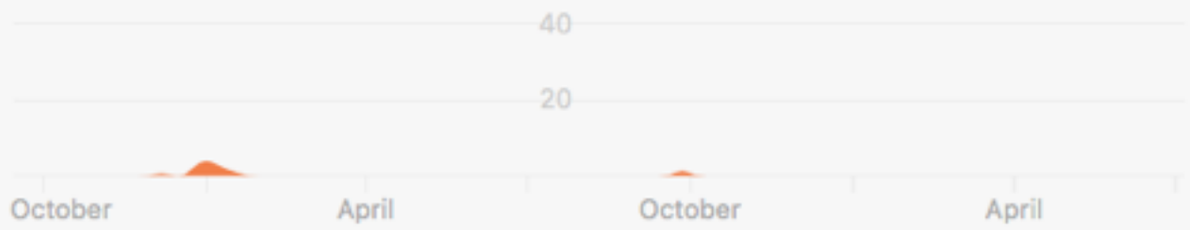
obscuren #4
19 commits / 1,520 ++ / 872 --



alexvandesande #5
14 commits / 226 ++ / 108 --



gavofyork #6
14 commits / 34,788 ++ / 179 --



JSON RPC 2.0

```
// Request
curl -X POST --data '{"jsonrpc":"2.0","method":"eth_accounts","params":[],"id":1}'

// Result
{
  "id":1,
  "jsonrpc": "2.0",
  "result": ["0x407d73d8a49eeb85d32cf465507dd71d507100c1"]
}
```

```
web3.eth.accounts
> ["0x407d73d8a49eeb85d32cf465507dd71d507100c1"]
```

Fabian Vogelsteller, web3.js and Mist Developer Ethereum
@feindura
http://frozeman.de/blog

Smart Contracts are EVM opcodes

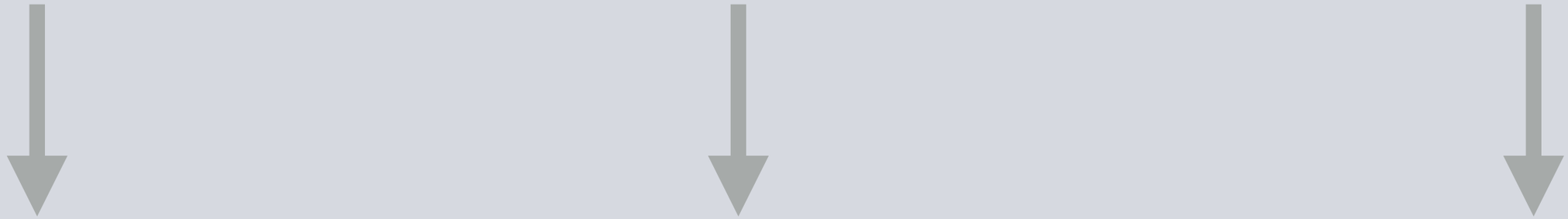
606060405260405160c0806132c883396101206040529051608051 60a051925160e0516101005193949293828282600f829055601083905560118054610100830261010060
a860020a0319909116179055604051309060019061 01bc8061033e8339600160a060020a03909316908301526101408201526040519081900361016001906000f060128054
600160a060020a031916919091179055505060038054600160a060020a0319908116891790915560 0e805490911687179055 50600c84905560405130906000906101bc8061
04fa8339018083600160a060020a0316815260200182815260200192505050604051809103906000f0600760006101000a815481600160a060020a0302191690830217905 5
50306000604051 6101bc806106b6833901808360 0160a060020a031681526020018281526020019250505060405180910390 6000f060088054600160a060020a0319169190
911790556007546001 60a060020a03166000141561017555761 0002565b600854600160a060020a03166000141561018e57610002565b426002556005600190815560008054
828255829080158290116101ca57600e0281600e028360005260206000209182019101610 1ca9190610245565b50505030600160a060020a03908116600090815260046020
526040808220805460ff19908116600190811790925560035490941683529120805490921617905550505050505050612a568061087260003960 00f35b5050600060098201
819055600a820155600d81018054600160a060020a031916905560010 15b8082111561033a578054600160a060020a03191681556000600182810182905560028381018054
848255909281161561010002600019011604601f81901061030c57505b506000600383018190556004838101805461ffff1916905560058401829055600684018290556007

Fabian Vogelsteller, web3.js and Mist Developer Ethereum
@feindura
http://frozeman.de/blog

ABI encoding

```
transfer(address,uint256)
```

```
transfer(0x692a70d2e424a56d2c6c27aa97d1a86395877b3a, 2000)
```



Fabian Vogelsteller, web3.js and Mist Developer Ethereum
@feindura
http://frozeman.de/blog

Other useful functions

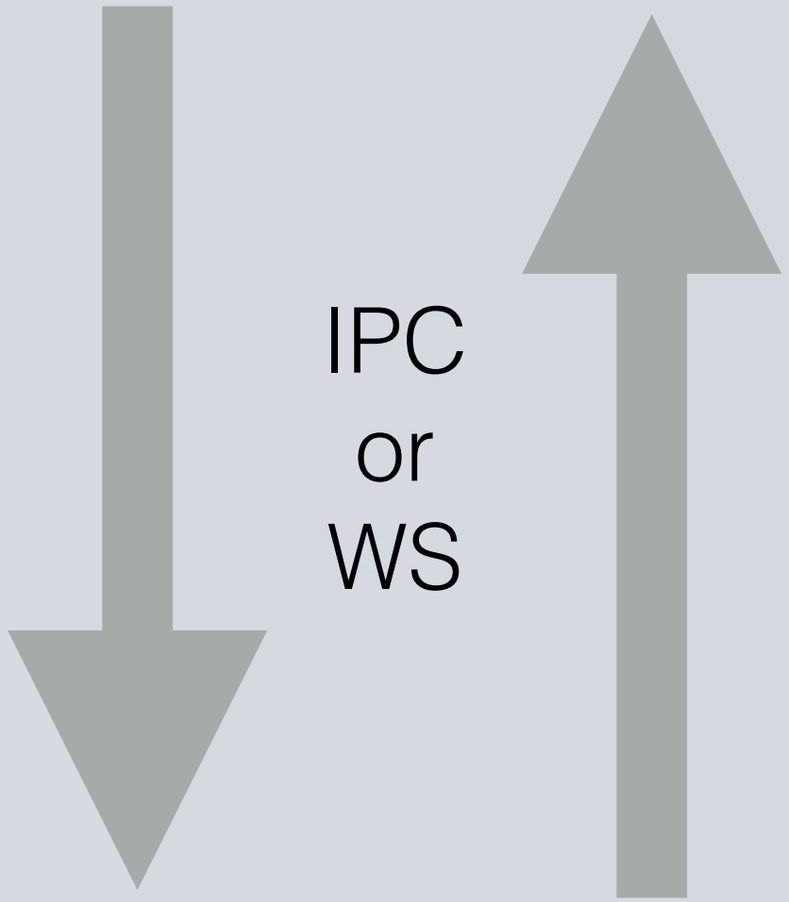
```
▶ createBatch: function ()
▶ fromAscii: function (t)
▶ fromDecimal: function (t)
▶ fromICAP: function (t)
▶ fromUtf8: function (t)
▶ fromWei: function (t,e)
▶ isAddress: function (t)
▶ isChecksumAddress: function (t)
▶ isConnected: function ()
  isIBAN: undefined
▶ reset: function (t)
▶ setProvider: function (t)
▶ sha3: function (t,e)
▶ toAscii: function (t)
▶ toBigNumber: function (t)
▶ toChecksumAddress: function (t)
▶ toDecimal: function (t)
▶ toHex: function (t)
▶ toUtf8: function (t)
▶ toWei: function (t,e)
```

Fabian Vogelsteller, web3.js and Mist Developer Ethereum
@feindura
http://frozeman.de/blog

The new web3.js

Fabian Vogelsteller, web3.js and Mist Developer Ethereum
@feindura
http://frozeman.de/blog

Socket based



Fabian Vogelsteller, web3.js and Mist Developer Ethereum
@feindura
http://frozeman.de/blog

The **old** filter

```
// OR object are log filter options
var filter = web3.eth.filter(options);

// watch for changes
filter.watch(function(error, result){
  if (!error)
    console.log(result);
});
```

Fabian Vogelsteller, web3.js and Mist Developer Ethereum
@feindura
http://frozeman.de/blog

The **old** filter

```
// OR object store log filter options
var filter = web3.eth.filter(options);

// watch for changes
filter.watch(function(error, result){
  if (!error)
    console.log(result);
});
```

Fabian Vogelsteller, web3.js and Mist Developer Ethereum
@feindura
http://frozeman.de/blog

The **new** subscriptions

```
var sub = web3.eth.subscribe('logs', options);

// watch for changes
sub.on('data', function(result){
    console.log(result);
});
```

Fabian Vogelsteller, web3.js and Mist Developer Ethereum
@feindura
http://frozeman.de/blog

Better modular structure

```
var utils = require('web3-utils');

utils.fromWei(2000000000, 'ether') > '0.00000000002'
utils.toString('0xffffff...') > 'Hello'
utils.toNumber('0xffffff...') > '12345'
utils.toChecksumAddress('0xffffffff...') > '0xfFFffffF...'
```

```
var web3 = require('web3');

web3.utils ...
web3.eth ...
web3.shh ...
web3.bzz ...
```

Fabian Vogelsteller, web3.js and Mist Developer Ethereum
@feindura
http://frozeman.de/blog

PromiseEvents

```
web3.eth.sendTransaction({from: '0x1234...'})
.then(function(receipt){
     // got tx receipt
}).catch(function(error){
});
```

```
eventEmitter.on('error', function(error){
 });
eventEmitter.on('transactionHash', function(hash){
 });
eventEmitter.on('mined', function(receipt) {
 });
```

Fabian Vogelsteller, web3.js and Mist Developer Ethereum
@feindura
http://frozeman.de/blog

PromiseEvents

```
 web3.eth.blockNumber()
.then(function(blockNumber){
      // got blockNumber
}).catch(function(error){
});
```

```
eventEmitter.on('error', function(error){
 });
eventEmitter.on('data', function(hash){
 });
```

Fabian Vogelsteller, web3.js and Mist Developer Ethereum
@feindura
http://frozeman.de/blog

Deploy contracts

```
var myContract = new web3.eth.contract(jsonInterface);

var eventEmitter = myContract.deploy({
     arguments: ['myValue', 1234],
     data: '0x1234..789'
})
.then(function(receipt){
     // have now contract receipt.contractAddress
     // as well as myContract.address is set
}).catch(function(error){
});

eventEmitter.on('error', function(error){
 });
eventEmitter.on('transactionHash', function(hash){
 });
eventEmitter.on('mined', function(receipt) {
 });
```

Fabian Vogelsteller, web3.js and Mist Developer Ethereum
@feindura
http://frozeman.de/blog

Better smart contract calling

```
// -> methods
myContract.myMethod(param1).estimateGas({from: '0x1234...'}) // > 50000

myContract.myMethod(param1).encode({from: '0x1234...'}) // > 0x12345600000...001234

myContract.myMethod(param1).call({from: '0x1234...'}) // > 'my return value'

myContract.myMethod(param1).send({from: '0x1234...'}) // > 0x1234...5678
```

```
// example
var eventEmitter = myContract.myMethod(param1).send({from: '0x1234...'})
.then(function(receipt){
     // got tx receipt
}).catch(function(error){
});

eventEmitter.on('error', function(error){
 });
eventEmitter.on('transactionHash', function(hash){
 });
eventEmitter.on('mined', function(receipt) {
 });
```

Get encoded ABI's

```
web3.utils.encode(jsonInterface, [1234, 'argument2'])
// > 0x123456780000...000a123000...00012345


var myContract = new web3.contract(jsonInterface, {data: '01234'});

// constructor
myContract.constructor(1234, 'argument2').encode()
// > 0x9876543210000...000a123000...00012345

// contract functions
myContract.myMethod(1234, 'argument2').encode()
// > 0x123456780000...000a123000...00012345
```

Fabian Vogelsteller, web3.js and Mist Developer Ethereum
@feindura
http://frozeman.de/blog

more to come…

e.g.
pass and return structs

Your input is welcome!

https://gitter.im/ethereum/web3.js
https://github.com/ethereum/web3.js
https://github.com/ethereum/EIPs/issues/68

Fabian Vogelsteller, web3.js and Mist Developer Ethereum
@feindura
http://frozeman.de/blog