

Ethereum Smart Contracts

Lars Hupel
Software Architecture Summit
2019-09-18

INNOQ

workshop.hupel.info

Refresher

Konto/Adresse Schlüsselpaar aus öffentlichem und privatem Schlüssel

Refresher

Transaktion Tupel aus Sender, Empfänger, Betrag; wird vom Sender kryptografisch signiert

Refresher

Block Sammlung mehrere Transaktionen plus Zufallszahl, damit eine bestimmtes kryptografisches Puzzle gelöst wird

Refresher

Miner Person oder Organisation, die Blöcke durch Einsatz von Rechenleistung findet

Refresher

Block-Belohnung spezielle Transaktion in jedem Block, mit der Löser des Puzzles eine festgelegte Menge an Tokens erhält

Refresher

Token äquivalent zu einer Geldmenge; entsteht ausschließlich durch die Block-Belohnung

Refresher

Transaktions-Gebühr

Trinkgeld

- vom Sender einer Transaktion
- an den Miner, der diese Transaktion in einen Block inkludiert
- als Anreiz ebendafür

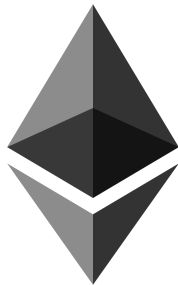
Refresher

Kontostand Σ Eingänge - Σ Ausgänge

Refresher

Blockchain monotone verkettete Liste aller bisher gefundenen Blöcke

Ethereum



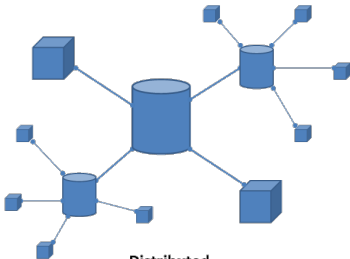
- eine der jüngeren Blockchains
- Währung: Ether (ETH)
- Platz 2–3 in Marktkapitalisierung
- technische Neuerung: Plattform für *Decentralized Apps*

Decentralized Apps



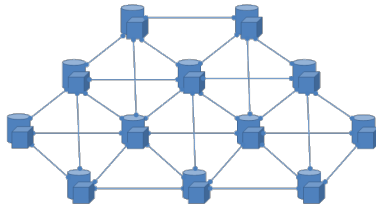
Centralized

one node does everything



Distributed

nodes distribute work to sub-nodes



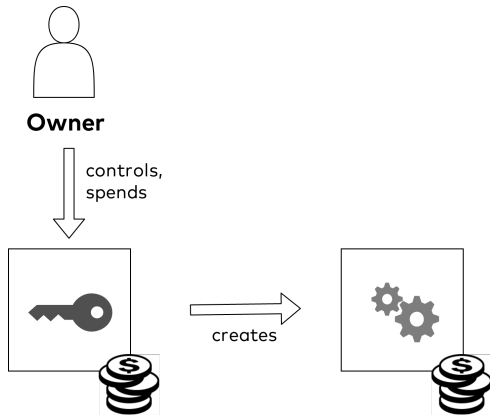
Decentralized

nodes are only connected to peers

Ethereum

Zwei Arten von Accounts:

1. Externally Owned
(natürliche Person)
2. Smart Contracts
(juristische Person)





Smart Contracts



Smart Contracts

... sind spezielle Accounts mit eigener Identität und Kontostand

Smart Contracts

- ... sind spezielle Accounts mit eigener Identität und Kontostand
- ... haben keine Ownership

Smart Contracts

- ... sind spezielle Accounts mit eigener Identität und Kontostand
- ... haben keine Ownership
- ... enthalten Speicherzellen mit veränderlichen Werten

Smart Contracts

- ... sind spezielle Accounts mit eigener Identität und Kontostand
- ... haben keine Ownership
- ... enthalten Speicherzellen mit veränderlichen Werten
- ... bestehen aus unveränderlichem Code

Smart Contracts

- ... sind spezielle Accounts mit eigener Identität und Kontostand
- ... haben keine Ownership
- ... enthalten Speicherzellen mit veränderlichen Werten
- ... bestehen aus unveränderlichem Code
- ... können mit anderen Contracts interagieren, werden aber nie selbstständig aktiv

Vergleich: **digitaler Notar**

- Rechte und Pflichten vertraglich festgelegt
- neutrale Instanz überwacht Ablauf
- Interaktion mit dritten Parteien

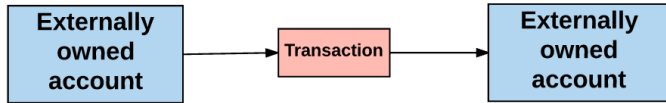
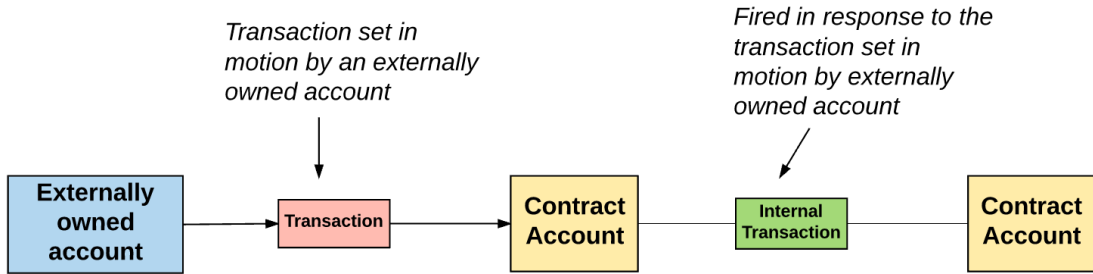


Transaktionsarten

Payment Übertrag von Geld in einen Contract oder EOA

Call Aufruf einer Funktion

Internal Contract interagiert mit anderem Contract

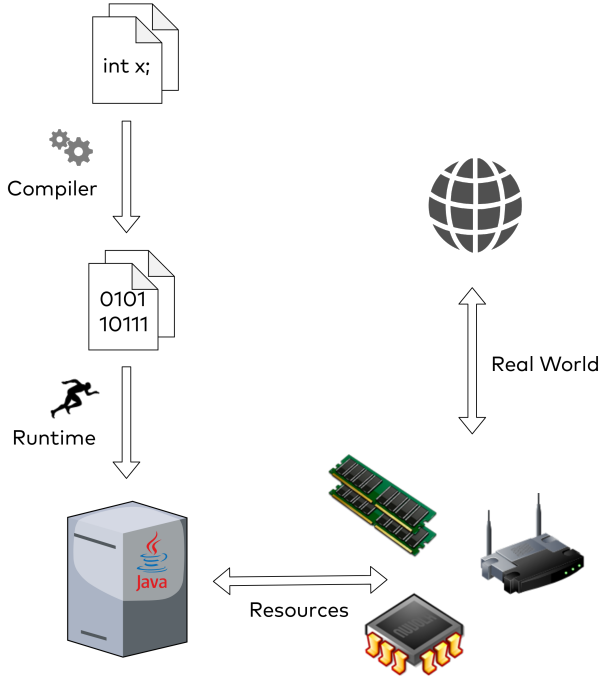




Ethereum Virtual Machine



Virtuelle Maschinen sind Mischformen zwischen Interpreter und Compiler



Ablauf einer Transaktion

Ablauf einer Transaktion

1. User erzeugt Transaktion:
 - ▶ Signatur (enthält Quelle)
 - ▶ Ziel
 - ▶ Geldbetrag
 - ▶ Daten

Ablauf einer Transaktion

1. User erzeugt Transaktion:

- ▶ Signatur (enthält Quelle)
- ▶ Ziel
- ▶ Geldbetrag
- ▶ Daten

2. Miner überprüfen Gültigkeit

Ablauf einer Transaktion

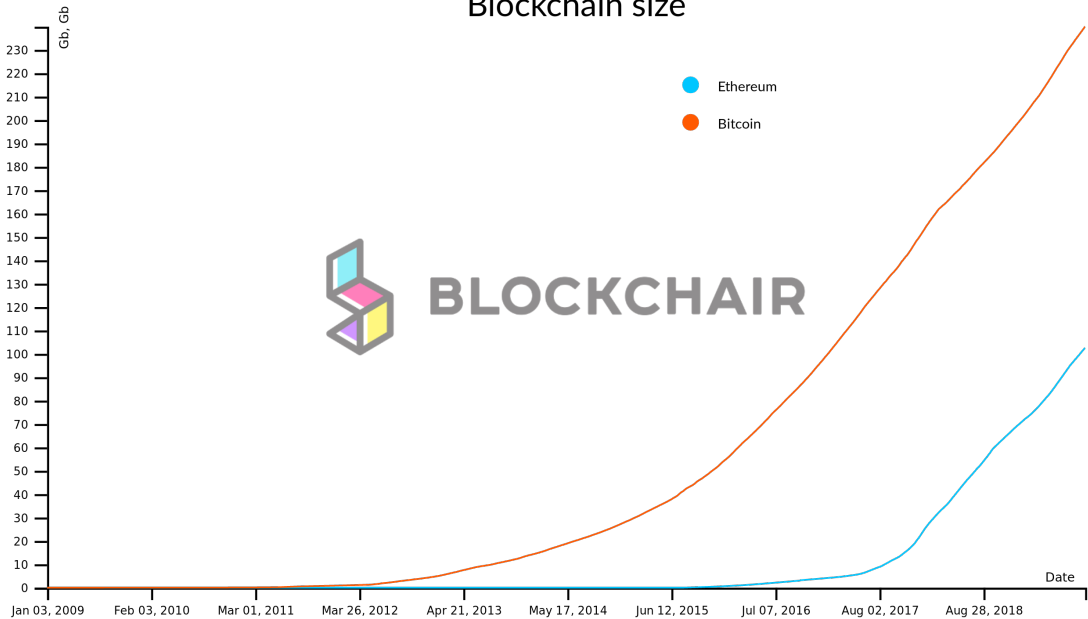
1. User erzeugt Transaktion:
 - ▶ Signatur (enthält Quelle)
 - ▶ Ziel
 - ▶ Geldbetrag
 - ▶ Daten
2. Miner überprüfen Gültigkeit
3. Miner führen Code aus, falls Ziel ein Contract ist

Ablauf einer Transaktion

1. User erzeugt Transaktion:
 - ▶ Signatur (enthält Quelle)
 - ▶ Ziel
 - ▶ Geldbetrag
 - ▶ Daten
2. Miner überprüfen Gültigkeit
3. Miner führen Code aus, falls Ziel ein Contract ist
4. Persistieren der Änderungen in die Blockchain

```
interface Transaction {  
    // signature: contains public key of EOA  
    Signature getSignature();  
  
    // recipient address, can be any address type  
    Address getRecipient();  
  
    // amount of Ether to be sent, can be 0  
    Value getValue();  
  
    // data for a call, can be empty  
    String getData();  
  
    // internal TXNs  
    List<Transaction> getInternalTxns();  
}
```

Blockchain size



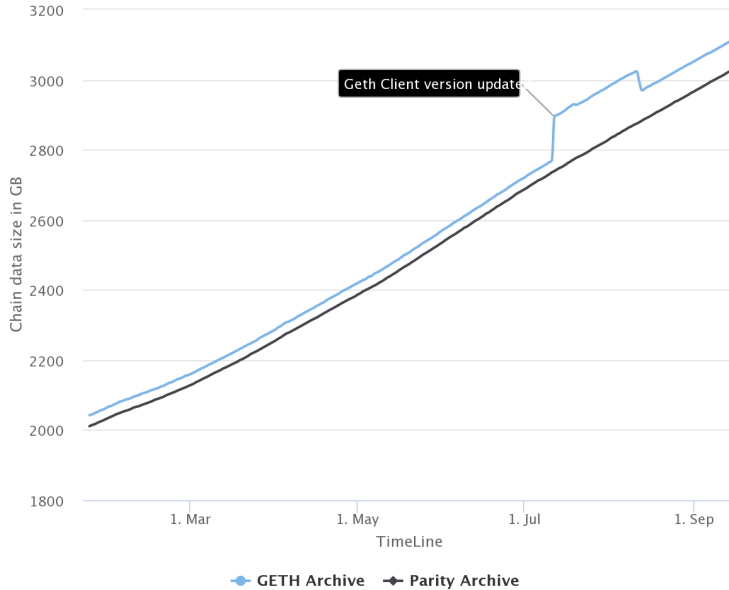
BLOCKCHAIR

<https://blockchair.com/ethereum/charts/blockchain-size?compare=bitcoin>

Ethereum Full Node Sync Archive Chart

Source: Etherscan.io

Click and drag in the plot area to zoom in



Was kann die EVM?

1. arithmetische Operationen
2. bitweise & logische Operationen
3. Hashes (meist SHA-3)
4. Umgebungsinformation (Kontostand, Aufrufer, ...)
5. Blockinformation (Zeitstempel, Blocknummer, ...)
6. Stack- und Speicheroperationen
7. Sprünge
8. Logging
9. Kontooperationen (Erzeugung, Zahlung, Aufruf, ...)

[illegible][illegible][illegible][illegible][illegible][illegible][illegible]

Name		Source	Category	Language	Version	License
1	Value	Homework 1	0	Overview		
2	Unit	1000000	1	1	1	1
3	Unit	1000000	1	1	1	1
4	Unit	1000000	1	1	1	1
5	Unit	1000000	1	1	1	1
6	Unit	1000000	1	1	1	1
7	Unit	1000000	1	1	1	1
8	Unit	1000000	1	1	1	1
9	Unit	1000000	1	1	1	1
10	Unit	1000000	1	1	1	1
11	Unit	1000000	1	1	1	1
12	Unit	1000000	1	1	1	1
13	Unit	1000000	1	1	1	1
14	Unit	1000000	1	1	1	1
15	Unit	1000000	1	1	1	1
16	Unit	1000000	1	1	1	1
17	Unit	1000000	1	1	1	1
18	Unit	1000000	1	1	1	1
19	Unit	1000000	1	1	1	1
20	Unit	1000000	1	1	1	1
21	Unit	1000000	1	1	1	1
22	Unit	1000000	1	1	1	1
23	Unit	1000000	1	1	1	1
24	Unit	1000000	1	1	1	1
25	Unit	1000000	1	1	1	1
26	Unit	1000000	1	1	1	1
27	Unit	1000000	1	1	1	1
28	Unit	1000000	1	1	1	1
29	Unit	1000000	1	1	1	1
30	Unit	1000000	1	1	1	1
31	Unit	1000000	1	1	1	1
32	Unit	1000000	1	1	1	1
33	Unit	1000000	1	1	1	1
34	Unit	1000000	1	1	1	1
35	Unit	1000000	1	1	1	1
36	Unit	1000000	1	1	1	1
37	Unit	1000000	1	1	1	1
38	Unit	1000000	1	1	1	1
39	Unit	1000000	1	1	1	1
40	Unit	1000000	1	1	1	1
41	Unit	1000000	1	1	1	1
42	Unit	1000000	1	1	1	1
43	Unit	1000000	1	1	1	1
44	Unit	1000000	1	1	1	1
45	Unit	1000000	1	1	1	1
46	Unit	1000000	1	1	1	1
47	Unit	1000000	1	1	1	1
48	Unit	1000000	1	1	1	1
49	Unit	1000000	1	1	1	1
50	Unit	1000000	1	1	1	1
51	Unit	1000000	1	1	1	1
52	Unit	1000000	1	1	1	1
53	Unit	1000000	1	1	1	1
54	Unit	1000000	1	1	1	1
55	Unit	1000000	1	1	1	1
56	Unit	1000000	1	1	1	1
57	Unit	1000000	1	1	1	1
58	Unit	1000000	1	1	1	1
59	Unit	1000000	1	1	1	1
60	Unit	1000000	1	1	1	1
61	Unit	1000000	1	1	1	1
62	Unit	1000000	1	1	1	1
63	Unit	1000000	1	1	1	1
64	Unit	1000000	1	1	1	1
65	Unit	1000000	1	1	1	1

[illegible][illegible]

Message-call into an account.

$$\mathbf{i} \equiv \mu_{\mathbf{m}}[\mu_{\mathbf{s}}[3] \dots (\mu_{\mathbf{s}}[3] + \mu_{\mathbf{s}}[4] - 1)]$$

$$(\sigma', g', A^+, \mathbf{o}) \equiv \begin{cases} \Theta(\sigma, I_{\mathbf{a}}, I_{\mathbf{o}}, t, t, C_{\text{CALLGAS}}(\mu), & \text{if } \mu_{\mathbf{s}}[2] \leq \sigma[I_{\mathbf{a}}]_{\mathbf{b}} \wedge \\ I_{\mathbf{p}}, \mu_{\mathbf{s}}[2], \mu_{\mathbf{s}}[2], \mathbf{i}, I_{\mathbf{e}} + 1, I_{\mathbf{w}}) & I_{\mathbf{e}} < 1024 \\ (\sigma, g, \emptyset, ()) & \text{otherwise} \end{cases}$$

$$n \equiv \min(\{\mu_{\mathbf{s}}[6], |\mathbf{o}|\})$$

$$\mu'_{\mathbf{m}}[\mu_{\mathbf{s}}[5] \dots (\mu_{\mathbf{s}}[5] + n - 1)] = \mathbf{o}[0 \dots (n - 1)]$$

$$\mu'_{\mathbf{o}} = \mathbf{o}$$

$$\mu'_{\mathbf{g}} \equiv \mu_{\mathbf{g}} + g'$$

$$\mu'_{\mathbf{s}}[0] \equiv x$$

$$A' \equiv A \uplus A^+$$

$$t \equiv \mu_{\mathbf{s}}[1] \bmod 2^{160}$$

where $x = 0$ if the code execution for this operation failed due to an

exceptional halting; (or for a REVERT) $\sigma' = \emptyset$ or if

$\mu_{\mathbf{s}}[2] > \sigma[I_{\mathbf{a}}]_{\mathbf{b}}$ (not enough funds) or $I_{\mathbf{e}} = 1024$ (call depth limit reached); $x = 1$ otherwise.

$$\mu'_{\mathbf{i}} \equiv M(M(\mu_{\mathbf{i}}, \mu_{\mathbf{s}}[3], \mu_{\mathbf{s}}[4]), \mu_{\mathbf{s}}[5], \mu_{\mathbf{s}}[6])$$

Thus the operand order is: gas, to, value, in offset, in size, out offset, out size.

$$C_{\text{CALL}}(\sigma, \mu) \equiv C_{\text{GASCAP}}(\sigma, \mu) + C_{\text{EXTRA}}(\sigma, \mu)$$

$$C_{\text{CALLGAS}}(\sigma, \mu) \equiv \begin{cases} C_{\text{GASCAP}}(\sigma, \mu) + G_{\text{callstipend}} & \text{if } \mu_{\mathbf{s}}[2] \neq 0 \\ C_{\text{GASCAP}}(\sigma, \mu) & \text{otherwise} \end{cases}$$

$$C_{\text{GASCAP}}(\sigma, \mu) \equiv \begin{cases} \min\{L(\mu_{\mathbf{g}} - C_{\text{EXTRA}}(\sigma, \mu)), \mu_{\mathbf{s}}[0]\} & \text{if } \mu_{\mathbf{g}} \geq C_{\text{EXTRA}}(\sigma, \mu) \\ \mu_{\mathbf{s}}[0] & \text{otherwise} \end{cases}$$

$$C_{\text{EXTRA}}(\sigma, \mu) \equiv G_{\text{call}} + C_{\text{XFER}}(\mu) + C_{\text{NEW}}(\sigma, \mu)$$

$$C_{\text{XFER}}(\mu) \equiv \begin{cases} G_{\text{callvalue}} & \text{if } \mu_{\mathbf{s}}[2] \neq 0 \\ 0 & \text{otherwise} \end{cases}$$

$$C_{\text{NEW}}(\sigma, \mu) \equiv \begin{cases} G_{\text{newaccount}} & \text{if } \text{DEAD}(\sigma, \mu_{\mathbf{s}}[1] \bmod 2^{160}) \wedge \mu_{\mathbf{s}}[2] \neq 0 \\ 0 & \text{otherwise} \end{cases}$$

- Smart Contracts werden von **allen Minern** ausgeführt



- Smart Contracts werden von **allen Minern** ausgeführt
- Berechnung der **Transaktionsgebühr** anhand der Ausführungsdauer



- Smart Contracts werden von **allen Minern** ausgeführt
- Berechnung der **Transaktionsgebühr** anhand der Ausführungsdauer
- User muss eine **Tankgröße** und einen **Spritpreis** angeben



- Smart Contracts werden von **allen Minern** ausgeführt
- Berechnung der **Transaktionsgebühr** anhand der Ausführungsdauer
- User muss eine **Tankgröße** und einen **Spritpreis** angeben
- jede Operation hat einen definierten **Verbrauch**



Send ETH



Only send ETH to an Ethereum address.

From:



Test

2.998131 ETH
\$411.85 USD

To:

0x0a1c8acab8a47D27dA0e

Amount:

1 ETH

\$137.37 USD



Max

Transaction
Fee:

Slow

0.00008 ETH
\$0.01

Average

0.00021 ETH
\$0.03

Fast

0.00042 ETH
\$0.06

[Advanced Options](#)

CANCEL

NEXT

Customize Gas

[Close](#)

Advanced

New Transaction Fee

0.00021 ETH

~Transaction Time

~31 sec

Gas Price (GWEI)

10



Gas Limit

21000



Live Gas Price Predictions



Send Amount

1 ETH

Transaction Fee

0.00021 ETH

New Total

1.00021 ETH

\$137.40

SAVE

```

pragma solidity >=0.4.22 <0.6.0;

contract MyToken {
    /* This creates an array with all balances */
    mapping (address => uint256) public balanceOf;

    /* Initializes contract with initial supply tokens to the creator of the contract */
    constructor(
        uint256 initialSupply
    ) public {
        balanceOf[msg.sender] = initialSupply;          // Give the creator all initial tokens
    }

    /* Send coins */
    function transfer(address _to, uint256 _value) public returns (bool success) {
        require(balanceOf[msg.sender] >= _value);        // Check if the sender has enough
        require(balanceOf[_to] + _value >= balanceOf[_to]); // Check for overflows
        balanceOf[msg.sender] -= _value;                  // Subtract from the sender
        balanceOf[_to] += _value;                          // Add the same to the recipient
        return true;
    }
}

```

Programmierung & Tools

Ein einfacher Contract

608060405234801561001057600080fd5b506000808190555033600160006101000a81548173ffff
ff021916908373ffffffffffffffffffffffffffffffff
ffffffff1602179055506101be806100686000396000f30060806040526004361061004c57600035
7c01000900463ffffff168063
3ccfd60b146100515780638f9595d414610068575b600080fd5b34801561005d57600080fd5b5061
0066610086565b005b61007061011d565b6040518082815260200191505060405180910390f35b60
0160009054906101000a900473ff1673ffffffffffff
ffffffffffffffffffffffffffffffff163373ffffffffffffffffffffffffffffffff1614
15156100e257600080fd5b600160009054906101000a900473ffffffffffffffffffffffffffff
ffffffff1673ffffffffffffffffffffffff16ff5b60006001600090549061
01000a900473ffffffffffffffffffffffff1673ffffffffffffffffffff
ffffffff163373ffffffffffffffff1614151561017b5760
0080fd5b3460008082825401925050819055506000549050905600a165627a7a72305820b4d2d5a2
acfa0e197908c7c57826d35a987f64b29f8b97020a4ca96f63ce66630029

Ein einfacher Contract

A black silhouette of a cat, likely a Siamese cat, is positioned in the center of the frame. The cat is facing right, with its head turned slightly towards the viewer, showing its mouth open in a meow or purr. The background is a light gray color with a pattern of black binary code (0s and 1s) scattered across it. The cat's silhouette is solid black, contrasting sharply with the binary background.

Solidity

- populärste Hochsprache für EVM-Bytecode
- getypt, objektorientiert
- syntaktisch an Javascript angelehnt



```
pragma solidity ^0.4.22;
contract Wallet {
    uint256 balance;
    address owner;
    constructor () public {
        balance = 0;
        owner = msg.sender;
    }
    function addfund() payable public returns (uint256) {
        require (msg.sender == owner);
        balance += msg.value;
        return balance;
    }
    function withdraw() public {
        require (msg.sender == owner);
        selfdestruct(owner);
    }
}
```

```
pragma solidity ^0.4.22;
contract Wallet {
    uint256 balance;
    address owner;
    constructor () public {
        balance = 0;
        owner = msg.sender;
    }
    function addfund() payable public returns (uint256) {
        require (msg.sender == owner);
        balance += msg.value;
        return balance;
    }
    function withdraw() public {
        require (msg.sender == owner);
        selfdestruct(owner);
    }
}
```

```
pragma solidity ^0.4.22;
contract Wallet {
    uint256 balance;
    address owner;
    constructor () public {
        balance = 0;
        owner = msg.sender;
    }
    function addfund() payable public returns (uint256) {
        require (msg.sender == owner);
        balance += msg.value;
        return balance;
    }
    function withdraw() public {
        require (msg.sender == owner);
        selfdestruct(owner);
    }
}
```

```
pragma solidity ^0.4.22;
contract Wallet {
    uint256 balance;
    address owner;
    constructor () public {
        balance = 0;
        owner = msg.sender;
    }
    function addfund() payable public returns (uint256) {
        require (msg.sender == owner);
        balance += msg.value;
        return balance;
    }
    function withdraw() public {
        require (msg.sender == owner);
        selfdestruct(owner);
    }
}
```

```
pragma solidity ^0.4.22;
contract Wallet {
    uint256 balance;
    address owner;
    constructor () public {
        balance = 0;
        owner = msg.sender;
    }
    function addfund() payable public returns (uint256) {
        require (msg.sender == owner);
        balance += msg.value;
        return balance;
    }
    function withdraw() public {
        require (msg.sender == owner);
        selfdestruct(owner);
    }
}
```

```
pragma solidity ^0.4.22;
contract Wallet {
    uint256 balance;
    address owner;
    constructor () public {
        balance = 0;
        owner = msg.sender;
    }
    function addfund() payable public returns (uint256) {
        require (msg.sender == owner);
        balance += msg.value;
        return balance;
    }
    function withdraw() public {
        require (msg.sender == owner);
        selfdestruct(owner);
    }
}
```

```
pragma solidity ^0.4.22;
contract Wallet {
    uint256 balance;
    address owner;
    constructor () public {
        balance = 0;
        owner = msg.sender;
    }
    function addfund() payable public returns (uint256) {
        require (msg.sender == owner);
        balance += msg.value;
        return balance;
    }
    function withdraw() public {
        require (msg.sender == owner);
        selfdestruct(owner);
    }
}
```

```
pragma solidity ^0.4.22;
contract Wallet {
    uint256 balance;
    address owner;
    constructor () public {
        balance = 0;
        owner = msg.sender;
    }
    function addfund() payable public returns (uint256) {
        require (msg.sender == owner);
        balance += msg.value;
        return balance;
    }
    function withdraw() public {
        require (msg.sender == owner);
        selfdestruct(owner);
    }
}
```

```
pragma solidity ^0.4.22;
contract Wallet {
    uint256 balance;
    address owner;
    constructor () public {
        balance = 0;
        owner = msg.sender;
    }
    function addfund() payable public returns (uint256) {
        require (msg.sender == owner);
        balance += msg.value;
        return balance;
    }
    function withdraw() public {
        require (msg.sender == owner);
        selfdestruct(owner);
    }
}
```

web3.js



- official JavaScript API for Ethereum
- available for the browser and Node.js

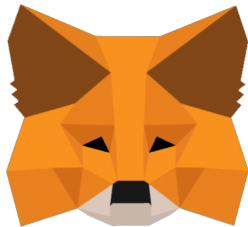
web3.js



```
var contract = new web3.eth.Contract(
  abi,
  {data: codeHex, from: web3.eth.defaultAccount });
var deployTransaction = contract.deploy({data: codeHex, arguments: []});
web3.eth.personal.unlockAccount(web3.eth.defaultAccount, "user")
  .then(() => deployTransaction.estimateGas())
  .then(gas =>
    deployTransaction.send({gasLimit: gas, from: web3.eth.defaultAccount}))
  .then(contract =>
    console.log("Address of new contract: " + contract.options.address))
  .catch(err => console.log(err));
```

Metamask

- Browser-Plugin zur Verwaltung von Ethereum-Adressen
- kommuniziert über RPC mit Blockchain-Knoten
- integriert mit web3.js



browser

config

```
1 pragma solidity >=0.4.22 <0.6;
2
3 contract Wallet {
4     uint256 balance;
5     address payable owner;
6     constructor () public {
7         balance = 0;
8         owner = msg.sender;
9     }
10    function addfund() payable public returns (uint256) {
11        require (msg.sender == owner);
12        balance += msg.value;
13        return balance;
14    }
15    function withdraw() public {
16        require (msg.sender == owner);
17        selfdestruct(owner);
18    }
19 }
20
```

Current version:0.5.5+commit.47a71e8f.Emscripten.clang

Select new compiler version

☒ Auto compile☐ Enable Optimization☐ Hide warnings

Start to compile (Ctrl-S)

Wallet

Swarm

Details

ABI

Bytecode

Static Analysis raised 2 warning(s) that requires your attention.

Click here to show the warning(s).

Wallet

[2] only remix transactions, sc... Search transactions

- Checking transactions details and start debugging.
- Running JavaScript scripts. The following libraries are accessible:
 - [web3_version 1.0.0](#)
 - [ethers.js](#)
 - [swarmgw](#)
 - compilers - contains currently loaded compiler
- Executing common command to interact with the Remix interface (see list of commands above). Note that these commands can also be included and run from a JavaScript script.
- Use `exports.register(key, obj).remove(key).clear()` to register and reuse object across script executions.

```
1 pragma solidity >=0.4.22 <0.6;
2
3 contract Wallet {
4     uint256 balance;
5     address payable owner;
6     constructor () public {
7         balance = 0;
8         owner = msg.sender;
9     }
10    function addfund() payable public returns (uint256) {
11        require (msg.sender == owner);
12        balance += msg.value;
13        return balance;
14    }
15    function withdraw() public {
16        require (msg.sender == owner);
17        selfdestruct(owner);
18    }
19 }
20 }
```

[2] only remix transactions,

- [web3 version 1.0.0](#)
- [ethers.js](#)
- [swarmgw](#)
- compilers - contains currently loaded compilers
- Executing common command to interact with the contract these commands can also be included and run
- Use exports/.register(key, obj)/.remove(key)/.clear()

creation of Wallet pending...

moz-extension://acaf48e3-a854...otification - Mozilla Firefox

Kovan Test Network

Test → New Contract

CONTRACT DEPLOYMENT

0.00

DETAILS DATA

GAS FEE 0.000194 \$0.03

AMOUNT + GAS FEE

TOTAL 0.000194 \$0.03

EDIT

REJECT CONFIRM

Environment Injected Web3 Kovan (42) i

Account + 0x1c9...231a3 (2.99813089 ether) i

Gas limit 3000000

Value 0 wei

Wallet i

Deploy

or

At Address Load contract from Address

Transactions recorded: 1

Deployed Contracts

Currently you have no contract instances to interact with.

```
1 pragma solidity >=0.4.22 <0.6;
2
3 contract Wallet {
4     uint256 balance;
5     address payable owner;
6     constructor () public {
7         balance = 0;
8         owner = msg.sender;
9     }
10    function addfund() payable public returns (uint256) {
11        require (msg.sender == owner);
12        balance += msg.value;
13        return balance;
14    }
15    function withdraw() public {
16        require (msg.sender == owner);
17        selfdestruct(owner);
18    }
19 }
20 }
```

[2] only remix transactions,

- [web3 version 1.0.0](#)
- [ethers.js](#)
- [swarmgw](#)
- compilers - contains currently loaded compilers
- Executing common command to interact with the contract these commands can also be included and run
- Use exports.register(key, obj).remove(key) for custom actions.

creation of Wallet pending...

moz-extension://acaf48e3-a854...otification - Mozilla Firefox

Kovan Test Network

Test → New Contract

CONTRACT DEPLOYMENT

0
\$0.00

DETAILS DATA

GAS FEE 0.000194
\$0.03

AMOUNT + GAS FEE

TOTAL 0.000194
\$0.03

EDIT

REJECT CONFIRM

Environment Injected Web3 Kovan (42)

Account 0x1c9...231a3 (2.99813089 ether)

Gas limit 3000000

Value 0 wei

Deploy

or

At Address Load contract from Address

Transactions recorded: 1

Deployed Contracts

Currently you have no contract instances to interact with.

Metamask

web3.js

Remix IDE





Beispiele



Beispiele

Beispiele

- Multi-Signatur-Wallets

Der Staatspräsident des Landes Baden:



der Ministerpräsident des Landes Bayern:



der Senatspräsident der Hansestadt Bremen:



der 1. Bürgermeister der Hansestadt Hamburg:



der Ministerpräsident des Landes Hessen:



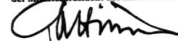
der Ministerpräsident des Landes Niedersachsen:



der Ministerpräsident des Landes Nordrhein-Westfalen:



der Ministerpräsident des Landes Rheinland-Pfalz:



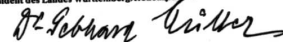
der Ministerpräsident des Landes Schleswig-Holstein:



der Ministerpräsident des Landes Württemberg-Baden:



der Ministerpräsident des Landes Württemberg-Hohenzollern:



Beispiele

- Multi-Signatur-Wallets
- ERC-20-Token



Beispiele

- Multi-Signatur-Wallets
- ERC-20-Token
- Handelsbörsen



Beispiele

- Multi-Signatur-Wallets
- ERC-20-Token
- Handelsbörsen
- Wetten



Beispiele

- Multi-Signatur-Wallets
- ERC-20-Token
- Handelsbörsen
- Wetten
- Cryptokitties



Komponenten einer Dapp

- Webapplikation mit JS (meist open source)
- Smart Contract(s)
- Blockchain



Softwarequalität



Entwicklungs-Workflow

Entwicklungs-Workflow

1. Contract in Solidity entwickeln

Entwicklungs-Workflow

1. Contract in Solidity entwickeln
2. mit Remix/Metamask ausrollen

Entwicklungs-Workflow

1. Contract in Solidity entwickeln
2. mit Remix/Metamask ausrollen
3. ???

- Smart Contracts sind **unveränderlich**

9

0800 Antan started
 1000 " stopped - antan ✓
 13⁰⁰ (033) MP-MC 2.130
 (033) PRO 2 2.130
 conv 2.130
 Relays 6-2 in 033 failed
 in Relay
 Relays changed
 1100 Started Cosine Tape (Si)
 1525 Started Multi Adder T
 1545 Relays (moth)
 First actual case of bu
 1630 Antan started.
 1700 closed down.

- Smart Contracts sind **unveränderlich**
- Transaktionen sind **endgültig**

9

0800 Antan started
 1000 " stopped - antan ✓
 13⁰⁰ (033) MP-MC 2.130
 (033) PRO 2 2.130
 conv 2.130
 Relays 6-2 in 033 fail
 in Relay
 Relays changed
 1100 Started Cosine Tape (Si
 1525 Started Mult+ Adder T
 1545 Relan
 (moth)
 First actual case of bu
 1630 Antan started.
 1700 closed down.

- Smart Contracts sind **unveränderlich**
- Transaktionen sind **endgültig**
- es gibt **keine Kontrollinstanz***

9

0800 Antan started
 1000 " stopped - antan ✓
 13⁰⁰ (033) MP-MC 1.982
 (033) PRO 2 2.130
 conv 2.130
 Relays 6-2 in 033 failed
 in Relay
 Relays changed
 1100 Started Cosine Tape (Si
 1525 Started Multi Adder To
 1545 Relays (moth)
 First actual case of bu
~~1630~~ 1630 Antan started.
 1700 closed down.

- Smart Contracts sind **unveränderlich**
- Transaktionen sind **endgültig**
- es gibt **keine Kontrollinstanz***
- Fehler sind **nicht korrigierbar***

9

0800 Antan started
 1000 " stopped - antan ✓
 13⁰⁰ (033) MP-MC 1.982
 (033) PRO 2 2.130
 conv 2.130
 Relays 6-2 in 033 failed
 in Relay
 Relays changed
 1100 Started Cosine Tape (Si
 1525 Started Multi Adder To
 1545  Relays
 (moth)
 First actual case of bu
~~1630~~ 1630 Antan started.
 1700 closed down.

Fatale Fehler



The DAO (2016)

- *Decentralized Autonomous Organization*
- eine Art digitaler Geber von Wagniskapital
- Idee: Investor*innen stimmen über Kapitalnutzung ab
- Bug im Contract führte zum Abgreifen von 50 Millionen USD

Fatale Fehler



Parity Wallet (2017)

- Online Wallet für Ether und andere Token
- Bug im Contract führte zum Abgreifen von 30 Millionen USD

Fatale Fehler



Parity Wallet (2017)

- Online Wallet für Ether und andere Token
- Bug im Contract führte zum Abgreifen von 30 Millionen USD
- weiterer Bug führte zum **Totalverlust von 360 Millionen USD**

Qualitätssicherung



Truffle: Framework für den gesamten Entwicklungszyklus

- Testing (Solidity und JavaScript)
- Deployment
- Migrations

Code is Law

“ However, these more complex agreements, with greater engagement with real world goods and services, highlight the necessity of effective dispute resolution, as well as indicate a necessary interrelation with territorial legal systems.

Code is Law

“ However, these more complex agreements, with greater engagement with real world goods and services, highlight the necessity of effective dispute resolution, as well as indicate a necessary interrelation with territorial legal systems.

Contestation mechanisms are necessary to 'soften' the effects of self-executing 'smart contracts', and make transactions reversible, allowing the outcomes of dispute resolution to be enforced. ”

Goldenfein & Later: "Legal Engineering on the Blockchain: 'Smart Contracts' as Legal Conduct" (2018)

Hürden für Smart Contracts

- bisher keine etablierten formalen Methoden
- unklare Jurisdiktion
- Nachvollziehbarkeit des Codes

Smart Contracts?

“ *There are only two things you need to know about smart contracts:*

- 1. They're not smart.*
- 2. They're not contracts.*

”

Philip Wadler (at Full Stack Fest: <https://twitter.com/steveklabnik/status/1169599735956672513>)



Varianten von Ethereum

Ethereum Classic



- *Hard Fork* im Juli 2016
- Ursache: Unstimmigkeit über Vorgehen nach DAO-Hack
- Platz 18 in Marktkapitalisierung

Proof of Work

- eingeführt durch Bitcoin
- Rückgrat der meisten Kryptowährungen
- basiert auf Investition von Strom



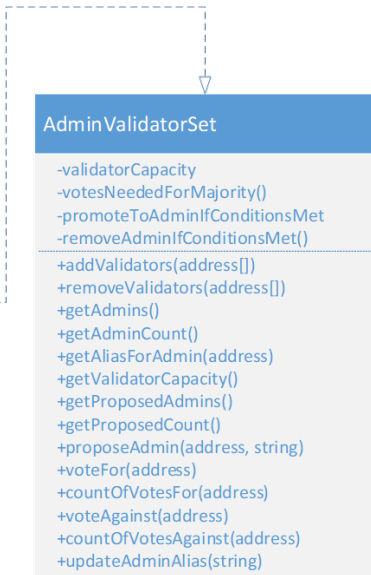
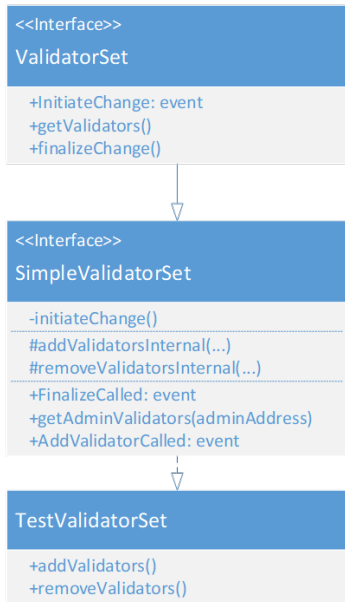
Proof of Stake

- experimentelle(s) Verfahren
- basiert auf Abstimmung über Blöcke
- Gewichtung nach *Stake*: z.B. Vermögen oder Vermögensalter
- als Ziel für Ethereum vorgesehen
- Umsetzung unklar

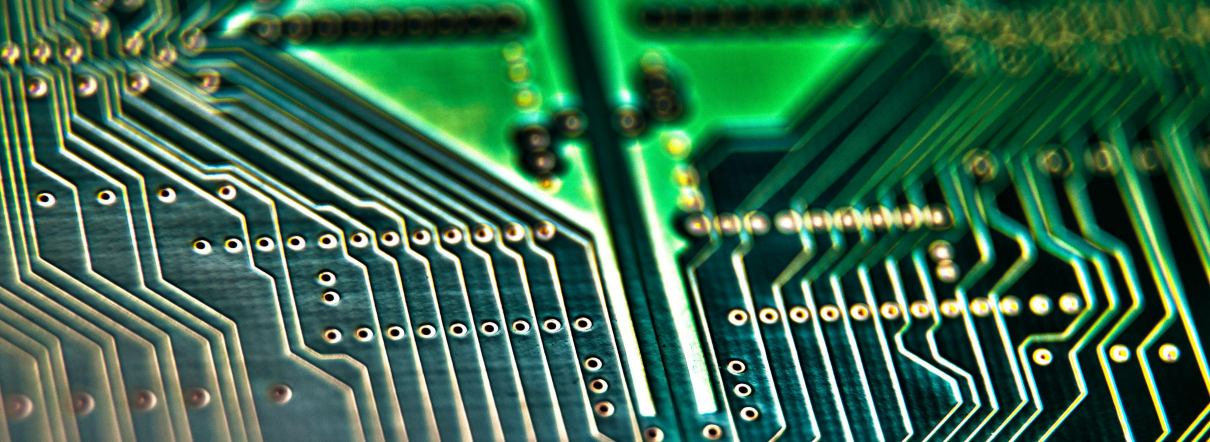
Proof of Authority

- ein Konsortium einigt sich über vertrauenswürdige Knoten
- abwechselnde Erzeugung von Blöcken
- keine Transaktionsgebühr nötig
- Konsortium kann selbst über Smart Contract gesteuert werden

BUNDESVERWALTUNGSGERICHT







Rust & WebAssembly

Rust & WebAssembly



Rust

- maschinennahe Sprache von Mozilla
- Fokus auf Speichersicherheit
- kein GC



WebAssembly

- Assemblersprache für den Browser
- läuft in der JS-Sandbox
- kein Zugriff auf den DOM

EWASM

- WebAssembly auf der EVM: vielversprechende Entwicklung
- derzeit noch unspezifiziert
- experimentelle Implementierung in Parity



```
#[eth_abi(CAEndpoint, CAClient)]
pub trait CAInterface {
    fn constructor(&mut self);

    #[constant]
    fn owner(&mut self) -> U256;
}

pub struct CAContract;

impl CAInterface for CAContract {
    fn constructor(&mut self) {
        OWNER.write_as(pwasm_ethereum::sender());
    }

    fn owner(&mut self) -> U256 {
        OWNER.read_as()
    }
}
```



Fazit



Fazit

- Ethereum ist bisher größte Plattform für Dapps
- hohe Entwicklungsgeschwindigkeit
- leidet an mangelhaftem Sprachdesign
- Proof of Authority und EVM vielversprechend, aber noch in den Kinderschuhen

Q & A

INNOQ
www.innoq.com

Lars Hupel

 lars.hupel@innoq.com

 [@larsr_h](https://twitter.com/larsr_h)

innoQ Deutschland GmbH

Krischerstr. 100
40789 Monheim a. Rh.
Germany
+49 2173 3366-0

Ohlauer Str. 43
10999 Berlin
Germany

Ludwigstr. 180 E
63067 Offenbach
Germany

Kreuzstr. 16
80331 München
Germany

c/o WeWork
Hermannstrasse 13
20095 Hamburg
Germany

innoQ Schweiz GmbH

Gewerbestr. 11
CH-6330 Cham
Switzerland
+41 41 743 01 11

Albulastr. 55
8048 Zürich
Switzerland



LARS HUPEL

Consultant
innoQ Deutschland GmbH

Lars enjoys programming in a variety of languages, including Scala, Haskell, and Rust. He is known as a frequent conference speaker and one of the founders of the Typelevel initiative which is dedicated to providing principled, type-driven Scala libraries.

Bildquellen

Decentralized Apps: <https://medium.com/swlh/understanding-dapps-decentralized-applications-8f3668ebdc9a>

Smart Contract: <http://instaco.de/>,
<https://www.ethereum.org/token>

Akten: <https://pixabay.com/photos/files-ddr-archive-1633406/>

Ethereum Transactions:

<https://medium.com/@preethikasireddy/how-does-ethereum-work-anyway-22d1df506369>

Platine:

https://commons.wikimedia.org/wiki/File:Computer_Circuit_Board_MOD_45153619.jpg,
Harland Quarrington, OGL 1.0

Gas Pump:

https://commons.wikimedia.org/wiki/File:Antique_yard_vintage_gas_pumps,_Washington_LCCN2010630650.tif, Library of Congress

Treaty of Rome:

https://en.wikipedia.org/wiki/File:Treaty_of_Rome.jpg, author unknown
Scared cat:

<https://pixabay.com/vectors/cat-scared-hunched-spooky-black-30788/>

Coins: <https://pixabay.com/photos/money-coins-euro-coins-currency-515058/>

Macaroons:

<https://pixabay.com/photos/macaroon-dessert-sweet-bakery-food-2815075/>

Power station:

<https://pixabay.com/photos/power-station-energy-electricity-374097/>

UN General Assembly: <https://www.flickr.com/photos/50371131@N04/29192350713>,

Gobierno de Chile, CC-BY

Bundesverwaltungsgericht:

<https://pixabay.com/photos/supreme-administrative-court-court-3687414/>

Ratifikation des Grundgesetzes: https://de.wikipedia.org/w/index.php?title=Datei:Grundgesetz_Ratifikationsunterschriften.jpg&filetimestamp=20110526174813&

NYSE: <https://pixabay.com/photos/stock-exchange-trading-floor-738671/>

Poker chips:

<https://pixabay.com/photos/chips-play-poker-casino-gambling-2038348/>

Contract UML: <https://github.com/Azure-Samples/blockchain/blob/master/ledger/template/ethereum-on-azure/permissioning-contracts/validation-set/media/contract-uml.png>

Server rack:

<https://pixabay.com/photos/server-cloud-development-business-1235959/>

Lightning:

<https://pixabay.com/photos/lightning-storm-weather-sky-399853/>

Ferris the Crab:

<https://www.rust-lang.org/what/wasm>, MIT

Construction site:

<https://pixabay.com/photos/building-construction-site-cranes-768815/>

Railroad: <https://pixabay.com/photos/track-railway-line-rail-traffic-3670209/>