



Ethereum Workshop

An Introduction to Tools, Solidity & Smart Contracts



Preparation

Follow the instructions on:
<http://bit.ly/2um6cGA>



Agenda

- 1) A brief introduction to Ethereum
- 2) Setting up a private blockchain
- 3) Interacting with the blockchain
- 4) Mist
- 5) Solidity & Smart Contracts
- 6) Remix IDE



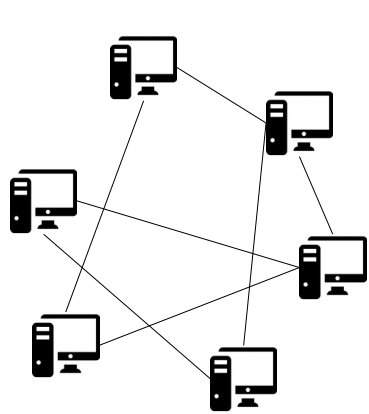
Warning

- Bleeding edge technology - Things might not work!
- Disable your Firewall or open port 30303 (UDP and TCP!)



1. A brief introduction to Ethereum

Blockchain Introduction



Peer-to-peer
network

+



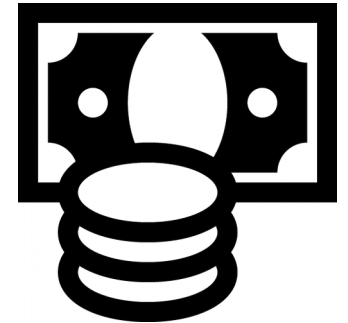
Decentralized
ledger/database

+



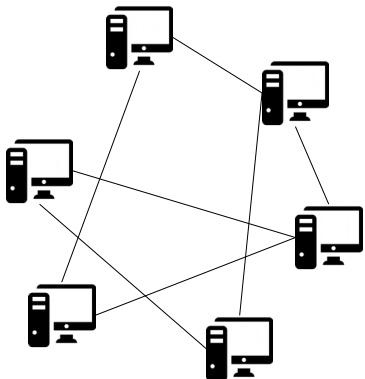
Consensus
algorithm

+



Cryptocurrency/
Economic incentives

Etherum



Peer-to-peer network

+



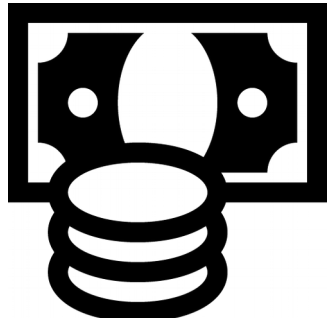
Decentralized ledger/database

+



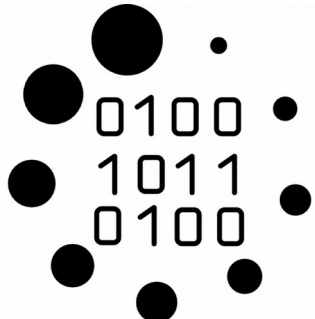
Consensus algorithm

+



Cryptocurrency/
Economic incentives

+



Smart Contracts

Why?

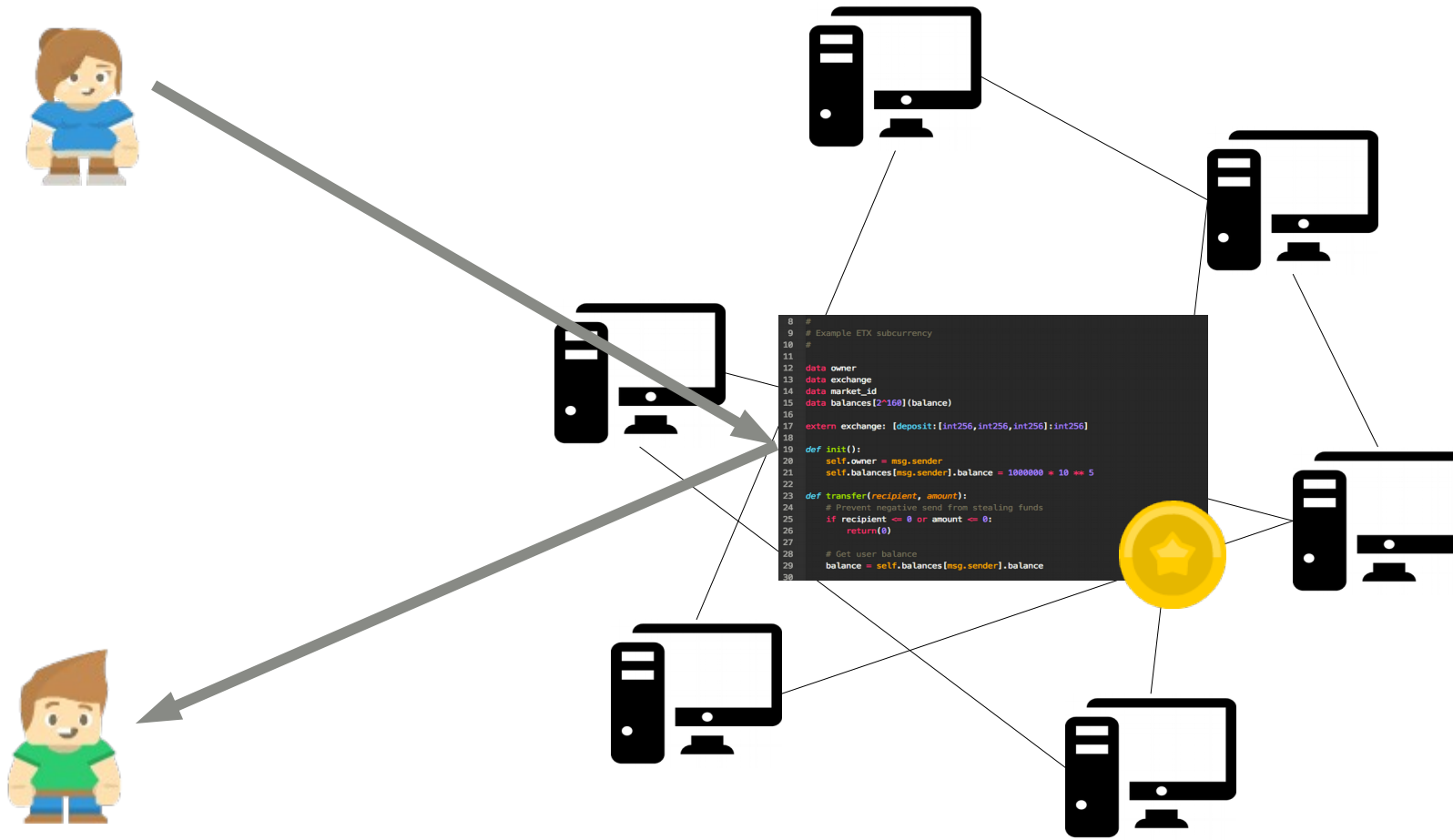


If X happens I'll send 500 Euro to Bob



I can make X happen ... but I don't trust Alice to send me the money

Why?





2. Setting up a private blockchain

Accounts



```
geth --datadir ~/.ethereum/workshop  
account new
```



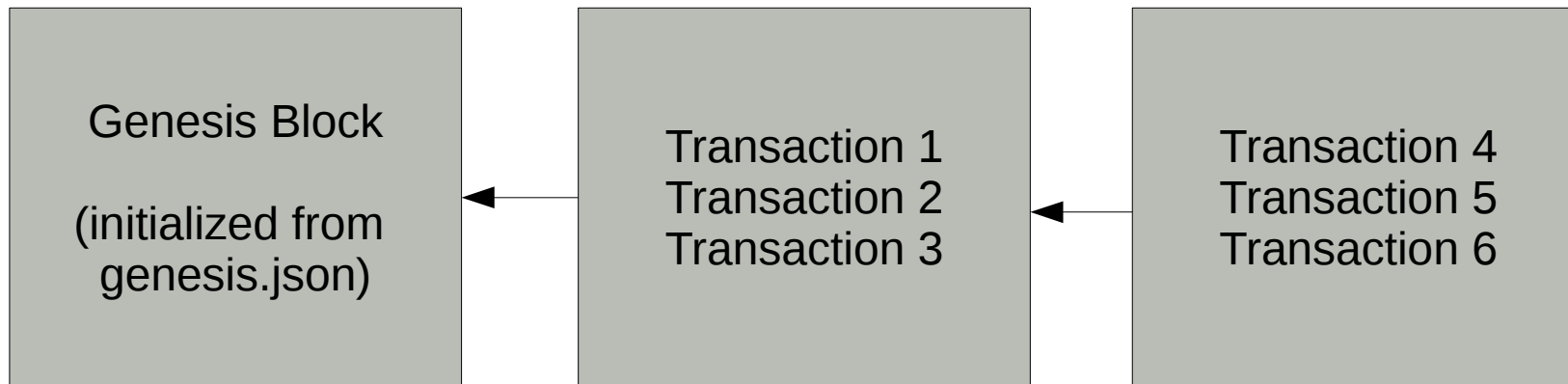
```
geth --datadir C:\Users\%HOMEPATH  
%\workshop account new
```

geth – Go Ethereum client

--datadir <DIRECTORY> – Store all data here. Avoids conflicts with the public chain

account new – Create a new account

Blockchain Data Structure



<https://ethereum.stackexchange.com/questions/2376/what-does-each-genesis-json-parameter-mean>
<https://ethereum.stackexchange.com/questions/15682/the-meaning-specification-of-config-in-genesis-json/15687#15687>
<https://ethereum.stackexchange.com/questions/5833/why-do-we-need-both-nonce-and-mixhash-values-in-a-block>

Initialize the blockchain



```
geth --datadir ~/.ethereum/workshop  
init genesis.json
```



```
geth --datadir C:\Users\%HOMEPATH  
%\workshop init genesis.json
```

init <GENESIS FILE> - Initialize a new blockchain from a genesis file

Start a miner



```
> geth --datadir ~/.ethereum/workshop  
--mine --networkid 1259
```



```
> geth --datadir C:\Users\%HOMEPATH  
%\workshop --mine --networkid 1259
```

--mine - Make this blockchain node a miner

--networkid <NUMBER> - Unique identifier for this network

Start a console

- Start a new terminal/cmd window and run:



```
geth attach ipc:///
$HOME/.ethereum/workshop/geth.ipc
```



```
geth attach
```

attach <PATH> - Attach a console to a running geth instance using IPC

Plan B if it doesn't work

- `geth --dev account new`

```
geth --dev --mine
```

--dev – Developer mode: pre-configured private network – Cannot connect to other nodes

- In a new terminal/cmd window run:

```
geth attach
```

```
/tmp/ethereum_dev_mode/geth.ipc
```

```
geth attach
```




2. Interacting with the blockchain

admin API

- `admin.nodeInfo` – Gives us the enode id and a bunch of useful information
- `admin.peers` – Lists all connected nodes our node knows
- `admin.addPeer("enode://fc[...]03")` – Manually add another node

<https://github.com/ethereum/go-ethereum/wiki/JavaScript-Console#management-api-reference>

Enode URL

```
enode://797038b92a15ebfbc181a2f68feb82  
0fd3c69c63b8094b35c23cc378c0a645f73c08  
31ab9b096301f30259b72436e82e2425f8683  
b5f9e6214030f8942b929b@[::]:30303
```

Replace [::] with your IP address. Example:

```
enode://797038b92a15ebfbc181a2f68feb82  
0fd3c69c63b8094b35c23cc378c0a645f73c08  
31ab9b096301f30259b72436e82e2425f8683  
b5f9e6214030f8942b929b@192.168.43.77:  
30303
```

personal API

- `personal.newAccount()` – Create a new account
- `personal.listAccounts` – List of all (local) accounts
- `personal.unlockAccount("0xc73[...]5b")` – Lists all connected nodes our node knows

<https://github.com/ethereum/go-ethereum/wiki/JavaScript-Console#management-api-reference>



web3js

- `web3.eth.getBalance("0xc[...]5b")` – Get balance of account. This works for all accounts.

<https://github.com/ethereum/wiki/wiki/JavaScript-API#web3js-api-reference>

Dealing with numbers

1000000000000000000	Wei
1000000000000000	Kwei
1000000000000	Mwei
1000000000	Gwei
1000000	Szabo
1000	Finney
1	Ether
0.001	Kether
0.000001	Mether
0.000000001	Gether
0.0000000000001	Tether

Dealing with numbers

- `web3.fromWei(number, "ether")` – Converts from wei to ether
<https://github.com/ethereum/wiki/wiki/JavaScript-API#web3fromwei>
- `web3.toWei(number, "ether")` – From ether to wei
<https://github.com/ethereum/wiki/wiki/JavaScript-API#web3towei>
- `number.toString()` – Converts a bignumber to a human-readable string
<http://mikemcl.github.io/bignumber.js/>
<https://github.com/ethereum/wiki/wiki/JavaScript-API#a-note-on-big-numbers-in-web3js>

web3js

- ```
web3.eth.sendTransaction(
 {"from": "0xc73e[...]2cfbc025b",
 "to": "0x00[...]00",
 "value": 111111
})
```

  - Send wei from an address to another address





# 3. Mist

# Starting Mist



```
mist --rpc ~/.ethereum/workshop/geth.ipc
```



```
"C:\Program Files\Mist\Mist.exe"
--rpc \\.\pipe\geth.ipc
```



```
/Applications/Mist.app/Contents/MacOS/Mist
--rpc ~/.ethereum/workshop/geth.ipc
```

- **--rpc** – Path to node IPC socket file OR HTTP RPC hostport

# Mist

The screenshot displays the Mist Ethereum wallet interface. At the top, the title bar reads "Mist" and the menu bar includes "Mist File Edit View Develop Window Help". The main header shows "Ethereum Wallet" and a URL: "https://wallet.ethereum.org send-from ▶ 0xC73E37361DE2D45e635E433d4F0d66e2CFbC025b". On the right, the balance is shown as "58,396.00 ETHER\*".

The central area is titled "Send funds" and contains the following fields:

- FROM:** Main account (Etherbase) - 306.00 ETHER
- TO:** 0xee3099424b2e9761b42364efe960a60f690519e3
- AMOUNT:** 123
- CURRENCY:** ETH (selected)
- RECIPIENT BALANCE:** 305.999999999999977778 ETHER

Below the amount field, there is a checkbox for "Send everything" which is unchecked. A summary line states: "You want to send **123 ETHER**". A blue button labeled "SHOW MORE OPTIONS" is positioned below this line.

At the bottom left, a sidebar shows network statistics: "44.4 KH/s", "11,879 1", and "14s". A red "Private" indicator is visible at the bottom left corner.

At the bottom right, a note reads: "This is the most amount of money that might be used to process this transaction. Your transaction will be mined **probably within 30 seconds**".



# **4. Solidity & Smart Contracts**

# Smart Contracts

```
pragma solidity ^0.4.15;
```

```
contract MyCoin {
```

```
 mapping (address => uint) balances;
```

```
 function MyCoin() {
```

```
 balances[tx.origin] = 10000;
```

```
 }
```

```
 function sendCoin(address receiver, uint amount) returns(bool sufficient) {
```

```
 if (balances[msg.sender] < amount) return false;
```

```
 balances[msg.sender] -= amount;
```

```
 balances[receiver] += amount;
```

```
 return true;
```

```
 }
```

```
 function getBalance(address addr) returns(uint) {
```

```
 return balances[addr];
```

```
 }
```

```
}
```

# Smart Contracts

```
pragma solidity ^0.4.15;
```

```
contract MyCoin {
 mapping (address => uint) balances;
```

```
 function MyCoin() {
 balances[tx.origin] = 10000;
 }
}
```

```
 function sendCoin(address receiver, uint amount) returns(bool sufficient) {
 if (balances[msg.sender] < amount) return false;
 balances[msg.sender] -= amount;
 balances[receiver] += amount;
 return true;
 }
}
```

```
 function getBalance(address addr) returns(uint) {
 return balances[addr];
 }
}
```

# Smart Contracts

```
pragma solidity ^0.4.15;
```

```
contract MyCoin {
 mapping (address => uint) balances;
```

```
 function MyCoin() {
 balances[tx.origin] = 10000;
 }
```

```
 function sendCoin(address receiver, uint amount) returns(bool sufficient) {
 if (balances[msg.sender] < amount) return false;
 balances[msg.sender] -= amount;
 balances[receiver] += amount;
 return true;
 }
```

```
 function getBalance(address addr) returns(uint) {
 return balances[addr];
 }
}
```

# Smart Contracts

```
pragma solidity ^0.4.15;
```

```
contract MyCoin {
 mapping (address => uint) balances;
```

```
 function MyCoin() {
 balances[tx.origin] = 10000;
 }
```

```
 function sendCoin(address receiver, uint amount) returns(bool sufficient) {
 if (balances[msg.sender] < amount) return false;
 balances[msg.sender] -= amount;
 balances[receiver] += amount;
 return true;
 }
```

```
 function getBalance(address addr) returns(uint) {
 return balances[addr];
 }
}
```



# Smart Contracts

```
pragma solidity ^0.4.15;
```

```
contract MyCoin {
 mapping (address => uint) balances;
```

```
 function MyCoin() {
 balances[tx.origin] = 10000;
 }
```

```
 function sendCoin(address receiver, uint amount) returns(bool sufficient) {
 if (balances[msg.sender] < amount) return false;
 balances[msg.sender] -= amount;
 balances[receiver] += amount;
 return true;
 }
```

```
 function getBalance(address addr) constant returns(uint) {
 return balances[addr];
 }
}
```



# 5. Remix IDE



# Opening Remix

- In Mist choose Develop → Open Remix IDE
- ... or ...
- Open <http://remix.ethereum.org/> in your browser

# Remix

The image shows the Remix Solidity IDE interface. The main editor displays the following Solidity code for a contract named `MyCoin`:

```
1 pragma solidity ^0.4.14;
2
3 contract MyCoin {
4 mapping (address => uint) balances;
5
6 function MyCoin() {
7 balances[tx.origin] = 10000;
8 }
9
10 function sendCoin(address receiver, uint amount) returns(bool sufficient) {
11 if (balances[msg.sender] < amount) return false;
12 balances[msg.sender] -= amount;
13 balances[receiver] += amount;
14 return true;
15 }
16
17 function getBalance(address addr) returns(uint) {
18 return balances[addr];
19 }
20 }
21
```

The right-hand sidebar contains the following elements:

- Environment: Injected Web3
- Account: 0xc73...c025b (1415.99999999999977778 €)
- Gas limit: 3000000
- Value: 0
- Transaction options: Publish, Attach, Transact, Transact(Payable), Call
- Contract name: browser/MyCoin.sol:MyCoin
- Buttons: Publish, At Address, Cre...
- Link: Contract details (bytecode, interface etc.)



# Questions?

[jonas.pfannschmidt@hpe.com](mailto:jonas.pfannschmidt@hpe.com)  
[jonas.pfannschmidt@gmail.com](mailto:jonas.pfannschmidt@gmail.com)



# Backup Slides



# Predefined network ids

- 0: Olympic - Deprecated test blockchain
- 1: Frontier/Homestead - Public blockchain
- 2: Morden - Deprecated test blockchain
- 3: Ropsten - Test blockchain
- 4: Rinkeby - Another test blockchain

# Gas

- Gas is the internal price of transactions and computational use
- Each computational step has a fixed gas usage count:

<https://docs.google.com/spreadsheets/d/1m89CVujrQe5LAFJ8-YAUCcNK950dUzMQPMJBxRtGCqs/edit#gid=0>

- Total cost = gasUsed \* gasPrice
- Unused gas is returned to the sender
- If a transaction runs out of gas it gets reverted (This prevents endless-loops, etc)