Blockchain Lab

Tut 02: Ethereum & Smart Contracts

Robert Muth
muth@tu-berlin.de

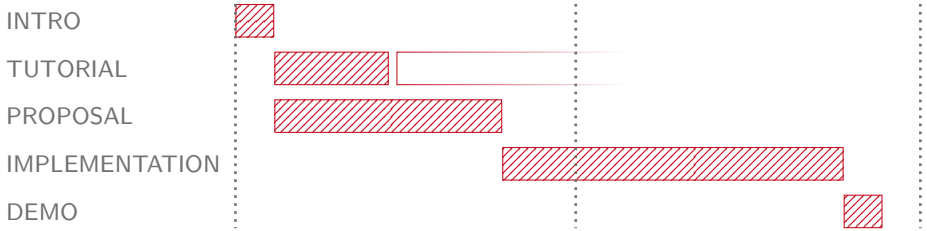
April 23, 2019

Today's Expections

- Organizational stuff ...
- Getting started with Ethereum and some of its tools
- Programming a basic smart contract
- Deployment sneak peak
- Time for your projects

Important Dates

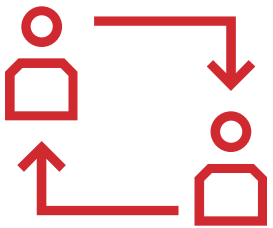
- Tutorial sessions
  - Holiday on May 1st
  - `Tut 03`: May 8
  - `Live coding 01`: May 15 (voluntary)
  - `Live coding 02`: May 22 (voluntary)
  - `Lecture`: May 30
- Group forming deadline: April 24, 2019, 23:59
- Proposal deadline: May 20, 2019, 23:59
- Project demo: July 3, 2019 (we don't know where, yet)



- Always check dates on ISIS!

Some of Last Year's Projects

- Money Lending Tracker
- Distributed Organ Transplantation Waiting List
- DNS Blockchain
- Blockchain Casino
- Chatting on Blockchain
- SmartDB (winning team)
- Donation Handling (charity purposes)



https://pad.systemli.org/p/lab

- The place to share snippets and links with your fellow students during this session
- Please don't be destructive!

Ethereum: Decentralized Smart Contract Platform

- Blockchain-based distributed system allowing for the decentralized execution of **smart contracts**.
- Introduced in 2013 by Vitalik Buterin[1].
- Some technical details can be found in the yellow paper[2].

[1] Buterin, Vitalik. "A Next-Generation Smart Contract and Decentralized Application Platform. White paper." `https://github.com/ethereum/wiki/wiki/White-Paper` (2013).

[2] Wood, Gavin. "Ethereum: A secure decentralised generalised transaction ledger." Ethereum Project Yellow Paper (2014).

Ethereum: Smart Contracts

- Ethereum virtual machine (EVM) is (almost) a Turing-complete computing platform.
- Every full node executes every instruction. But will they come to an end?

$\Rightarrow$ Halting problem, solved by limiting the amount of execution steps.

- Programmed in multiple high-level languages, most prevalent is **Solidity**.

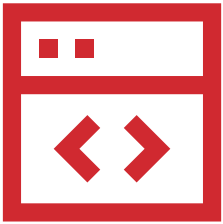
Ethereum: Accounts

- The shared state is comprised of small objects: **accounts**.
- Each account has an associated 160-bit identifier, its **address**.
- Two types of accounts:
  - **Externally owned** accounts are controlled by "the user" through her private keys.
  - **Contract** accounts are controlled by the resp. code.

Ethereum: Transactions

- Users can issue **transactions** from one externally owned account to another to transfer funds.
- Or they can use them to send **messages**, i.e., function calls to a contract account.
- This may induce contract accounts to send messages to other contract accounts.
- But: the user initiating the messages has to pay upfront for all resulting calculations.

Solidity Example with Remix



Live Coding! (sort of)

`https://remix.ethereum.org`

Group Task

DIY!

- `vote()` also returns the new results
- the owner cannot vote anymore
- a new `stopVoting()` function that does not self-destruct the contract but stops voting

Ethereum: Gas

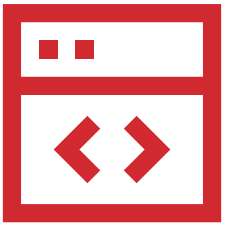
- Every transaction has to be paid via a transaction **fee**.
- Fees are typically measured in $10^9 Wei = 1 GWei = 10^{-9} Eth$.
- **Gas** is a relative unit to measure how much a specific computation consts.
- The sender of a message sets **gas price** and **gas limit**:

  Gas price: determines how much she is willing to pay for a unit of gas.
  Gas limit: determines the maximum amount of gas she is willing to pay.

Decentralized Apps (DApps)

- Decentralized: no typical client-server model.
- However, DApps usually consist of:
  - **Front-end** based on HTML / JavaScript in the browser.
  - A browser plugin functions as a light client that queries full nodes.
  - Ethereum full nodes function as a **back-end** which processes messages sent to smart contracts.

Deploying Contract



- ... to Ganache
- ... with the Truffle Framework
- ... to Rinkeby (Ethereum Testnet)

Proposed Development Toolchain

JavaScript API	`web3.js`	`https://github.com/ethereum/web3.js/`
Browser extension	`MetaMask`	`https://metamask.io`
Dev. Framework	`Truffle`	`http://truffleframework.com`
Dev. Blockchain	`Ganache`	`http://truffleframework.com/ganache/`
Testnet	`Rinkeby`	`https://www.rinkeby.io`

Assignments

Due April 24, 23:59:

- Register your group on ISIS

Due May 8:

- If not already: get your toolchain running (incl. `geth`)!

Strongly recommended:

- Do the Pet Shop tutorial[3]
- Try to deploy it to the Rinkeby testnet[4]
- Send me an email with the address of your deployed contract

[3] `http://truffleframework.com/tutorials/pet-shop`

[4] `https://blog.abuiles.com/blog/2017/07/09/deploying-truffle-contracts-to-rinkeby/`

Additional Resources

- Solidity Documentation[5]
- Truffle Framework Tutorials[6]
- CryptoZombies Solidity Tutorial[7]

[5] https://solidity.readthedocs.io/
[6] http://truffleframework.com/tutorials/
[7] https://cryptozombies.io