OST
Eastern Switzerland
University of Applied Sciences

Blockchain (BlCh)

DS1 part 3

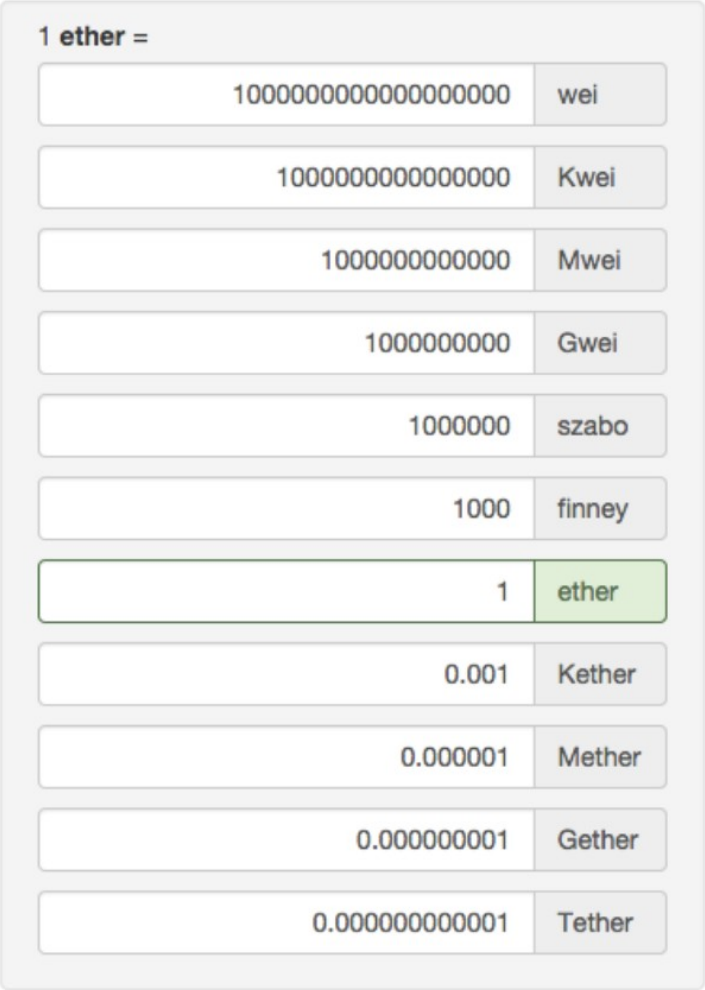
Thomas Bocek

04.10.2021

Lecture 11

Blocktime and Gas

- Gas Price set by Miner
  - Gas price ~59 gwei
- Miner decides which transaction at which gas price to include
  - Market for TX
- Gas price too low, longer waiting time until TX will be included



OST

Blocktime and Gas

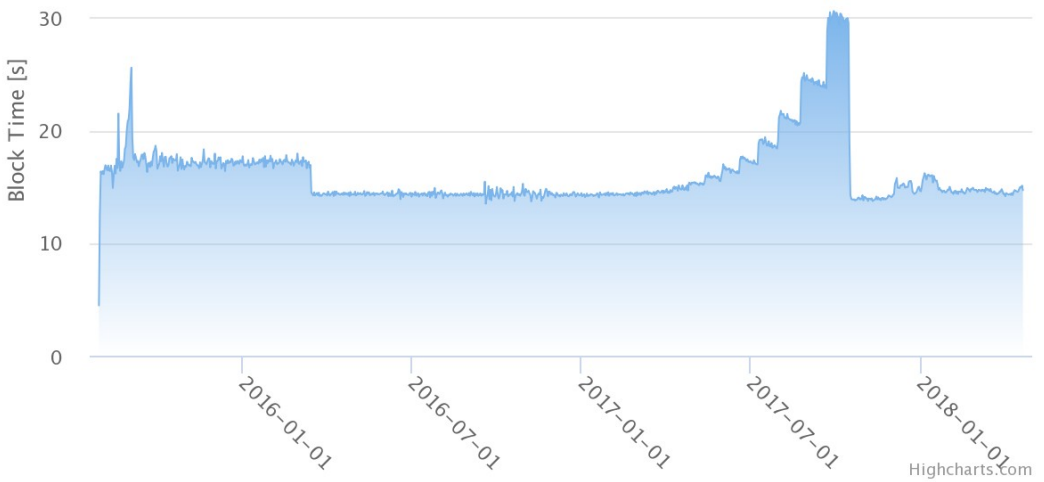
- Block time: ~14-15s
  - Ice age
- Smart Contracts are turing complete
  - Every instruction needs to be paid for (example)
- Gas price / Gas limit by miners
  - If you run out of gas, state is reverted, ETH gone

$W_{zero} = \{\text{STOP, RETURN}\}$
$W_{base} = \{\text{ADDRESS, ORIGIN, CALLER, CALLVALUE, CALLDATASIZE, CODESIZE, GASPRICE, COINBASE, TIMESTAMP, NUMBER, DIFFICULTY, GASLIMIT, POP, PC, MSIZE, GAS}\}$
$W_{verylow} = \{\text{ADD, SUB, NOT, LT, GT, SLT, SGT, EQ, ISZERO, AND, OR, XOR, BYTE, CALLDATALOAD, MLOAD, MSTORE, MSTORE8, PUSH*, DUP*, SWAP*}\}$
$W_{low} = \{\text{MUL, DIV, SDIV, MOD, SMOD, SIGNEXTEND}\}$
$W_{mid} = \{\text{ADDMOD, MULMOD, JUMP}\}$
$W_{high} = \{\text{JUMPI}\}$
$W_{extcode} = \{\text{EXTCODESIZE}\}$

Appendix G. Fee Schedule

The fee schedule G is a tuple of 31 scalar values corresponding to the relative costs, in gas, of a number of abstract operations that a transaction may effect.

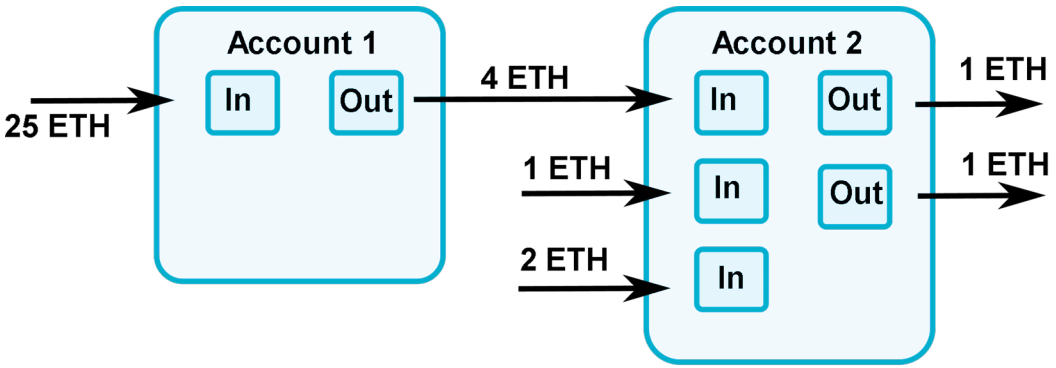
Name	Value	Description*
G_{zero}	0	Nothing paid for operations of the set W_{zero}.
G_{base}	2	Amount of gas to pay for operations of the set W_{base}.
$G_{verylow}$	3	Amount of gas to pay for operations of the set $W_{verylow}$.
G_{low}	5	Amount of gas to pay for operations of the set W_{low}.
G_{mid}	8	Amount of gas to pay for operations of the set W_{mid}.
G_{high}	10	Amount of gas to pay for operations of the set W_{high}.
$G_{extcode}$	700	Amount of gas to pay for operations of the set $W_{extcode}$.
$G_{balance}$	400	Amount of gas to pay for a BALANCE operation.
G_{sload}	200	Paid for a SLOAD operation.
$G_{jumpdest}$	1	Paid for a JUMPDEST operation.
G_{sset}	20000	Paid for an SSTORE operation when the storage value is set to non-zero from zero.
G_{sreset}	5000	Paid for an SSTORE operation when the storage value's zeroness remains unchanged or is set to zero.
R_{sclear}	15000	Refund given (added into refund counter) when the storage value is set to zero from non-zero.
$R_{suicide}$	24000	Refund given (added into refund counter) for suiciding an account.
$G_{suicide}$	5000	Amount of gas to pay for a SUICIDE operation.
G_{create}	32000	Paid for a CREATE operation.
$G_{codedeposit}$	200	Paid per byte for a CREATE operation to succeed in placing code into state.
G_{call}	700	Paid for a CALL operation.
$G_{callvalue}$	9000	Paid for a non-zero value transfer as part of the CALL operation.
$G_{callstipend}$	2300	A stipend for the called contract subtracted from $G_{callvalue}$ for a non-zero value transfer.
$G_{newaccount}$	25000	Paid for a CALL or SUICIDE operation which creates an account.
G_{exp}	10	Partial payment for an EXP operation.
$G_{expbyte}$	10	Partial payment when multiplied by $\lceil \log_{256}(exponent) \rceil$ for the EXP operation.
G_{memory}	3	Paid for every additional word when expanding memory.
$G_{txcreate}$	32000	Paid by all contract-creating transactions after the *Homestead transition*.
$G_{txdatazero}$	4	Paid for every zero byte of data or code for a transaction.
$G_{txdatanonzero}$	68	Paid for every non-zero byte of data or code for a transaction.
$G_{transaction}$	21000	Paid for every transaction.
G_{log}	375	Partial payment for a LOG operation.
$G_{logdata}$	8	Paid for each byte in a LOG operation's data.
$G_{logtopic}$	375	Paid for each topic of a LOG operation.
G_{sha3}	30	Paid for each SHA3 operation.
$G_{sha3word}$	6	Paid for each word (rounded up) for input data to a SHA3 operation.
G_{copy}	3	Partial payment for *COPY operations, multiplied by words copied, rounded up.
$G_{blockhash}$	20	Payment for BLOCKHASH operation.



Account vs UTXO - Introduction

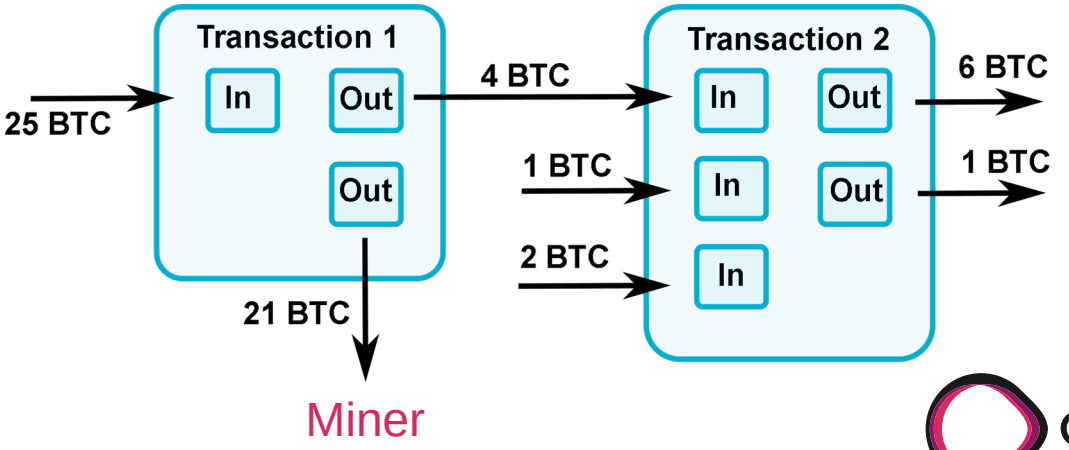
Account-based

- Global state stores a list of accounts with balances and code
- Transaction is valid if the sending account has enough balance
- Balance on sender is deducted, new balance
- Signature must match sending account
- If the receiving account has code, the code runs, and state may be changed



UTXO-based

- Every referenced input must be valid and not yet spent
- Total value of the inputs must equal or exceed the total value of the outputs
- You always spend all outputs
- Transaction needs to execute successful a script to determine if input is valid



OST

Solidity - https://solidity.readthedocs.io

- Version Pragma - reject compiled with specific compiler versions

```
// SPDX-License-Identifier:
pragma solidity ^0.8.9; //not before 0.8.9, not after 0.9.0
```

- Comments

```
// This is a single-line comment.
/*
This is a
multi-line comment.
*/
```

- Contract with State Variables (state change is expensive!)

```
pragma solidity ^0.8.9;
//minimal contract
contract Example1 {
    uint256 counter;
}
```

https://learnxinyminutes.com/docs/solidity/

https://solidity.readthedocs.io

OST

Solidity - https://solidity.readthedocs.io

- Types
  - bool: true and false, int / **uint** (int8, int16, …, int256), **address**: 20 bytes, fixed size arrays: bytes1, bytes2, bytes3, ..., bytes32, variable sized: bytes (push), strings
- Structs

```
struct Account {
  string name;
  uint256 amount;
}
```

- Mapping

```
mapping (address => uint256) accounts; //mapping with basic types
mapping (uint256 => Account) accounts; //mapping with structs
```


OST

Solidity – Basics

- Arrays

```
string[] names

uint256 newLength = names.push("John");
```

- Another contract

```
pragma solidity ^0.8.9;

contract Example2 {

    struct Account {

       string name;

       uint256 amount;

    }
    uint256 public counter;

    mapping (uint256 => Account) accounts;

}
```

- State changing/non-state changing

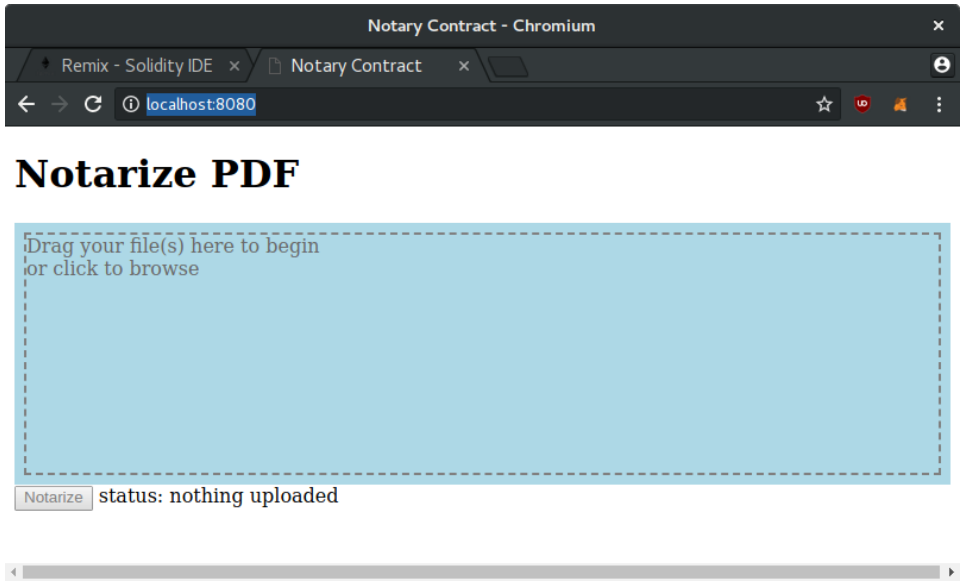
```
function get(uint256 nr) public view returns
(string memory) {

    return accounts[nr].addr;

}

function set(uint256 nr, string memory name) public
{

    require(owner == msg.sender);

    accounts[counter++] = Account(name, nr);

}
```

- Read state variables
  - “view”/”pure” function does not modify state
  - Reads “for free”, “pure” does not even read e.g., 2+2

OST

Example

- Installation
  - npm install
  - npx webpack
  - npx webpack-dev-server
- Open Browser: http://localhost:8080/




```
draft@home: ~/git/VSS-web3js
File  Edit  View  Search  Terminal  Help
draft@home:~/git/VSS-web3js$ ./node_modules/.bin/webpack-dev-server
i ｢wds｣: Project is running at http://localhost:8080/
i ｢wds｣: webpack output is served from /
i ｢wdm｣: Hash: c4c7c0d3279286de6649
Version: webpack 4.7.0
Time: 1139ms
Built at: 2018-05-06 12:57:52
                         Asset       Size  Chunks             Chunk Names
main.c4c7c0d3279286de6649.js    947 KiB    main  [emitted]  main
                  index.html  395 bytes          [emitted]
Entrypoint main = main.c4c7c0d3279286de6649.js
[./node_modules/ansi-html/index.js] 4.16 KiB {main} [built]
[./node_modules/loglevel/lib/loglevel.js] 7.68 KiB {main} [built]
[./node_modules/strip-ansi/index.js] 161 bytes {main} [built]
[./node_modules/url/url.js] 22.8 KiB {main} [built]
[./node_modules/vue/dist/vue.esm.js] 286 KiB {main} [built]
[./node_modules/webpack-dev-server/client/index.js?http://localhost:8080] (webpack)-dev-server/c
lient?http://localhost:8080 7.75 KiB {main} [built]
[./node_modules/webpack-dev-server/client/overlay.js] (webpack)-dev-server/client/overlay.js 3.5
8 KiB {main} [built]
[./node_modules/webpack-dev-server/client/socket.js] (webpack)-dev-server/client/socket.js 1.05
KiB {main} [built]
[./node_modules/webpack/hot sync ^\.\/log$] (webpack)/hot sync nonrecursive ^\.\/log$ 170 bytes
{main} [built]
[./node_modules/webpack/hot/emitter.js] (webpack)/hot/emitter.js 77 bytes {main} [built]
[./node_modules/webpack/hot/log.js] (webpack)/hot/log.js 1010 bytes {main} [optional] [built]
[./src/App.vue] 908 bytes {main} [built]
[./src/App.vue?vue&type=template&id=7ba5bd90] 194 bytes {main} [built]
[0] multi (webpack)-dev-server/client?http://localhost:8080 ./src 40 bytes {main} [built]
[./src/index.js] 129 bytes {main} [built]
    + 63 hidden modules
Child html-webpack-plugin for "index.html":
     1 asset
    Entrypoint undefined = index.html
    [./node_modules/html-webpack-plugin/lib/loader.js!./index.html] 527 bytes {0} [built]
    [./node_modules/lodash/lodash.js] 527 KiB {0} [built]
    [./node_modules/webpack/buildin/global.js] (webpack)/buildin/global.js 489 bytes {0} [built]
    [./node_modules/webpack/buildin/module.js] (webpack)/buildin/module.js 497 bytes {0} [built]
i ｢wdm｣: Compiled successfully.
```

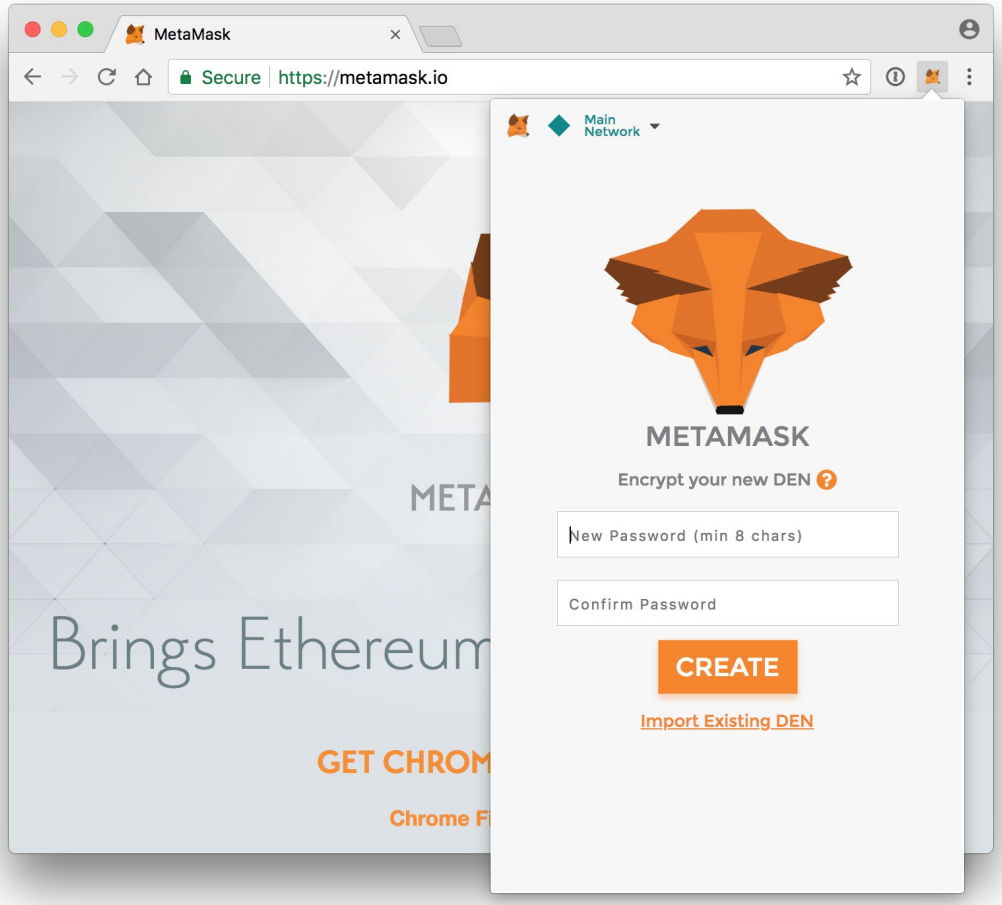

OST

MetaMask

- MetaMask
  - Browser plugin to make Ethereum transactions in browsers
  - Manage your key pairs and sign blockchain transactions
  - Use javascript library - ethers.js to call contracts
  - Uses infura

- Remix IDE: https://remix.ethereum.org
  - Use Notary.sol from https://github.com/tbocek/FS21/blob/main/ethereum/Notary.sol

  - Alternatively, use IntelliJ solidity plugin and deploy via geth, parity, or a local test blockchain



OST

ERC20

- ERC20 is a technical standard for smart contracts on the Ethereum blockchain for implementing tokens
  - proposed on November 19, 2015
  - Mai 2021: more than 392’868 ERC20 token contracts: …
- ERC20 Token (Simplified)
  - No allowance / approval / transferFrom, also no Approval event
  - Creator gets 1000 coins
  - Before 0.8.0 - SafeMath

```
abstract contract SimpleERC20 {
    function totalSupply() public virtual returns (uint256);
    function balanceOf(address who) public virtual returns (uint256);
    function transfer(address to, uint256 value) public virtual returns (bool);
    event Transfer(address indexed from, address indexed to, uint256 value);
}
    string public constant name = "VSS-TOKEN";
    string public constant symbol = "VST";
    uint8 public constant decimals = 18;

    mapping(address => uint256) balances;
    uint256 totalSupply_;

    constructor() {
        totalSupply_ = 1000 * (10**18);
        balances[msg.sender] = totalSupply_;
    }
...
```


OST