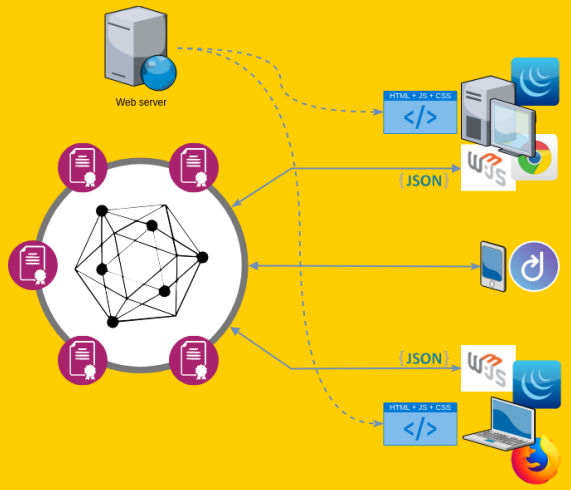
Blockchain architectures

From smart contracts to tokens and DApps

Utrecht University



Claudio Di Ciccio

c.diciccio@uu.nl

https://www.uu.nl/staff/CdiCiccio

Agenda for today

13:15 - 13:20:
Recap

13:20 - 14:00:
Smart Contracts and DApps

14:15 - 15:00:
Tokens and hands on Solidity

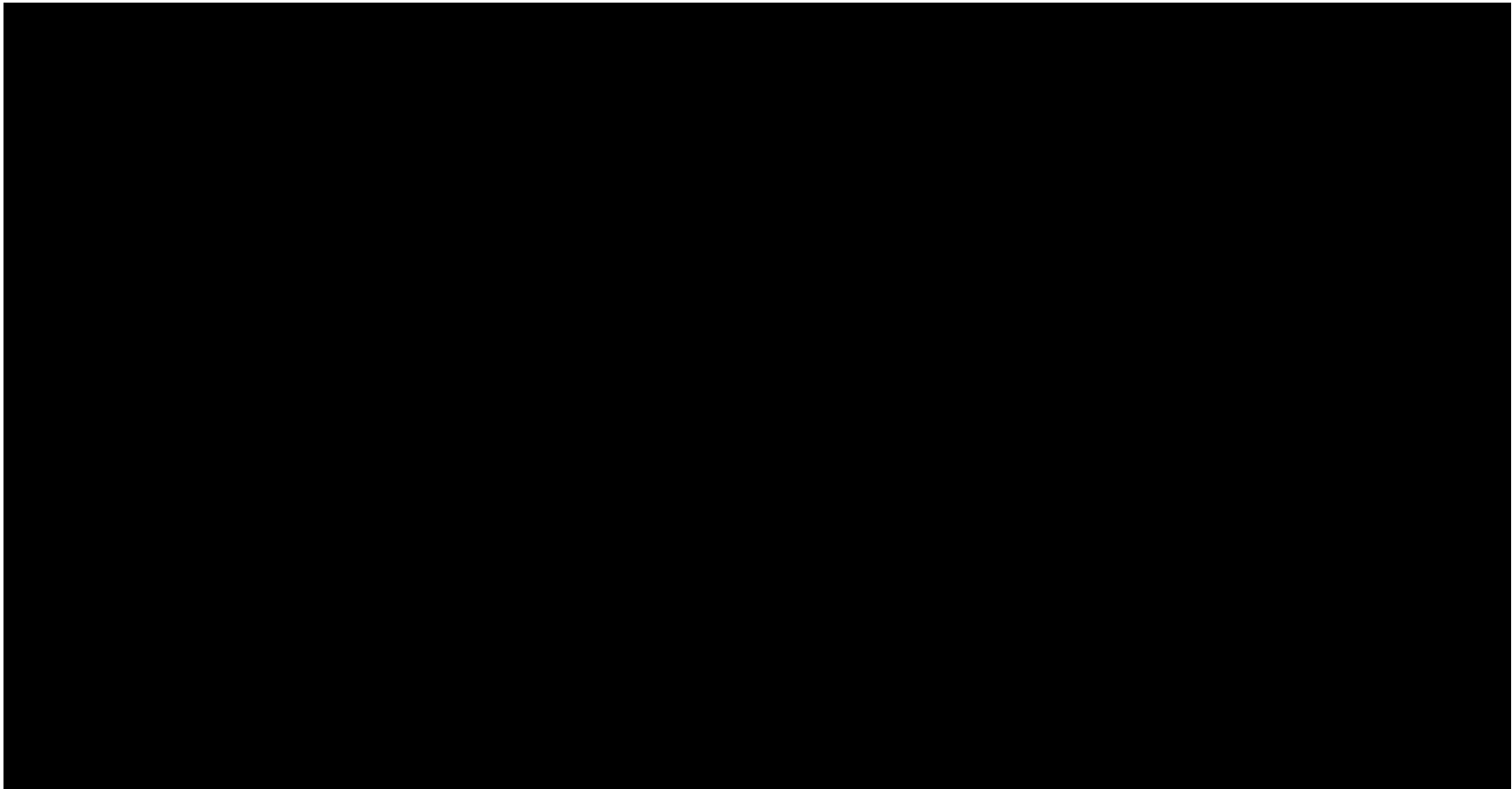
15:00 - 15:30:
Feedback

15:30 - 17:00:
Architectural debate

Kudos to Jan Martijn for the picture

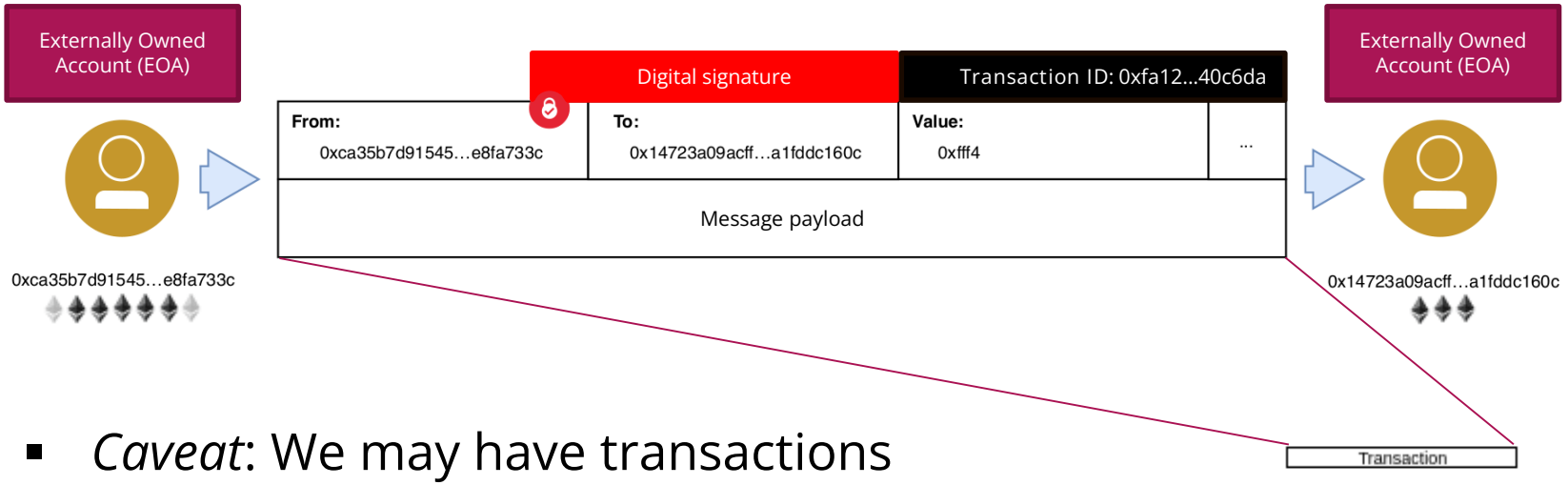
Utrecht University

Recap and Q&A



Transaction

- Transfer of (crypto)assets (Ether, Bitcoin, Litecoin, EOS, …) from account A to account B



- *Caveat*: We may have transactions
  from account A to accounts B, C and A itself
  or from account A to an unspecified/non-existing/burning-bin accounts

Metaphor: the cheque

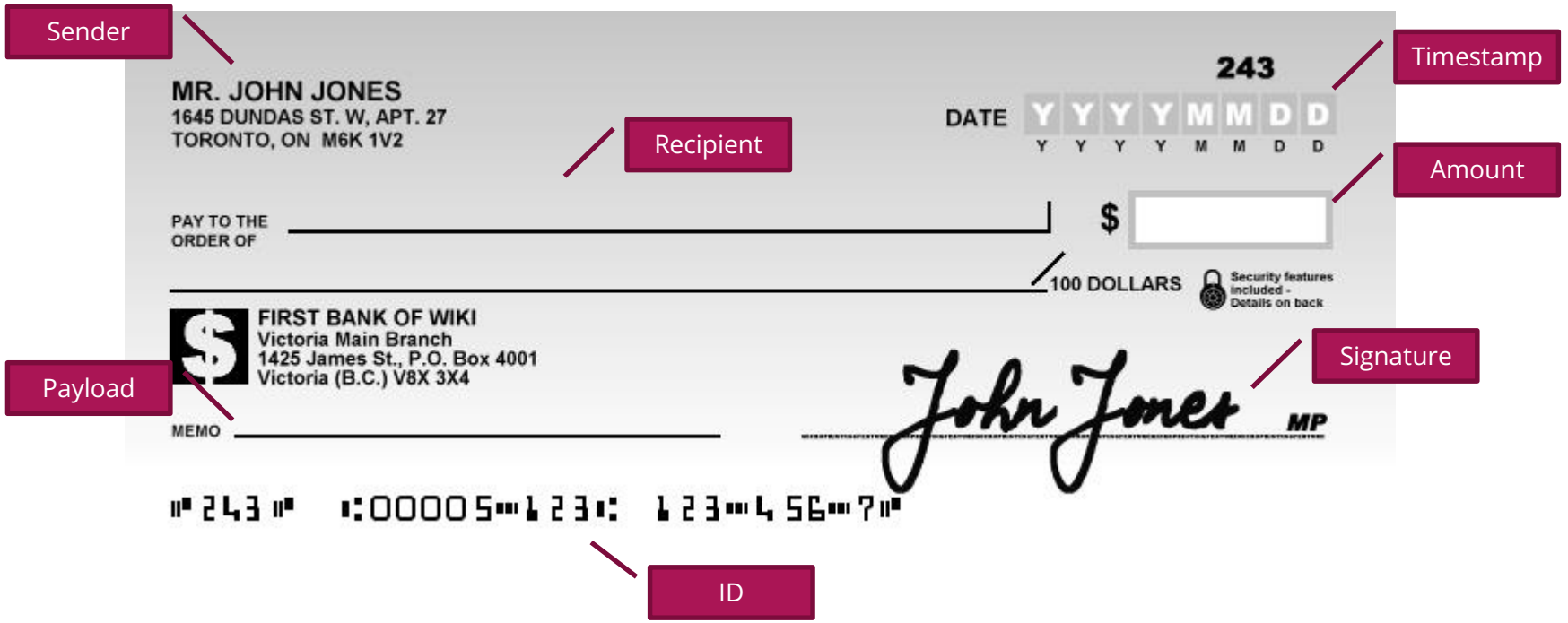


Image by Airodyssey, https://en.wikipedia.org/wiki/File:BlankCheque.png

Core concepts

- Transactions → Transfer of assets (and code invocation)
- Signatures → Authentication
- Ledger → Transaction ordering
- Distributed architecture → Data persistency
- Hashing → Robustness
- Proof-of-[…] → Publishing rights
- Consensus → Eventual consistency
- Smart contracts → Programmability (and tokens / app coins)

Smart Contracts are ~~codified autonomous agents~~ pieces of code[1]

```sol
// SPDX-License-Identifier: CC-BY-SA-4.0
pragma solidity >=0.8.0 <0.9.0;

contract HelloToken {
    address public minter;  // The creator of the contract instance
    mapping (address => uint) public balances;  // The balances in Hello-Tokens
    uint public constant PRICE = 2000000000;  // The price of a Hello Token (2 Gwei)

    constructor() {  // Deploys new instances of the smart contract
        minter = msg.sender;  // The sender is the creator
    }

    function mint() public payable {
        // Request the minimum amount for a Hello Token, or terminate
        require(msg.value >= PRICE, "Not enough value for a token!");
        // Add new Hello Tokens to the balance of the sender
        balances[msg.sender] += msg.value / PRICE;
        // The value of the transaction is acquired by the Smart Contract account
    }

    function transfer(uint amount, address to) public {
        require(balances[msg.sender] >= amount, "Not enough tokens!");
        // Decrese the amount from the sender
        balances[msg.sender] -= amount;
        // Increase the amount of Hello Tokens to a specified address
        balances[to] += amount;
    }

    function terminate() public {
        // Only the contract creator can terminate this instance
        require(msg.sender == minter, "You cannot terminate the contract!");
        // Terminate the contract instance and transfer the balance amount to the creator
        selfdestruct(payable(minter));
    }
}
```

- Smart Contracts in Ethereum
  - live in the Ethereum environment
  - execute a function when called
  - have direct control over their own balance and key/value storage
  - have their behaviour fully specified by their code

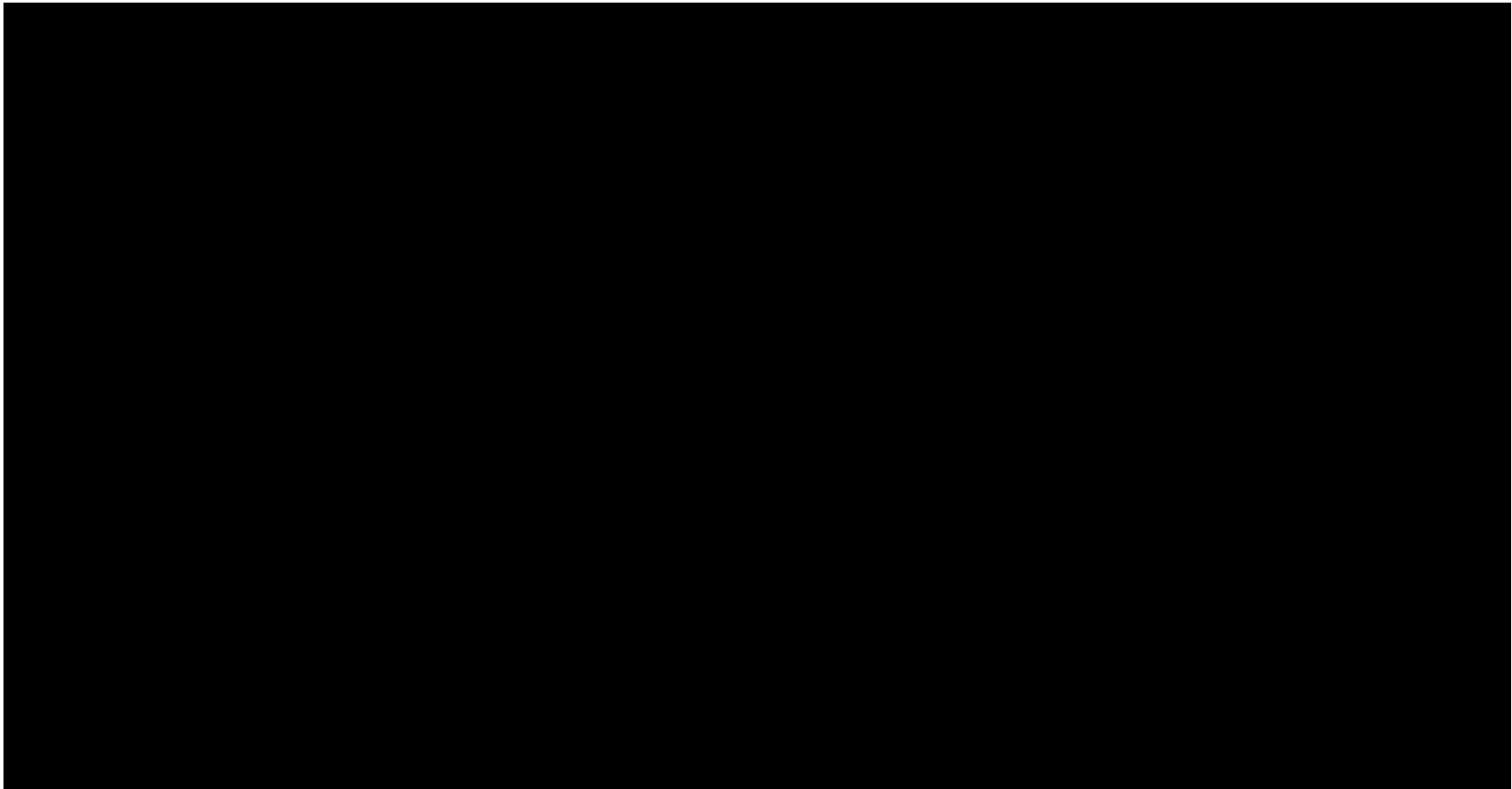
[1] Thanks to Prof Alexander Norta for the hint

From high-level code to bytecode to bits and bytes

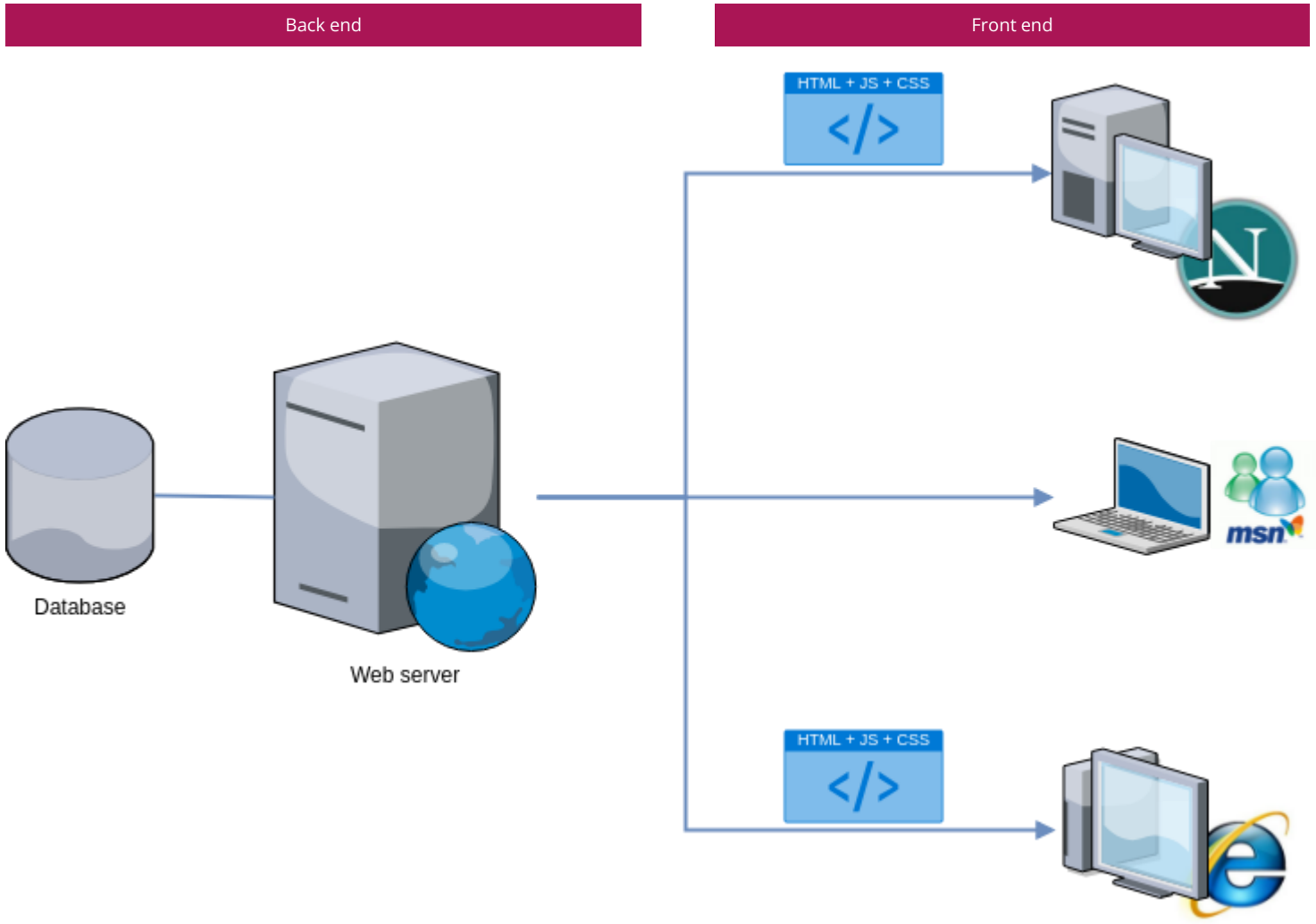
```solidity
// SPDX-License-Identifier: CC-BY-SA-4.0
pragma solidity >=0.8.0 <0.9.0;

contract HelloToken {
    address public minter;  // The creator of the contract instance
    mapping (address => uint) public balances;  // The balances in Hello-Tokens
    uint public constant PRICE = 2000000000;  // The price of a Hello Token (2 Gwei)

    constructor() {  // Deploys new instances of the smart contract
        minter = msg.sender;  // The sender is the creator
    }

    function mint() public payable {
        // Request the minimum amount for a Hello Token, or terminate
        require(msg.value >= PRICE, "Not enough value for a token!");
        // Add new Hello Tokens to the balance of the sender
        balances[msg.sender] += msg.value / PRICE;
        // The value of the transaction is acquired by the Smart Contract account
    }

    function transfer(uint amount, address to) public {
        require(balances[msg.sender] >= amount, "Not enough tokens!");
        // Decrese the amount from the sender
        balances[msg.sender] -= amount;
        // Increase the amount of Hello Tokens to a specified address
        balances[to] += amount;
    }

    function terminate() public {
        // Only the contract creator can terminate this instance
        require(msg.sender == minter, "You cannot terminate the contract!");
        // Terminate the contract instance and transfer the balance amount to the creator
        selfdestruct(payable(minter));
    }
}
```

Utrecht University

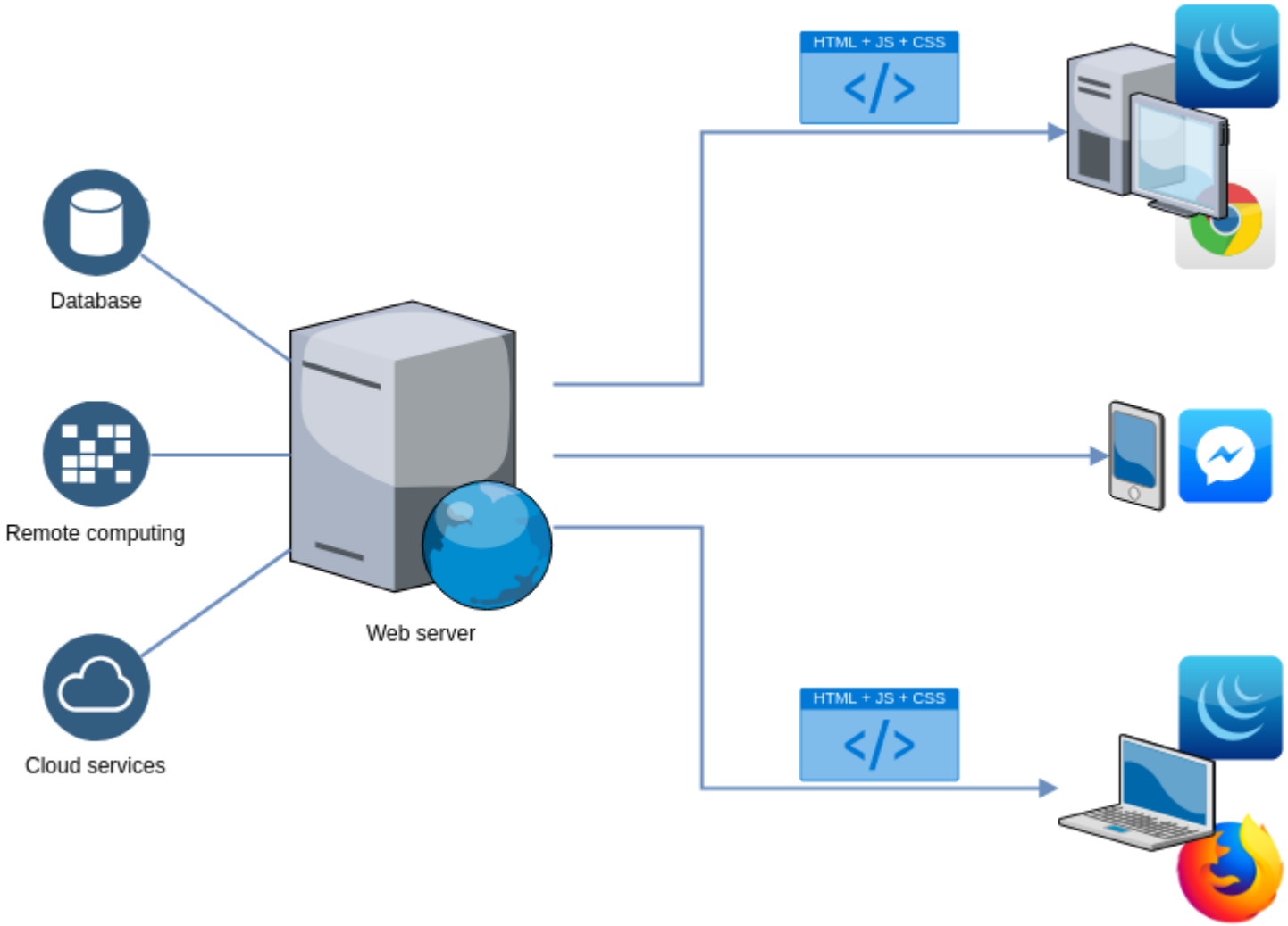
DApps



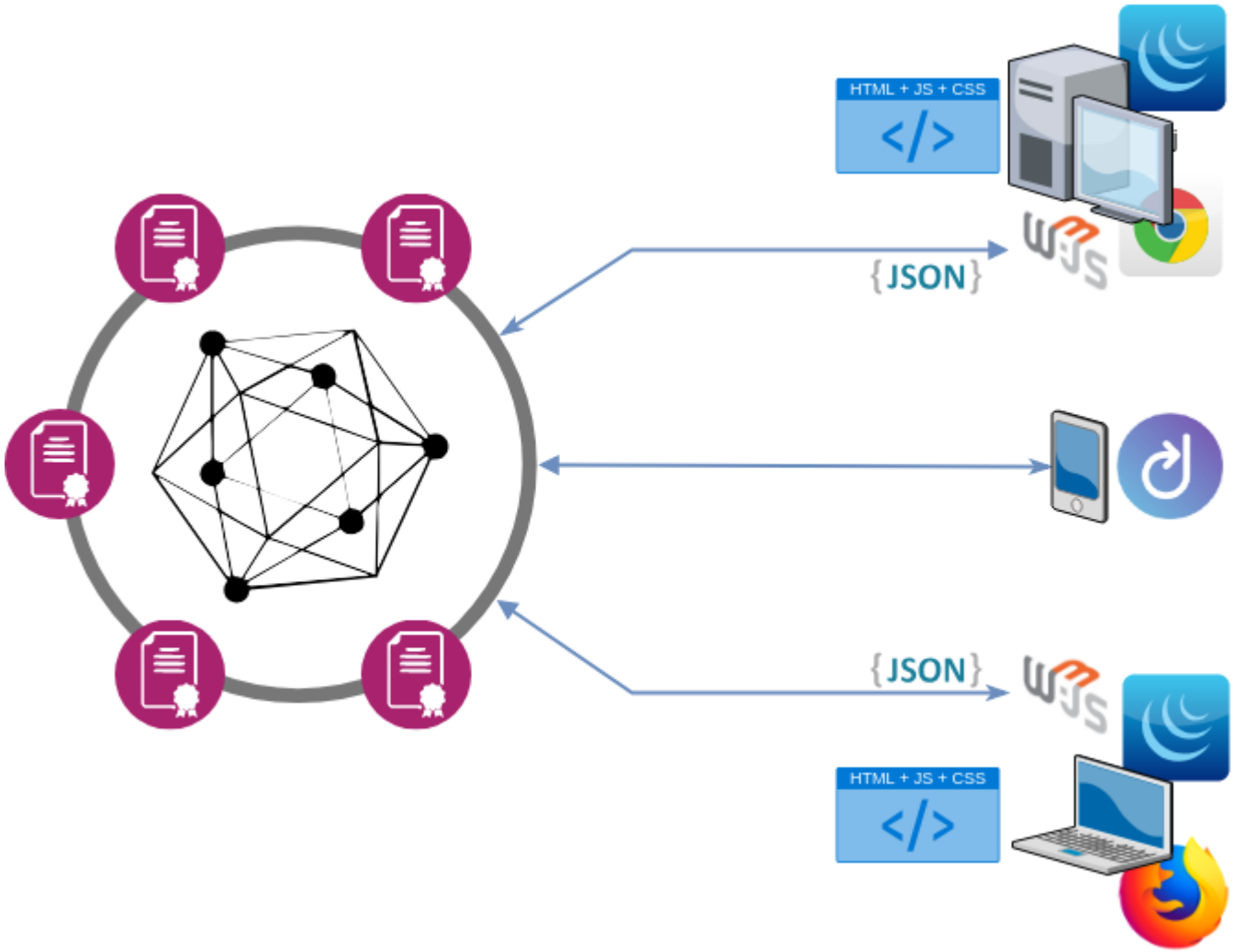
Web 1.0


Back end
Front end
HTML + JS + CSS
</>
Database
Web server
msn
HTML + JS + CSS
</>

Web 2.0

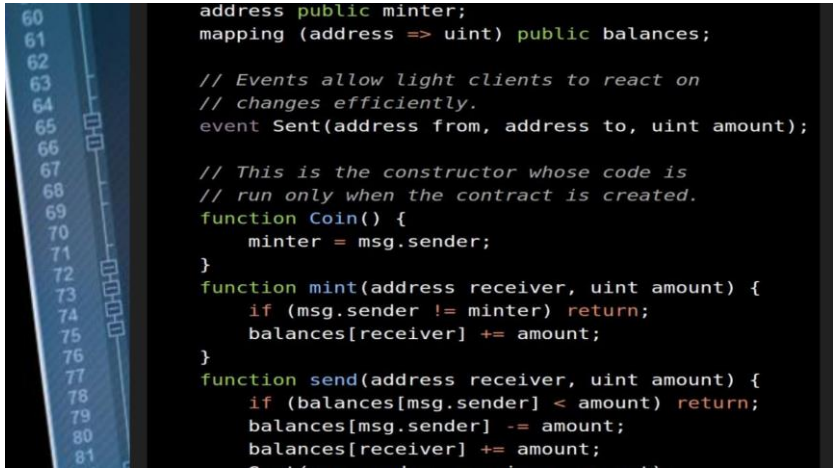

HTML + JS + CSS
Database
Remote computing
Cloud services
Web server
HTML + JS + CSS

Web 3.0



The Ethereum Virtual Machine (EVM)

- Think of the EVM as an emulation of a single, global "computer"
- A globally accessible virtual machine (like a mainframe)
  - in fact, lots of smaller computers
- There is a cost associated to the running of programs on the EVM
  - Please welcome the notion of gas



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

// Events allow light clients to react on
// changes efficiently.
event Sent(address from, address to, uint amount);

// This is the constructor whose code is
// run only when the contract is created.
function Coin() {
    minter = msg.sender;
}
function mint(address receiver, uint amount) {
    if (msg.sender != minter) return;
    balances[receiver] += amount;
}
function send(address receiver, uint amount) {
    if (balances[msg.sender] < amount) return;
    balances[msg.sender] -= amount;
    balances[receiver] += amount;
```

Table 3-2. *This is a complete list of EVM opcodes*

0s: Stop and Arithmetic Operations		
0x00	STOP	Halts execution.
0x01	ADD	Addition operation.
0x02	MUL	Multiplication operation.
0x03	SUB	Subtraction operation.
0x04	DIV	Integer division operation.
0x05	SDIV	Signed integer.
0x06	MOD	Modulo.
0x07	SMOD	Signed modulo.
0x08	ADDMOD	Modulo.
0x09	MULMOD	Modulo.
0x0a	EXP	Exponential operation.
0x0b	SIGNEXTEND	Extend length of 2s (complement signed integer).
10s: Comparison and Bitwise Logic Operations		
0x10	LT	Lesser-than comparison.
0x11	GT	Greater-than comparison.
0x12	SLT	Signed less-than comparison.
0x13	SGT	Signed greater-than comparison.
0x14	EQ	Equality comparison.

(*continued*)

Image on the top-right courtesy of BlockchainHub (blockchainhub.net). Image on the bottom-right corner: [IES]

Smart contracts are pieces of code (not for free)

Name	Value	Description
G_{zero}	0	Nothing paid for operations of the set W_{zero}.
G_{jumpdest}	1	Amount of gas to pay for a JUMPDEST operation.
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}.
$G_{\text{warmaccess}}$	100	Cost of a warm account or storage access.
$G_{\text{accesslistaddress}}$	2400	Cost of warming up an account with the access list.
$G_{\text{accessliststorage}}$	1900	Cost of warming up a storage with the access list.
$G_{\text{coldaccountaccess}}$	2600	Cost of a cold account access.
$G_{\text{coldsload}}$	2100	Cost of a cold storage access.
G_{sset}	20000	Paid for an SSTORE operation when the storage value is set to non-zero from zero.
G_{sreset}	2900	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.
$R_{\text{selfdestruct}}$	24000	Refund given (added into refund counter) for self-destructing an account.
$G_{\text{selfdestruct}}$	5000	Amount of gas to pay for a SELFDESTRUCT operation.
G_{create}	32000	Paid for a CREATE operation.
$G_{\text{codedeposit}}$	200	Paid per byte for a CREATE operation to succeed in placing code into state.
$G_{\text{callvalue}}$	9000	Paid for a non-zero value transfer as part of the CALL operation.
$G_{\text{callstipend}}$	2300	A stipend for the called contract subtracted from $G_{\text{callvalue}}$ for a non-zero value transfer.
$G_{\text{newaccount}}$	25000	Paid for a CALL or SELFDESTRUCT operation which creates an account.
G_{exp}	10	Partial payment for an EXP operation.
G_{expbyte}	50	Partial payment when multiplied by the number of bytes in the exponent 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.
$G_{\text{txdatazero}}$	4	Paid for every zero byte of data or code for a transaction.
$G_{\text{txdatanonzero}}$	16	Paid for every non-zero byte of data or code for a transaction.
$G_{\text{transaction}}$	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.
$G_{\text{keccak256}}$	30	Paid for each KECCAK256 operation.
$G_{\text{keccak256word}}$	6	Paid for each word (rounded up) for input data to a KECCAK256 operation.
G_{copy}	3	Partial payment for *COPY operations, multiplied by words copied, rounded up.
$G_{\text{blockhash}}$	20	Payment for each BLOCKHASH operation.

Wood , "Ethereum: A secure decentralised generalised transaction ledger" (2014), https://ethereum.github.com/yellowpaper/paper.pdf

The paradigm

Mainframe	Terminal

L: https://commons.wikimedia.org/wiki/File:SiemensHicom300_1200.jpg; R: https://commons.wikimedia.org/wiki/File:HP_2647A_terminal.jpg

Metaphor: the cheque

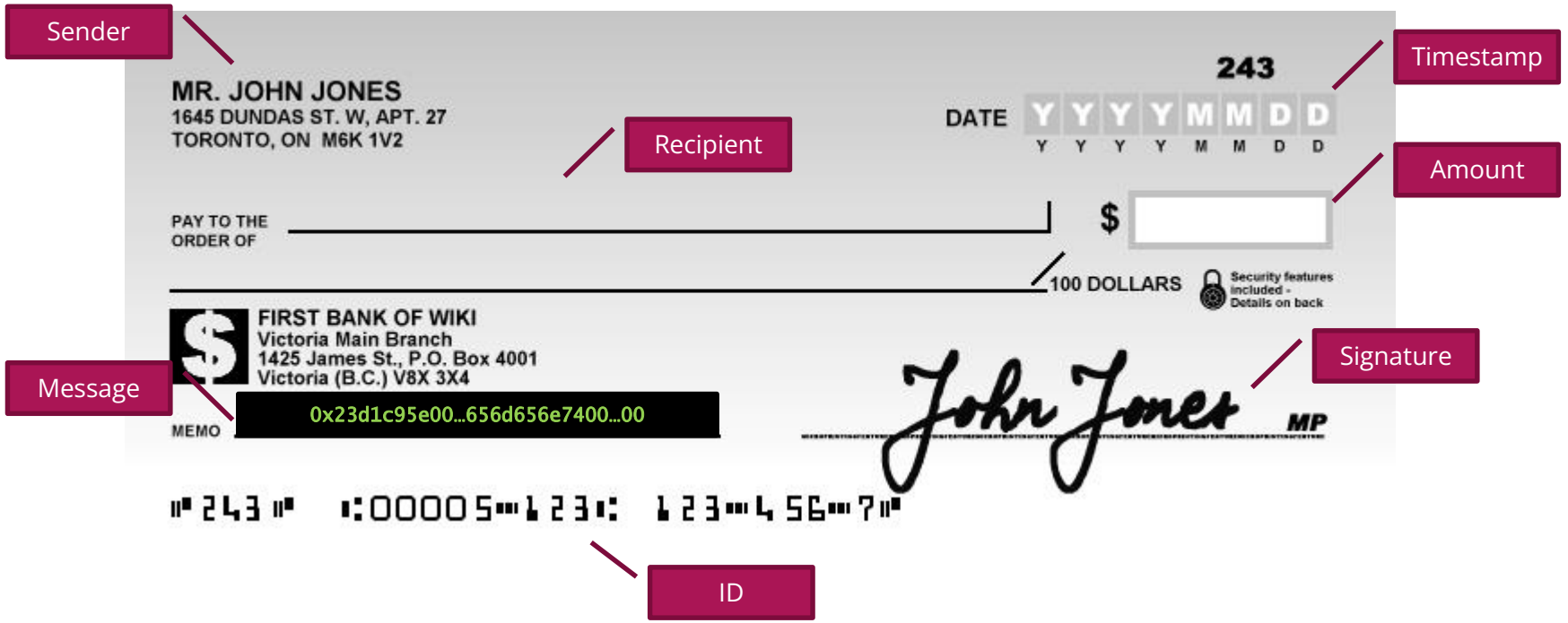


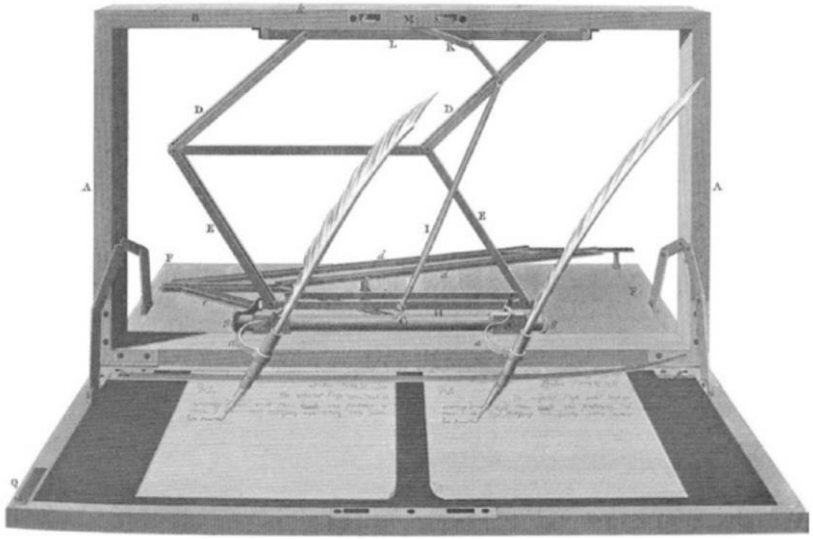
Image by Airodyssey, https://en.wikipedia.org/wiki/File:BlankCheque.png

The polygraph machine

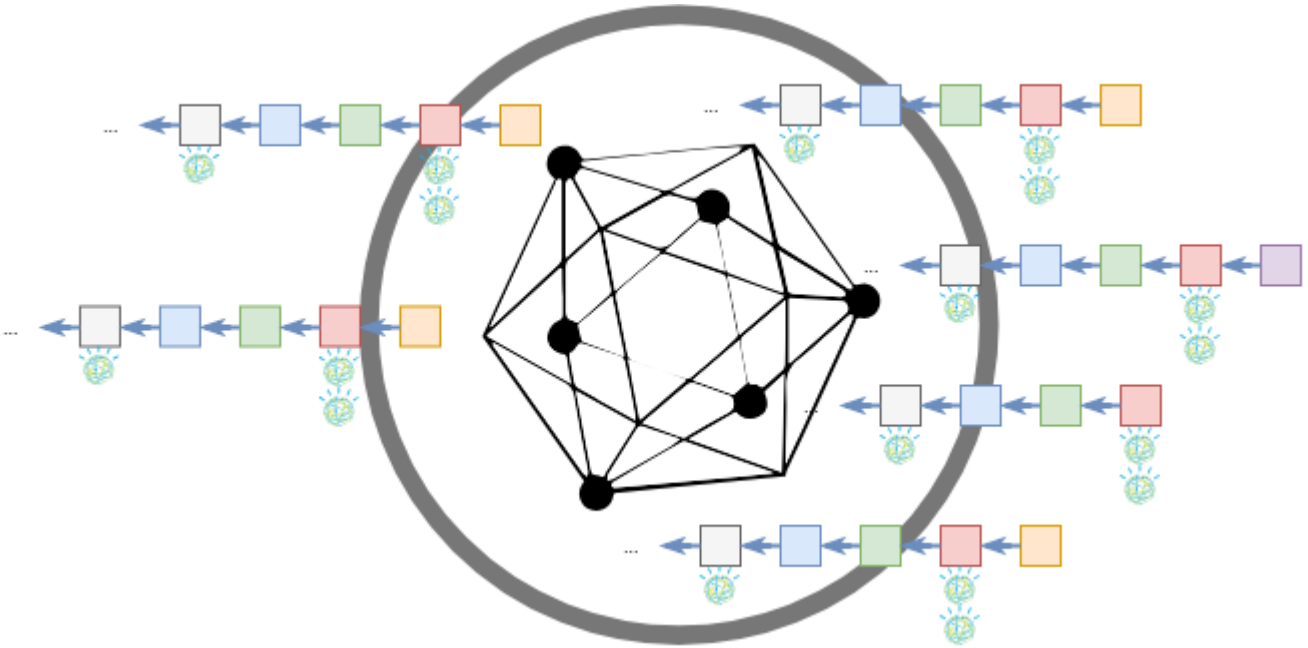
Where are Smart Contracts executed?

First on the mining nodes.
Then, potentially, on every node!

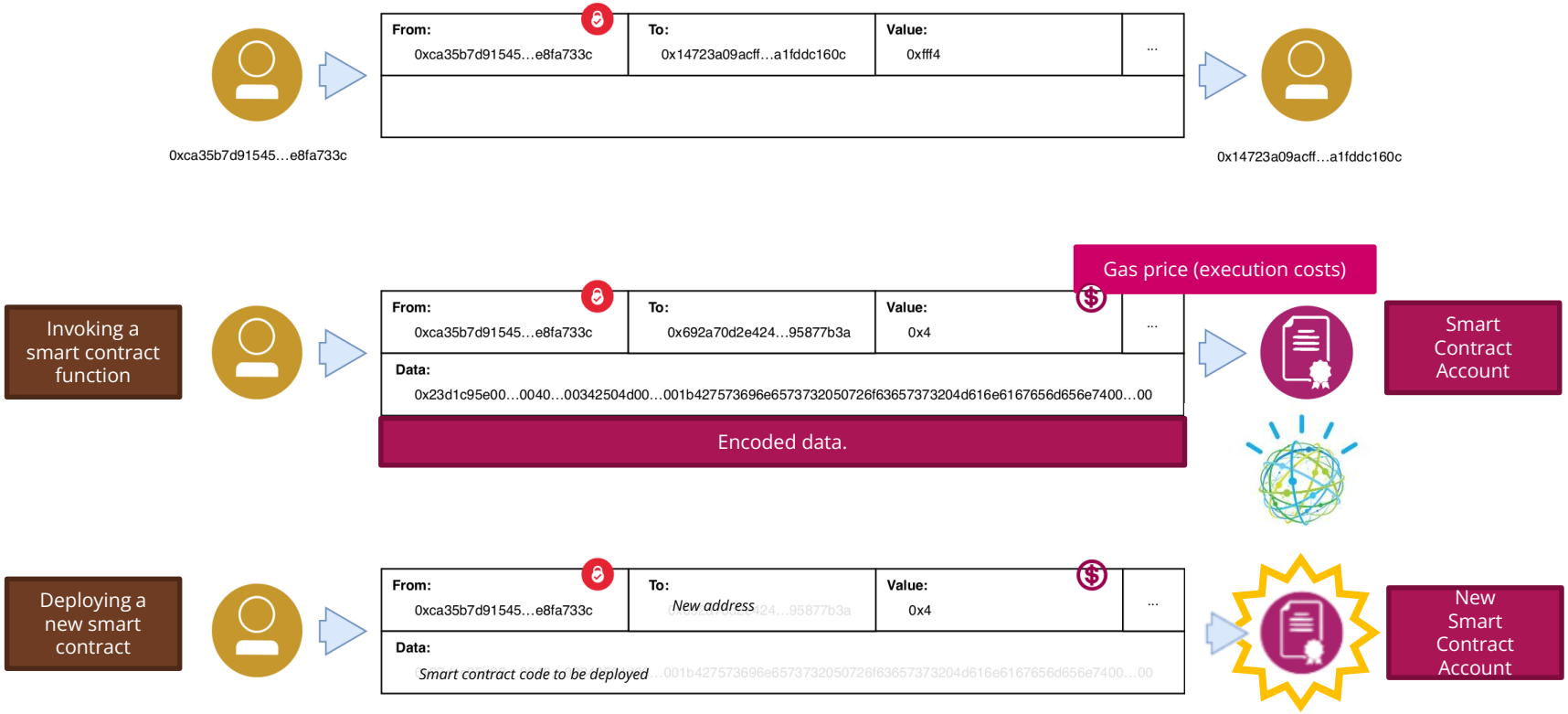
Only absolutely needed instructions should be in the code!



Distributed nature

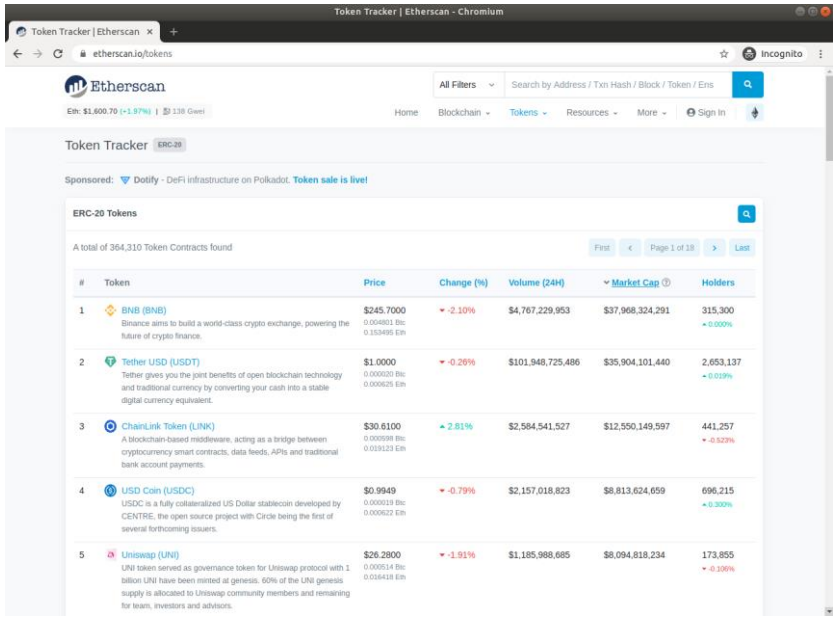


A programmable distributed environment



Smart contracts and tokens

```
pragma solidity ^0.4.0;

contract HelloToken {
    address public minter;
    mapping (address => uint) public balance;
    uint public constant PRICE = 2 finney;

    constructor() public {
        minter = msg.sender;
    }

    function mint() public payable {
        require(msg.value >= PRICE, "Not enough value for a token!");
        balance[msg.sender] += msg.value / 2 finney;
    }

    function transfer(uint amount, address to) public {
        require(balance[msg.sender] >= amount, "Not enough tokens!");
        balance[msg.sender] -= amount;
        balance[to] += amount;
    }

    function terminate() public {
        require(msg.sender == minter, "You cannot terminate the contract!");
        selfdestruct(minter);
    }
}

```



Token: A new asset class

! Notice you may find "*app coin*" as a synonym for token and "*app token*" for cryptocurrency

- Multiple functions:
  - Currency
    - Large-entity-backed medium of exchange for goods and services
  - Commodity
    - Basic good tradeable or exchangeable with goods of the same type
  - Utility
    - Satisfaction quantifier for an economic good or service
      - On the Merriam-Wesbter dictionary: https://www.merriam-webster.com/dictionary/utility%20token
  - Security
    - Financial instrument that guarantees ownership, credit, or decision power
- Different nature:
  - Fungible
    - Individual units can be mutually substituted
  - Non-fungible
    - Units are unique and not interchangeable
  - Semi-fungible
    - Fungible until redeemed or expired

More information on standards:
https://ethereum.org/en/developers/docs/standards/tokens/
https://eips.ethereum.org/EIPS/eip-20
https://eips.ethereum.org/EIPS/eip-721

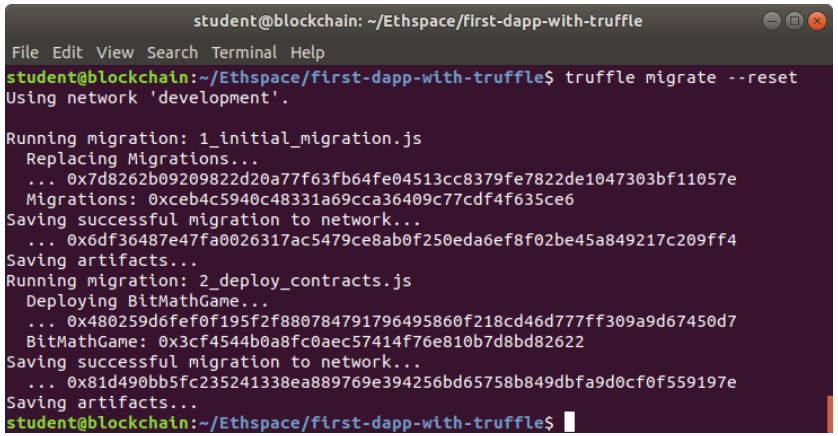
Reference implementations:
https://docs.openzeppelin.com/contracts/2.x/api/token/erc20
https://docs.openzeppelin.com/contracts/2.x/api/token/erc721

Some non-fungible tokens (NFTs)

- Crypto collectibles & games
- Property titles
  - Art
  - Collectibles
  - Memorabilia
  - Real Estate
  - Certificates
- Identity
- Keys & passes
- Fractional ownership of physical goods
- Purpose-driven, social-prescribing tokens
  - MCO2 token, TREE token, Plastic Bank token, …

Figures: https://www.mycryptoheroes.net/ (top) https://mco2token.moss.earth/ (bottom)

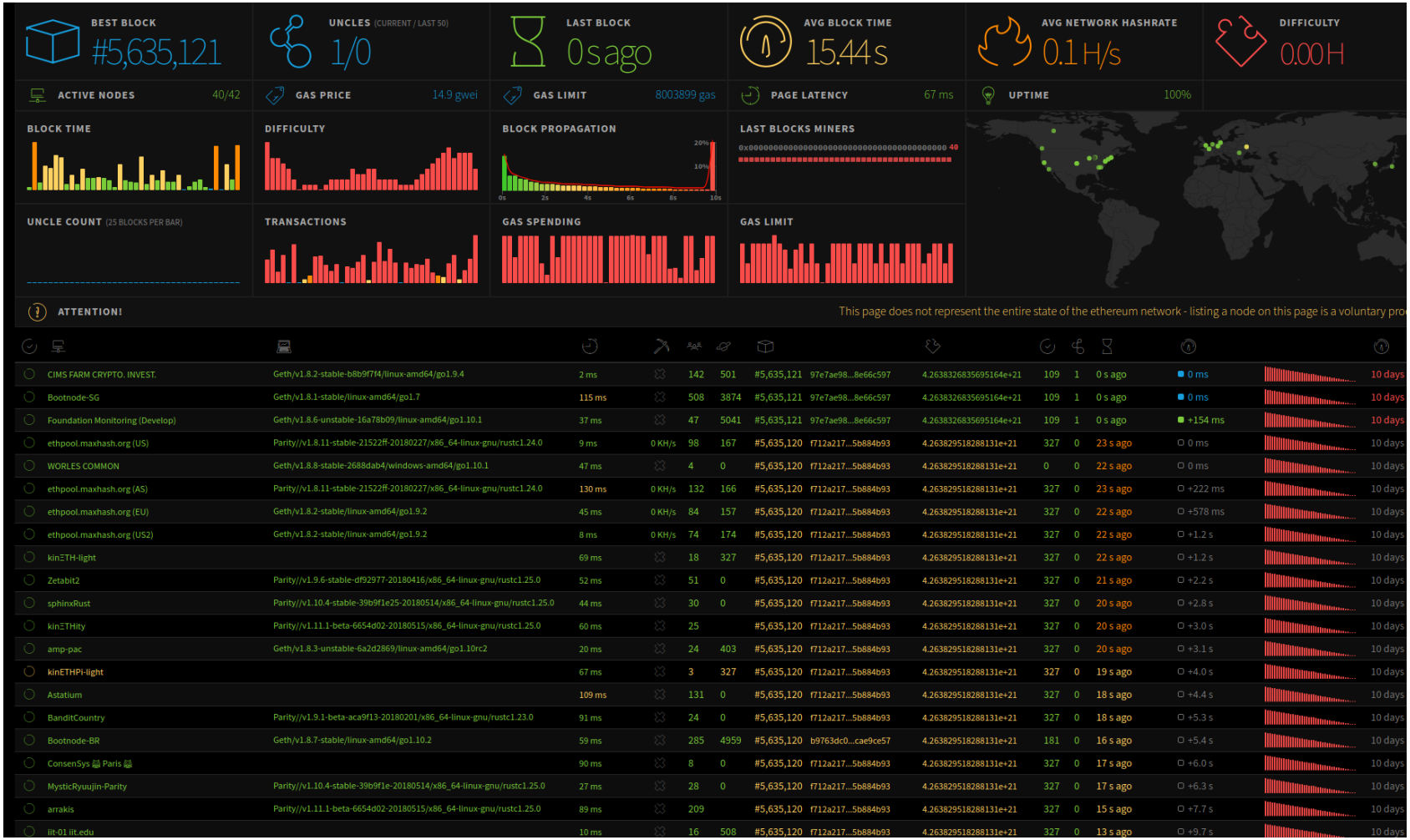
The Blockchain and the Internet



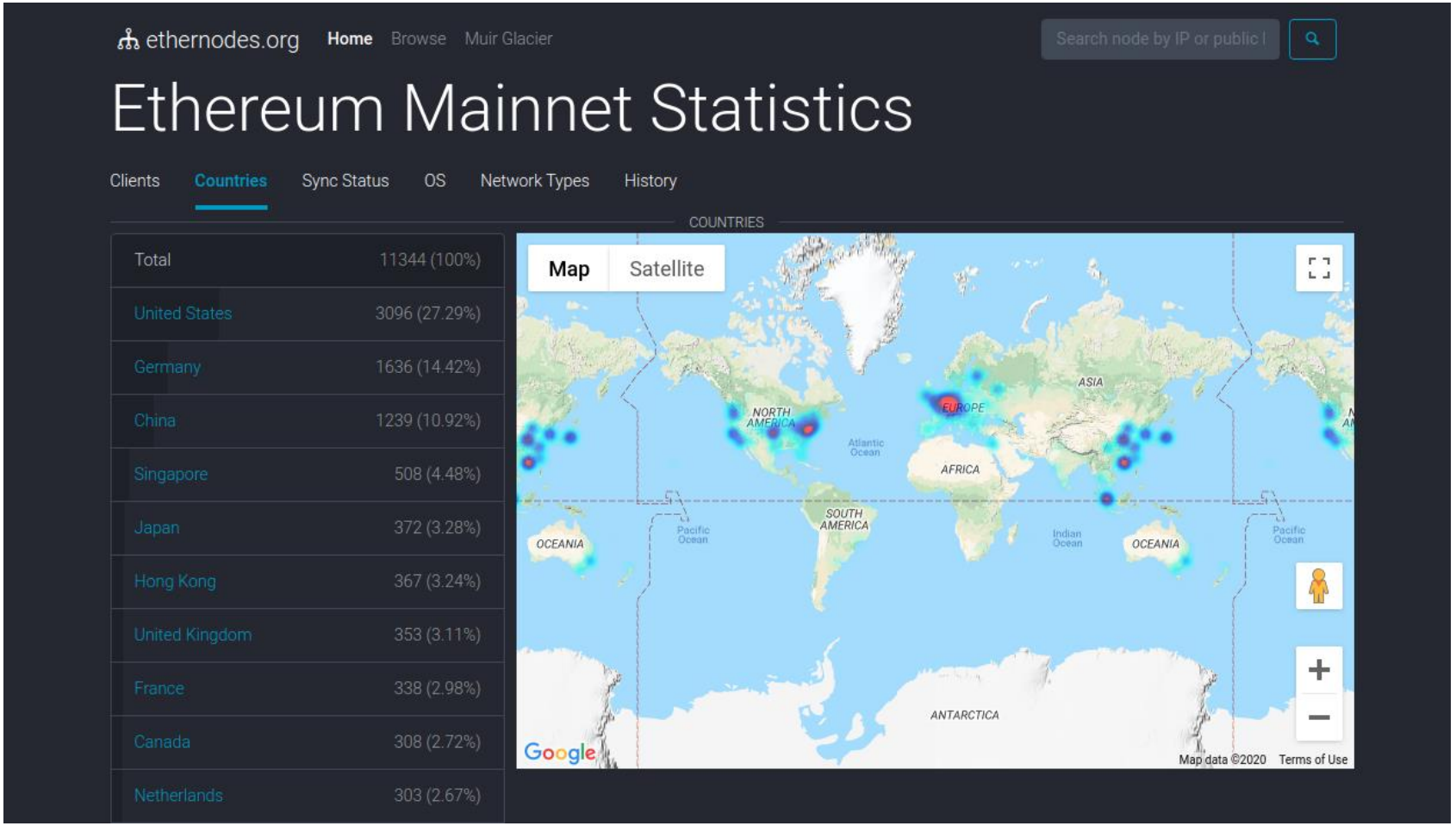
Decentralised Applications (DApps, or ĐApps)

- A DApp is a software program
- Its front-end runs on your computer system,
  but the back-end code runs on a decentralized peer-to-peer network
- In this course, we will only see DApps accessible from web browsers
- CryptoKitties is a DApp
- Dock.io is a DApp
- In the following two slides we have DApps
- Your teamwork's project is a (to-be) DApp

ethstats.net



ethernodes.org

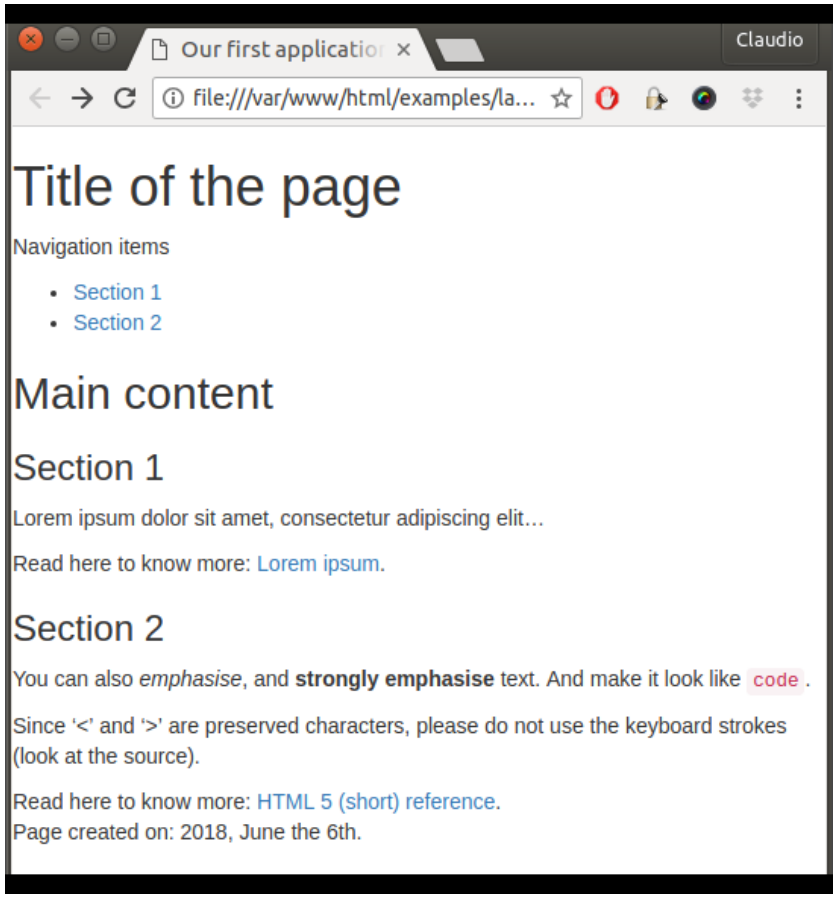


Hypertext Markup Language (HTML)

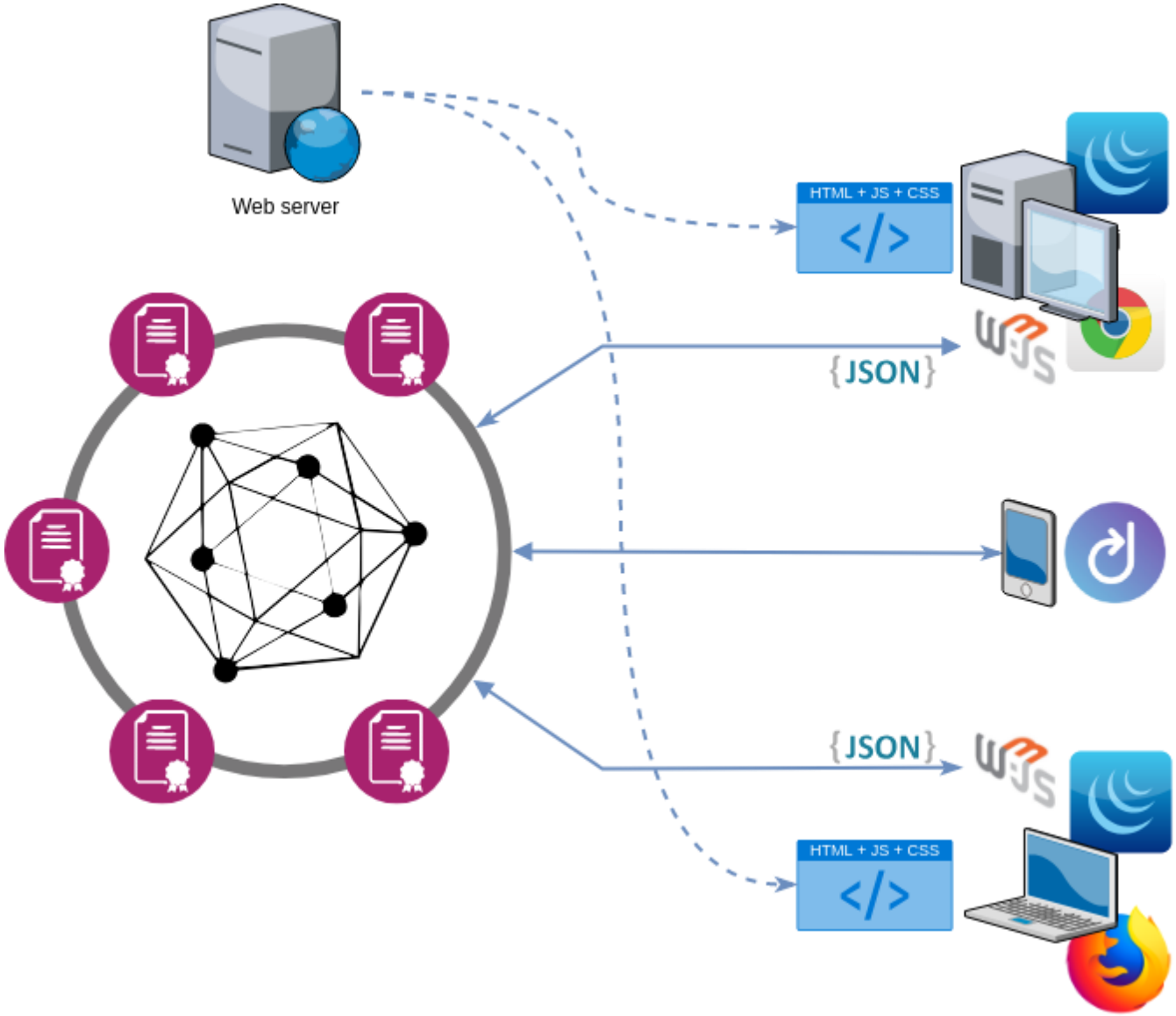
- The standard markup language for creating web pages
  - Based on XML (eXtensible Markup Language)
- It supports, among others:
  - Cascading Style Sheets (CSS)
    - To specify style and layout properties of HTML elements
    - Another W3C standard
  - Javascript (JS)
    - To dynamically change contents of the page
    - Not a standard: ECMA script is (https://www.ecma-international.org/publications/standards/Ecma-262.htm)
    - We will use jQuery to ease our life: http://api.jquery.com/
- Suggested links:
  - HTML
    - The W3C standard page for HTML (very detailed): https://www.w3.org/TR/html5/
    - Very nice, interactive tutorial: https://htmlreference.io/
  - CSS
    - Mozilla Development Network page for CSS: https://developer.mozilla.org/en-US/docs/Learn/CSS/Introduction_to_CSS

HTML example (file opened locally)

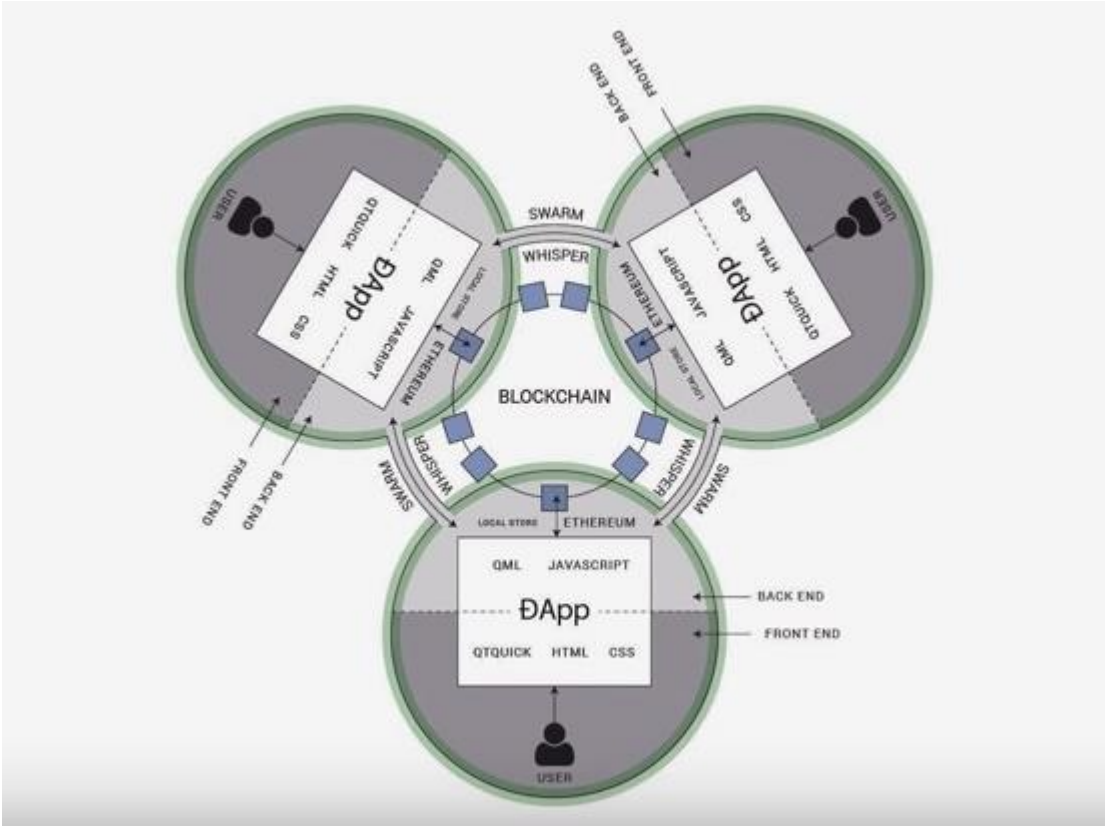
```html
<!DOCTYPE html>
<html lang="en">
  <meta charset="utf-8">
  <head>
      <title>A simple static web document</title>
      <!-- fancy styling of the page -->
      <link href="css/bootstrap.min.css" rel="stylesheet">
  </head>
  <body>
    <header>
      <h1>Title of the page</h1>
      <nav>
        <p>Navigation items</p>
        <ul>
          <li><a href="#section1">Section 1</a></li>
          <li><a href="#section2">Section 2</a></li>
        </ul>
      </nav>
    </header>
    <main>
      <h2>Main content</h2>
      <section id="section1">
        <header><h3>Section 1</h3></header>
        <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit…</p>
        <footer>Read here to know more:
          <a href="https://en.wikipedia.org/wiki/Lorem_ipsum">Lorem ipsum</a>.
        </footer>
      </section>
      <section id="section2">
        <header><h3>Section 2</h3></header>
        <p>You can also
          <em>emphasise</em>, and
          <strong>strongly emphasise</strong> text.
          And make it look like <code>code</code>.
        </p>
        <p>
          Since ‘&amp;lt;’ and ‘&amp;gt;’ are preserved characters, please
          do not use the keyboard strokes (look at the source).
        </p>
        <footer>Read here to know more:
          <a href="https://htmlreference.io/">HTML 5 (short) reference</a>.
        </footer>
      </section>
      <section id="section3">
        <header><h3>Section 3</h3></header>
        <p>This is the third Section, people!</p>
        <p>Hah, by the way, I own 1 wei.</p>
      </section>
    </main>
    <footer>
      Page created on:
      <time datetime="2020-12-02">2020, December the 2nd</time>.
    </footer>
  </body>
</html>
```



From where to retrieve HTML+CSS data?



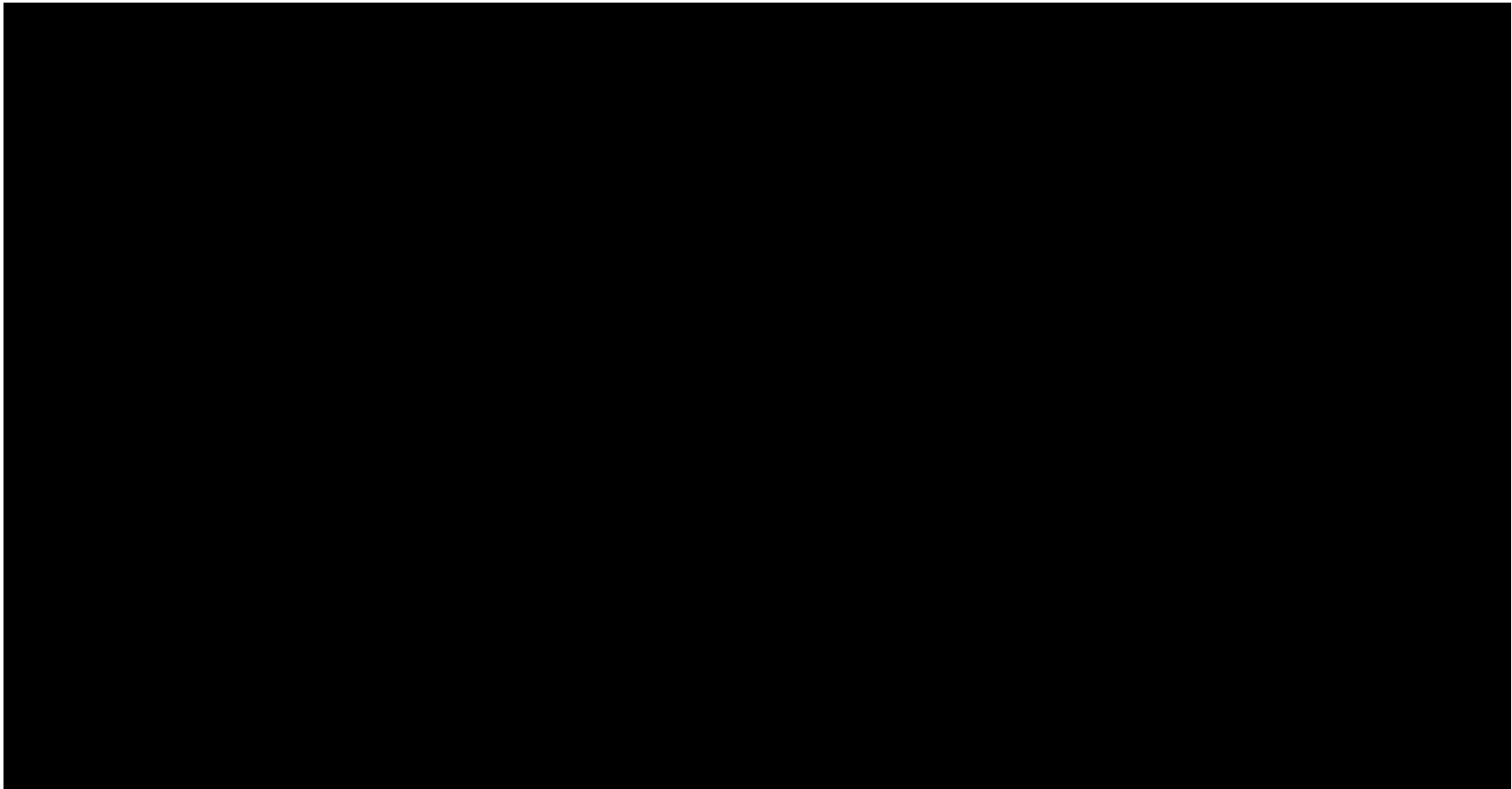
From where to retrieve HTML+CSS data?



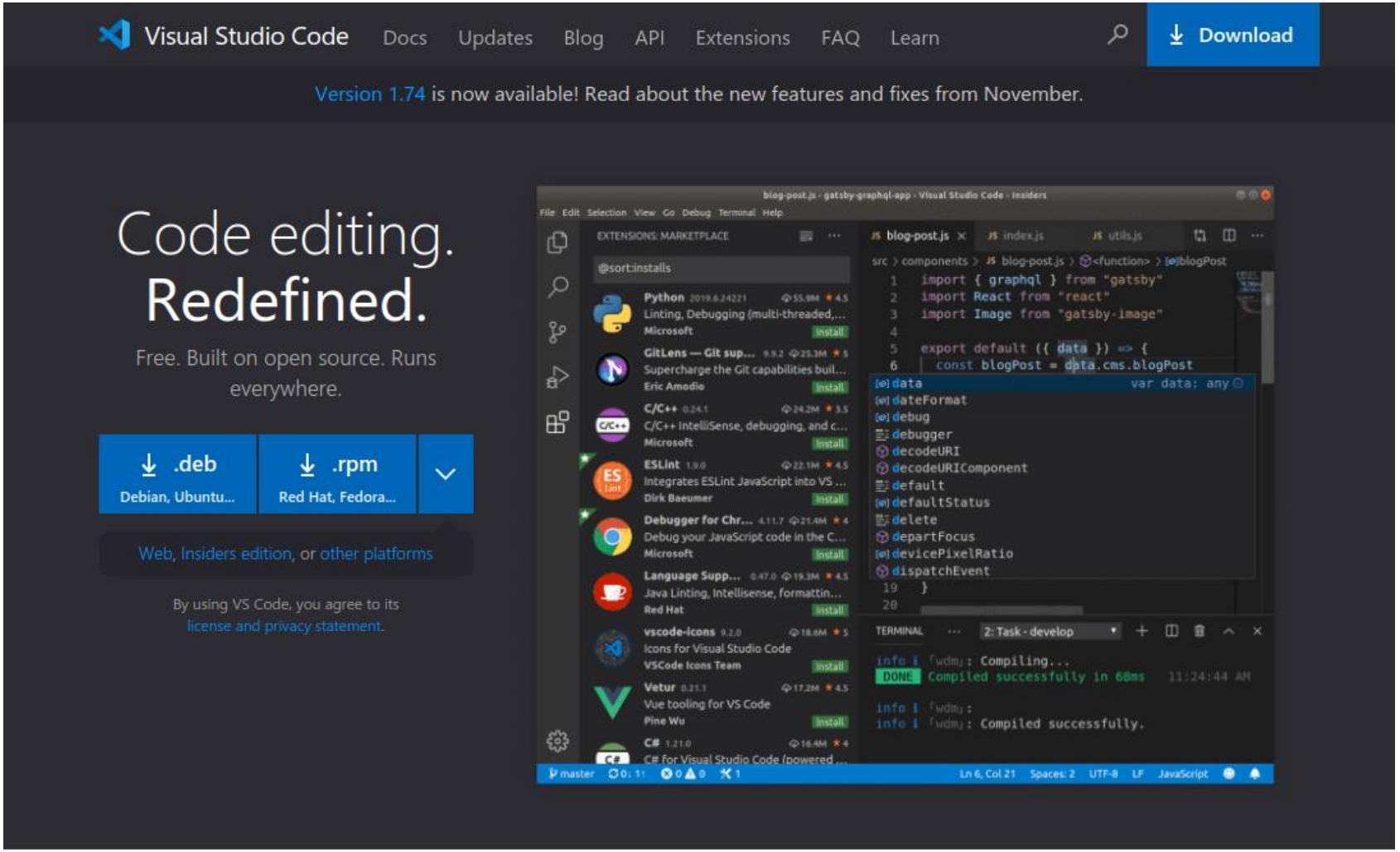
https://ethereum.stackexchange.com/tags/dapp-development/info

Utrecht University

Some (more) suggested tools



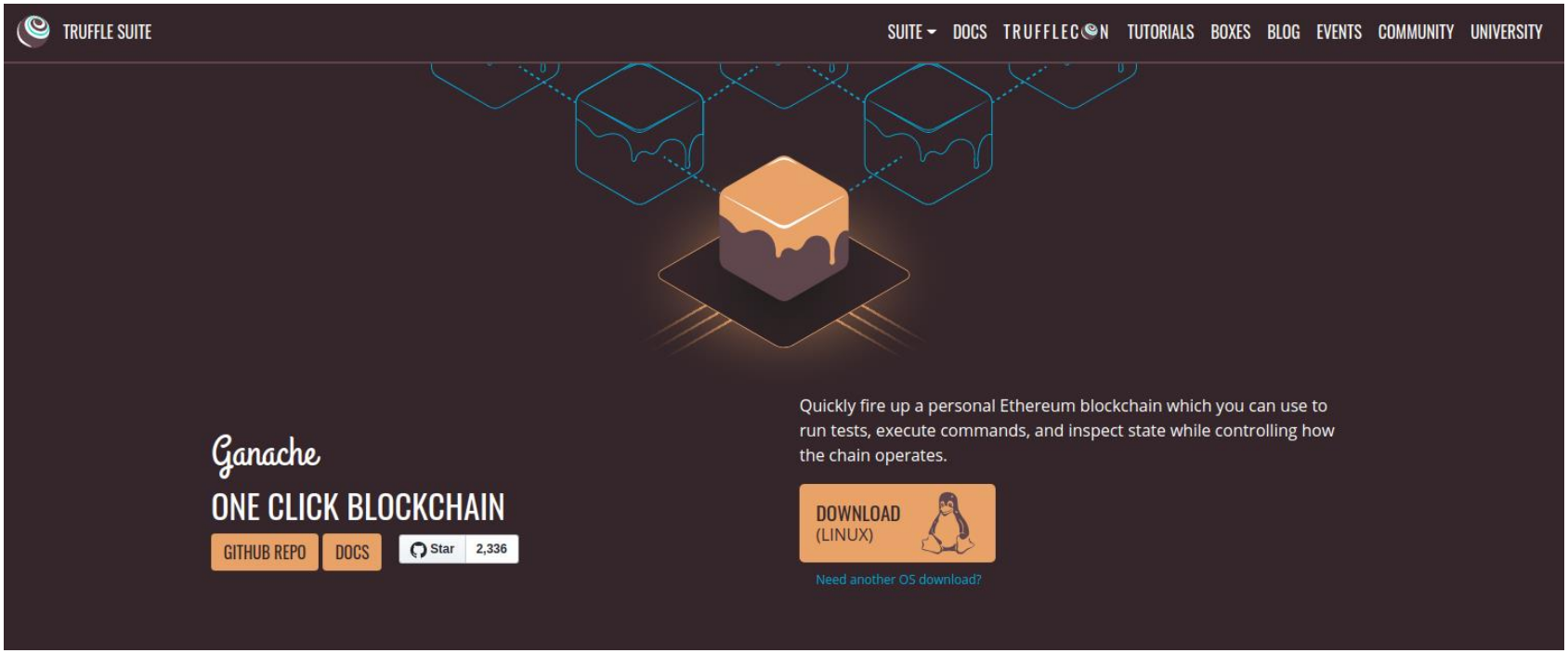
Visual Studio Code





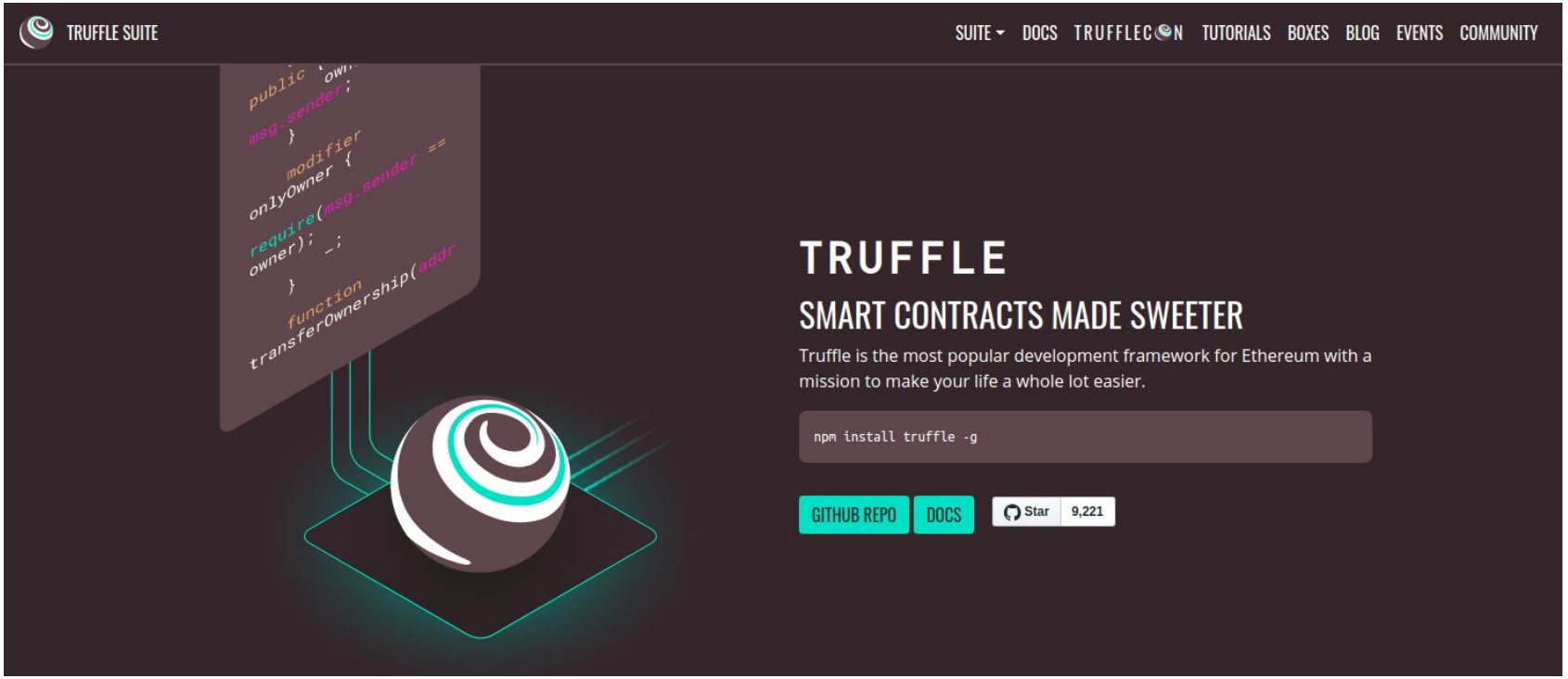

https://code.visualstudio.com/

Ganache



Ganache ships with an internal Javascript implementation of the Ethereum Blockchain which has additional programmatic capabilities

Truffle – https://www.trufflesuite.com/truffle



MetaMask – https://metamask.io/

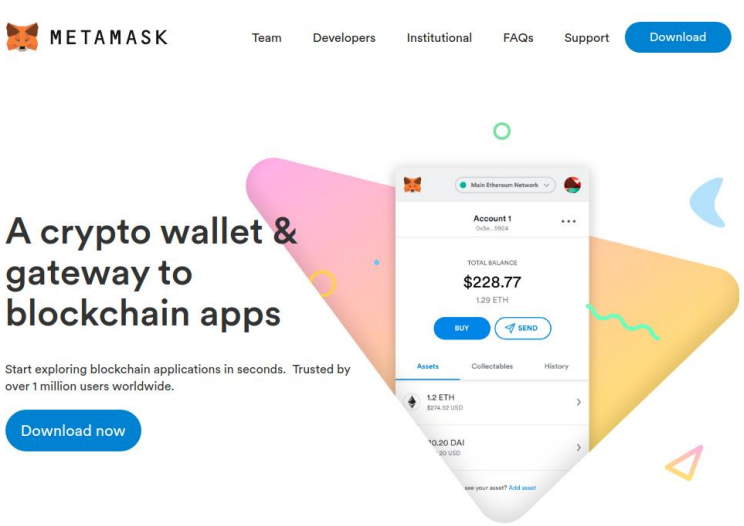
Home > Extensions > MetaMask

MetaMask

Offered by: https://metamask.io

1,803 | Productivity | 1,000,000+ users

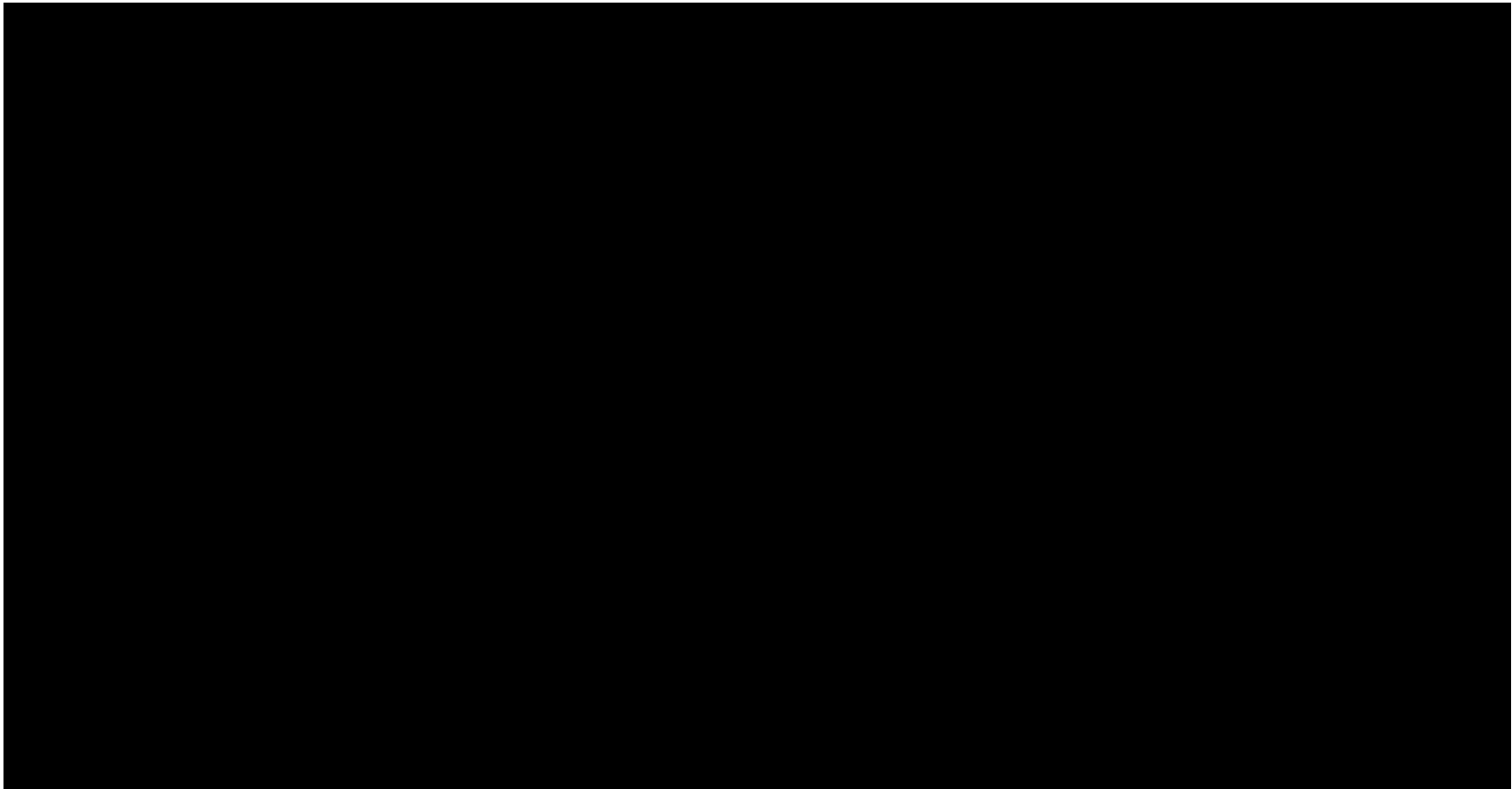
Add to Chrome



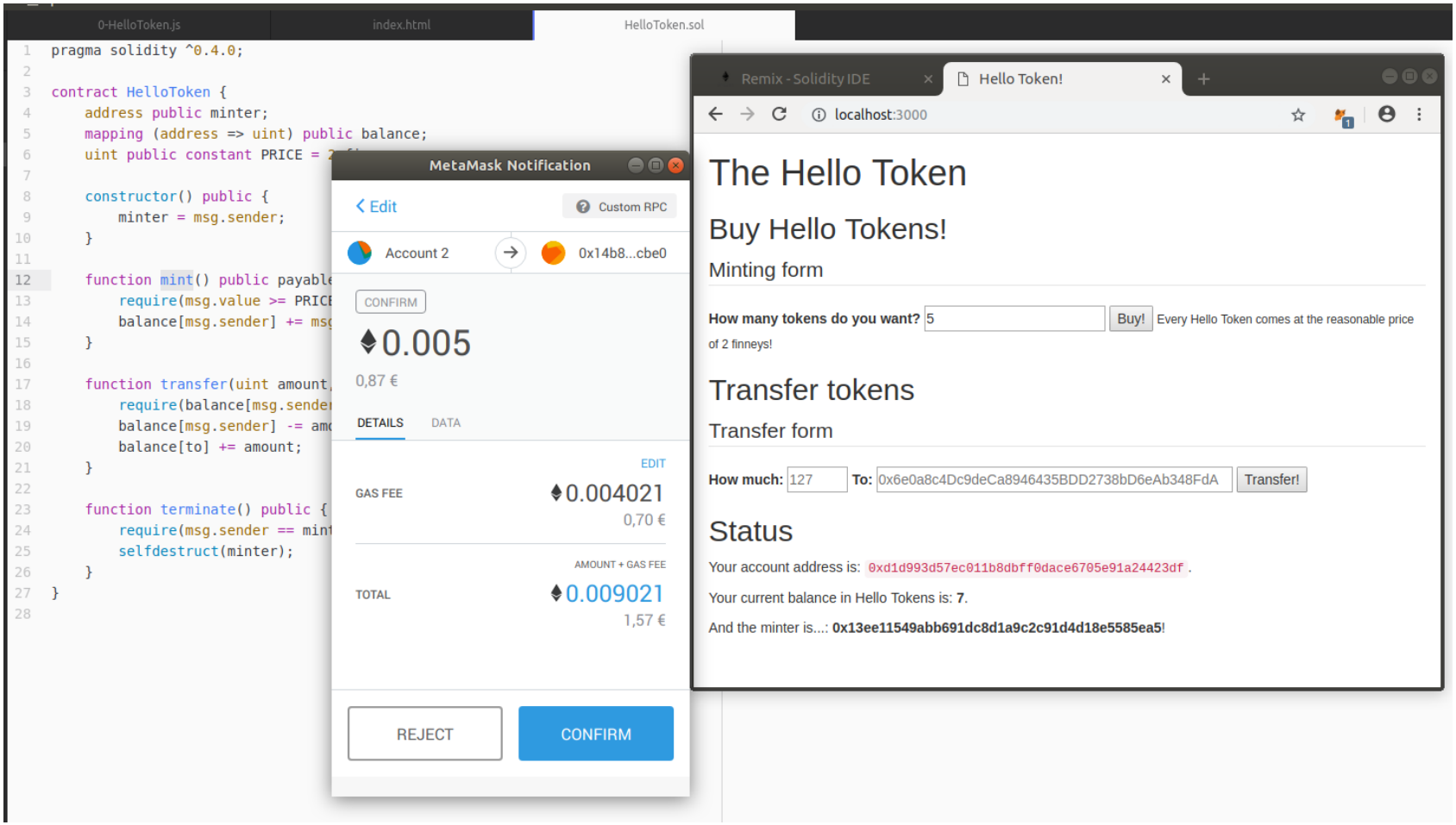
- Allows Ethereum DApps to be run in the browser without the need to run a full Ethereum node.
- Includes a (secure) identity vault, providing a user interface to manage identities on different sites and sign blockchain transactions

Utrecht University

Tokens

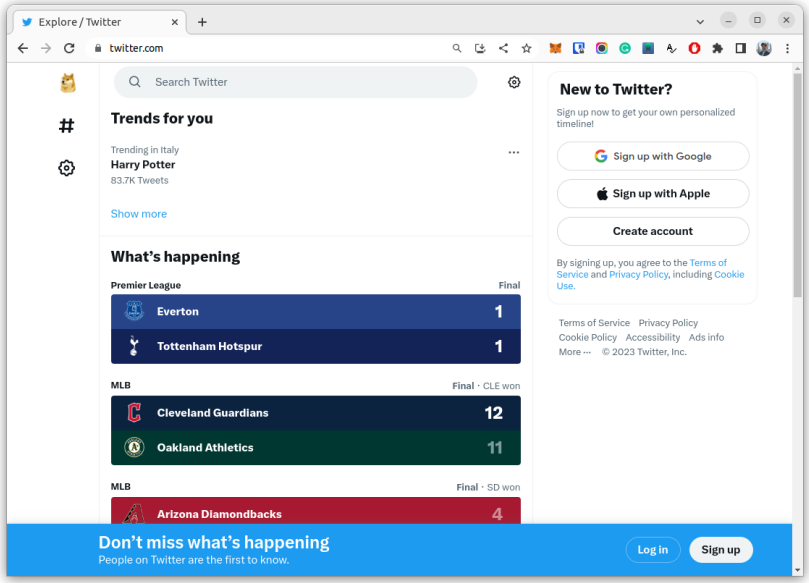


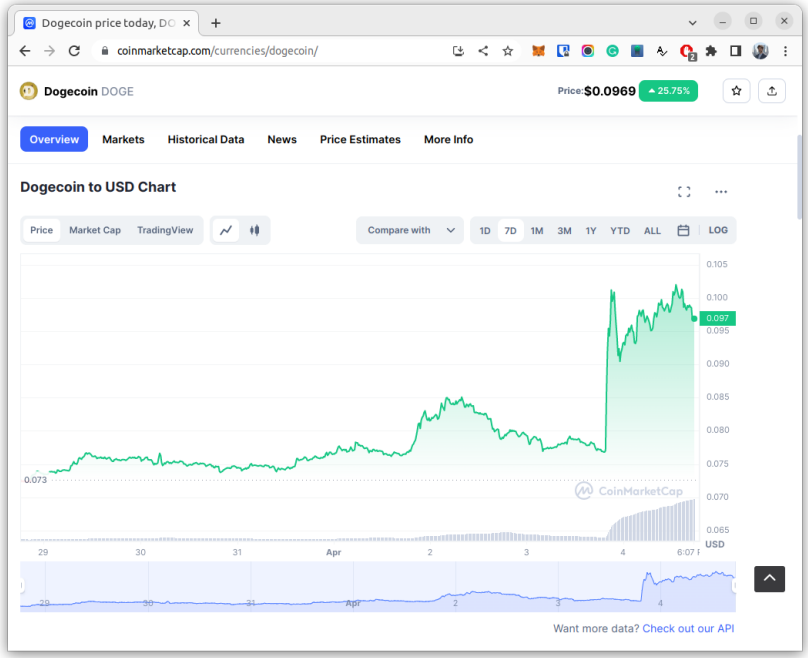
Your brand new token in 5 minutes or less



Tokens tokens tokens tokens

No comment





Sources: twitter.com (left), coinmarketcap.com (right)

Some non-fungible tokens (NFTs)

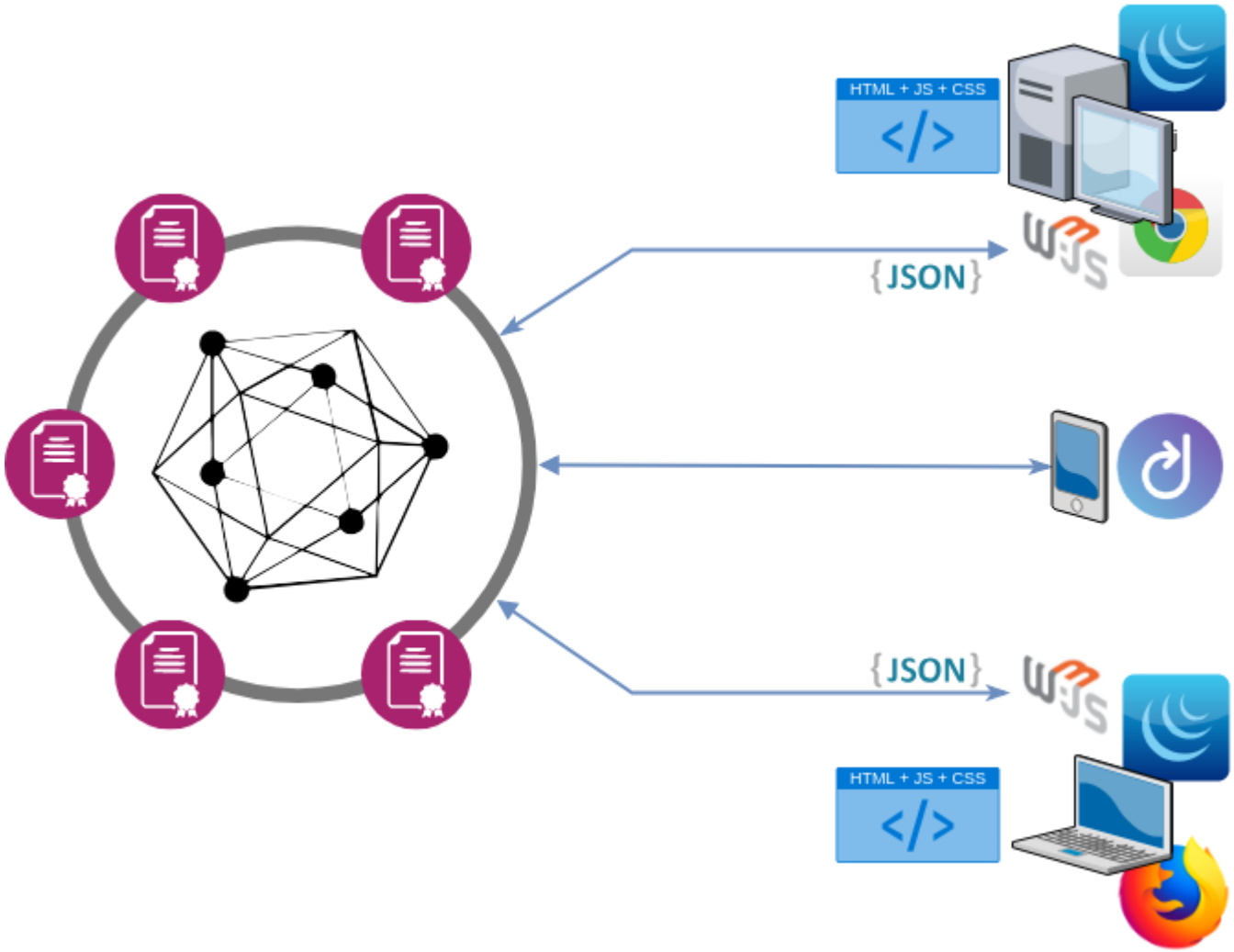
- Crypto collectibles & games
- Property titles
  - Art
  - Collectibles
  - Memorabilia
  - Real Estate
  - Certificates
- Identity
- Keys & passes
- Fractional ownership of physical goods
- Purpose-driven, social-prescribing tokens
  - MCO2 token, TREE token, Plastic Bank token, …

Figures: https://www.mycryptoheroes.net/ (top) https://mco2token.moss.earth/ (bottom)

Tokens are not cryptofuel: the Steemit example

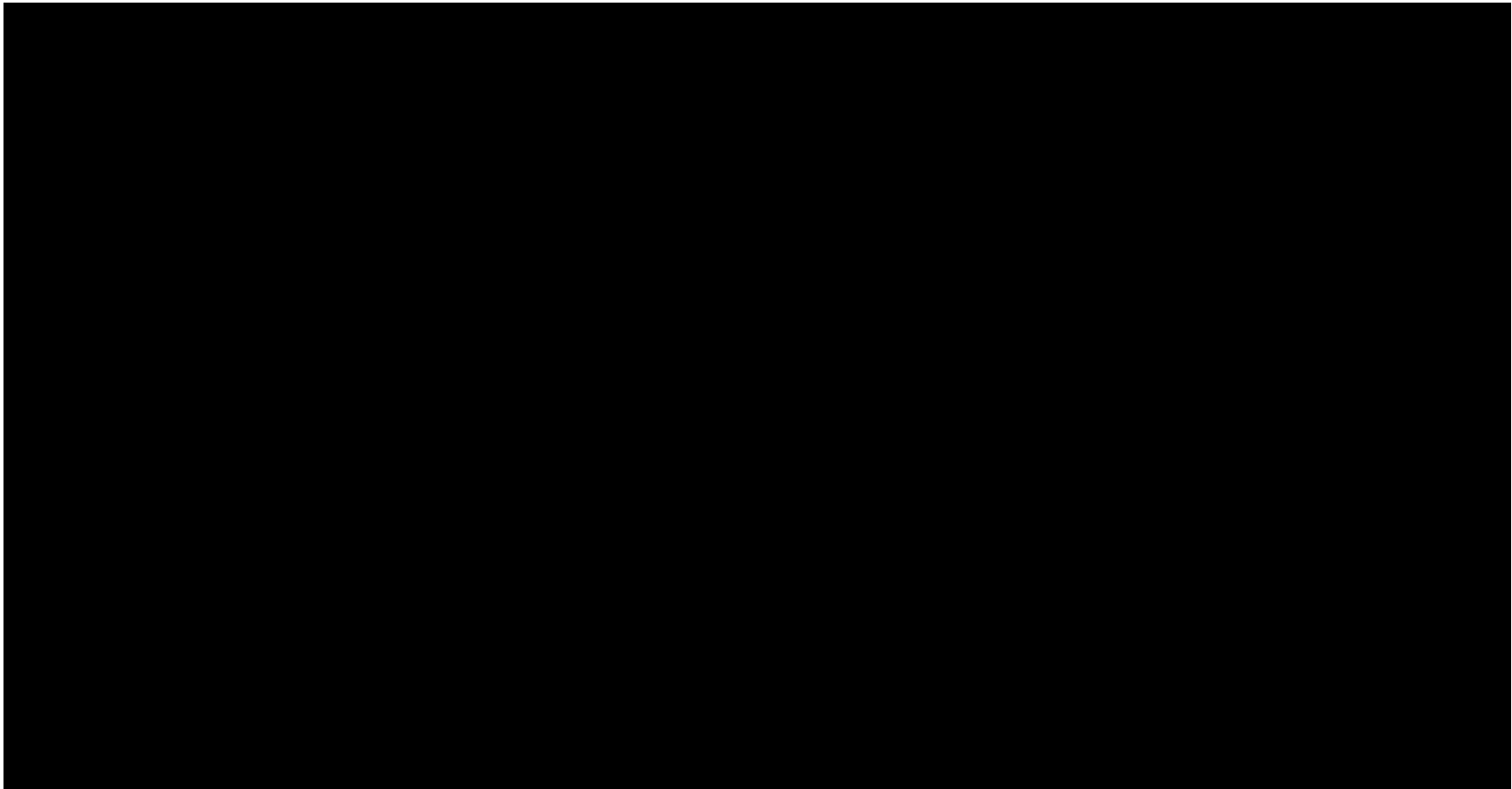
- A blockchain-based blogging and social media website
- Steem Token (STEEM): Access token
  - Gateway token
  - Instantly transferable
  - Does not incentivize behaviour directly
- Steem Power (SP): Reputation token
  - 1:1 convert ratio with Steem token
  - Needed to make posts, upvotes
  - One can earn SP with upvotes, posts, comments, delegation
  - Non transferrable
- Steem Dollars (SBD): 1 Steem Dollar = 1 USD
  - Stable coin, to act as store of value
  - Earn interest p.a. (only if market price is not higher than US Dollar)
  - Transferrable
  - Pay bots for liking contents

Decentralised application (DApp)



Utrecht University

Some interesting effects of Turing completeness

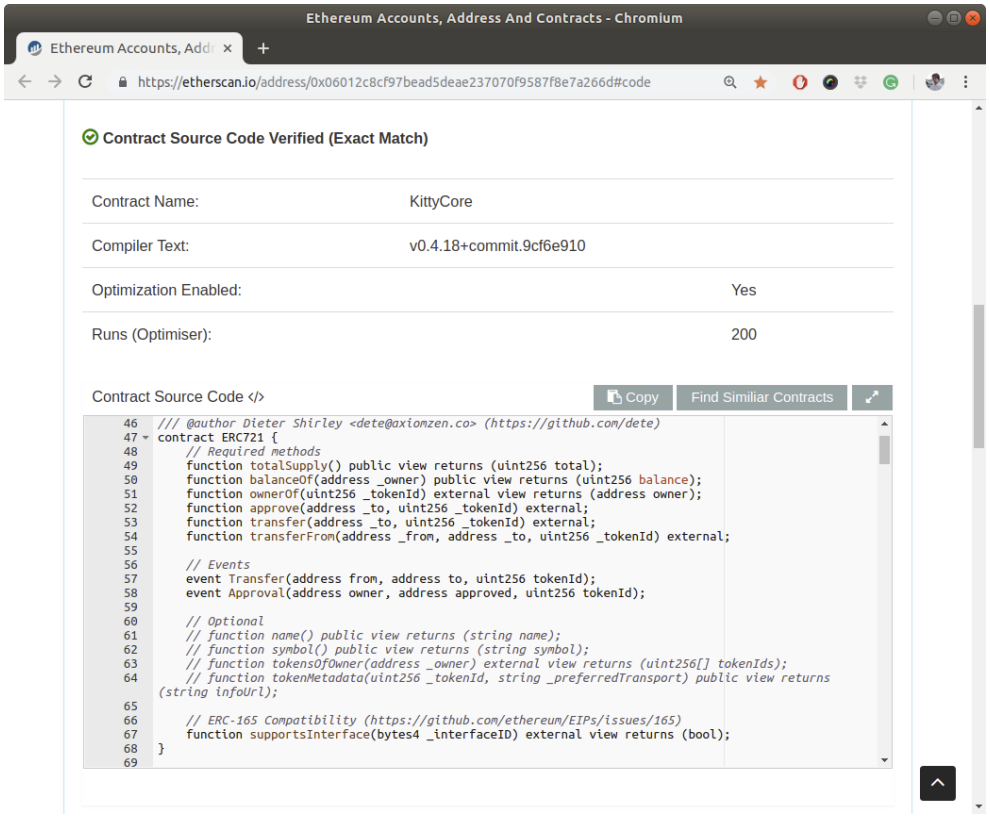


Things may go out of control

"CryptoKitties is a game centered around breedable, collectible, and oh-so-adorable creatures we call CryptoKitties! Each cat is one-of-a-kind and 100% owned by you; it cannot be replicated, taken away, or destroyed."

https://www.cryptokitties.co/

Still code, under the hood




```
/// @dev The main Kitty struct. Every cat in CryptoKitties is represented by a copy
///  of this structure, so great care was taken to ensure that it fits neatly into
///  exactly two 256-bit words. Note that the order of the members in this structure
///  is important because of the byte-packing rules used by Ethereum.
///  Ref: http://solidity.readthedocs.io/en/develop/miscellaneous.html
struct Kitty {
    // The Kitty's genetic code is packed into these 256-bits, the format is
    // sooper-sekret! A cat's genes never change.
    uint256 genes;

    // The timestamp from the block when this cat came into existence.
    uint64 birthTime;

    // The minimum timestamp after which this cat can engage in breeding
    // activities again. This same timestamp is used for the pregnancy
    // timer (for matrons) as well as the siring cooldown.
    uint64 cooldownEndBlock;

    // The ID of the parents of this kitty, set to 0 for gen0 cats.
    // Note that using 32-bit unsigned integers limits us to a "mere"
    // 4 billion cats. This number might seem small until you realize
    // that Ethereum currently has a limit of about 500 million
    // transactions per year! So, this definitely won't be a problem
    // for several years (even as Ethereum learns to scale).
    uint32 matronId;
    uint32 sireId;

    // Set to the ID of the sire cat for matrons that are pregnant,
    // zero otherwise. A non-zero value here is how we know a cat
    // is pregnant. Used to retrieve the genetic material for the new
    // kitten when the birth transpires.
    uint32 siringWithId;

    // Set to the index in the cooldown array (see below) that represents
    // the current cooldown duration for this Kitty. This starts at zero
    // for gen0 cats, and is initialized to floor(generation/2) for others.
    // Incremented by one for each successful breeding action, regardless
    // of whether this cat is acting as matron or sire.
    uint16 cooldownIndex;

    // The "generation number" of this cat. Cats minted by the CK contract
    // for sale are called "gen0" and have a generation number of 0. The
    // generation number of all other cats is the larger of the two generation
    // numbers of their parents, plus one.
    // (i.e. max(matron.generation, sire.generation) + 1)
    uint16 generation;
}
```

Cryptokitties use (secret) genetic algorithms for breeding simulation

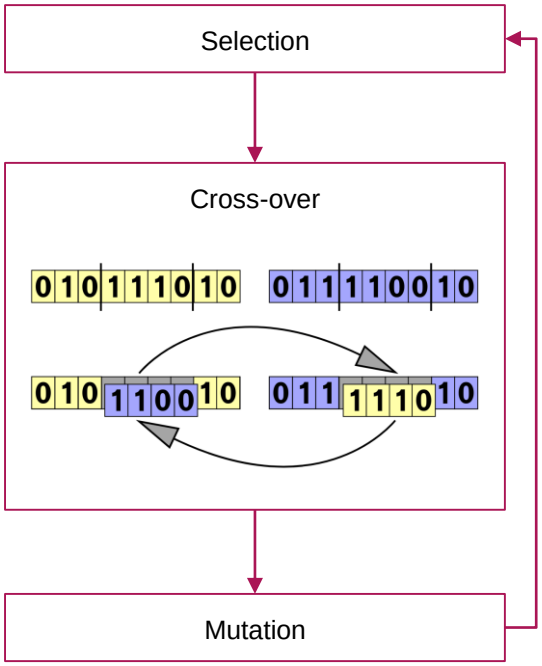
EthFiddle Security Audit Share Login

```
1022    function giveBirth(uint256 _matronId)
1023        external
1024        whenNotPaused
1025        returns(uint256)
1026    {
1027        // Grab a reference to the matron in storage.
1028        Kitty storage matron = kitties[_matronId];
1029
1030        // Check that the matron is a valid cat.
1031        require(matron.birthTime != 0);
1032
1033        // Check that the matron is pregnant, and that its time has come!
1034        require(_isReadyToGiveBirth(matron));
1035
1036        // Grab a reference to the sire in storage.
1037        uint256 sireId = matron.siringWithId;
1038        Kitty storage sire = kitties[sireId];
1039
1040        // Determine the higher generation number of the two parents
1041        uint16 parentGen = matron.generation;
1042        if (sire.generation > matron.generation) {
1043            parentGen = sire.generation;
1044        }
1045
1046        // Call the sooper-sekret gene mixing operation.
1047        uint256 childGenes = geneScience.mixGenes(matron.genes, sire.genes, matron.cooldownEndBlock - 1);
1048
1049        // Make the new kitten!
1050        address owner = kittyIndexToOwner[_matronId];
1051        uint256 kittenId = _createKitty(_matronId, matron.siringWithId, parentGen + 1, childGenes, owner);
1052
```

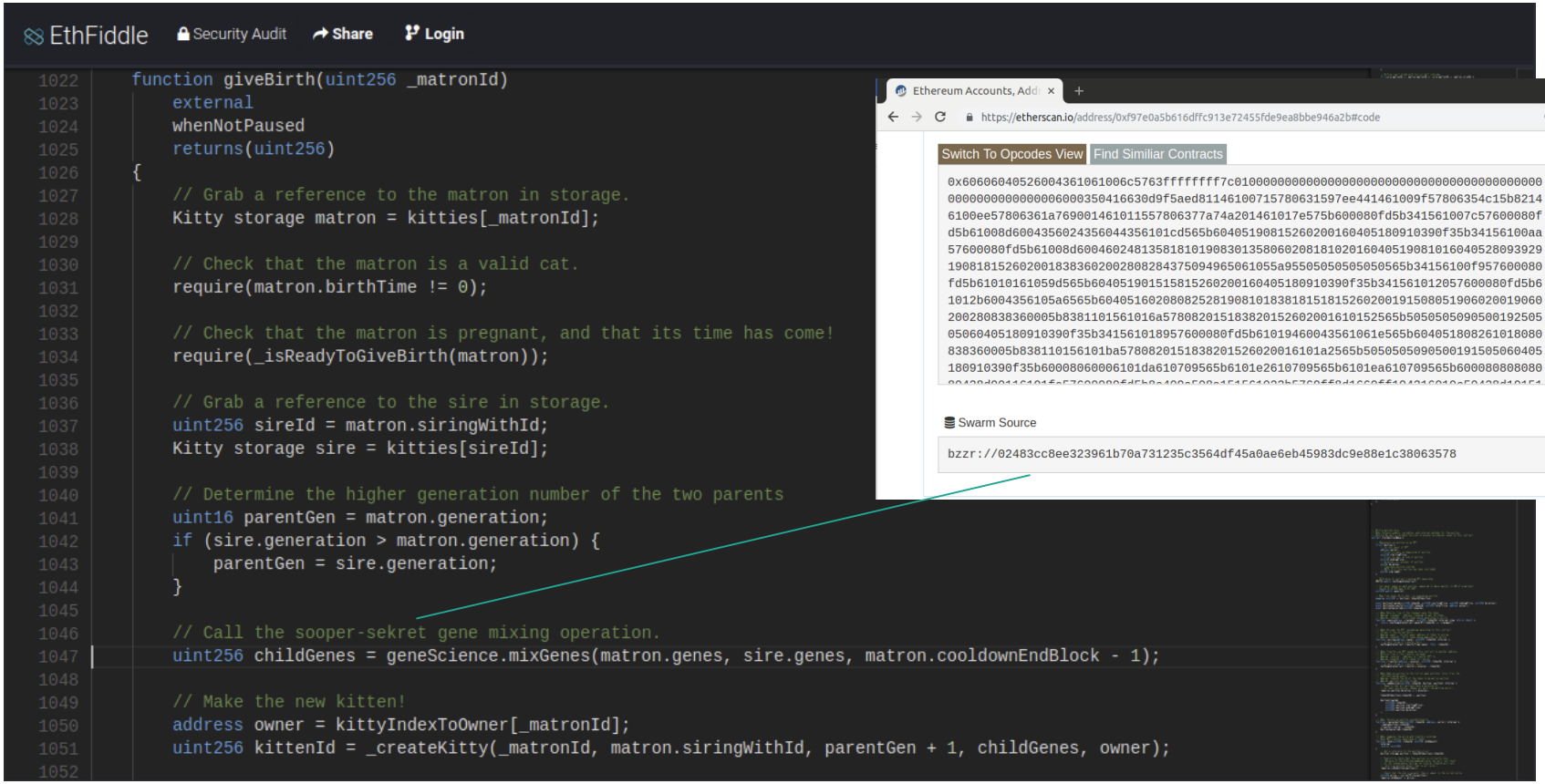
Cryptokitties use (secret) genetic algorithms for breeding simulation

EthFiddle Security Audit Share Login

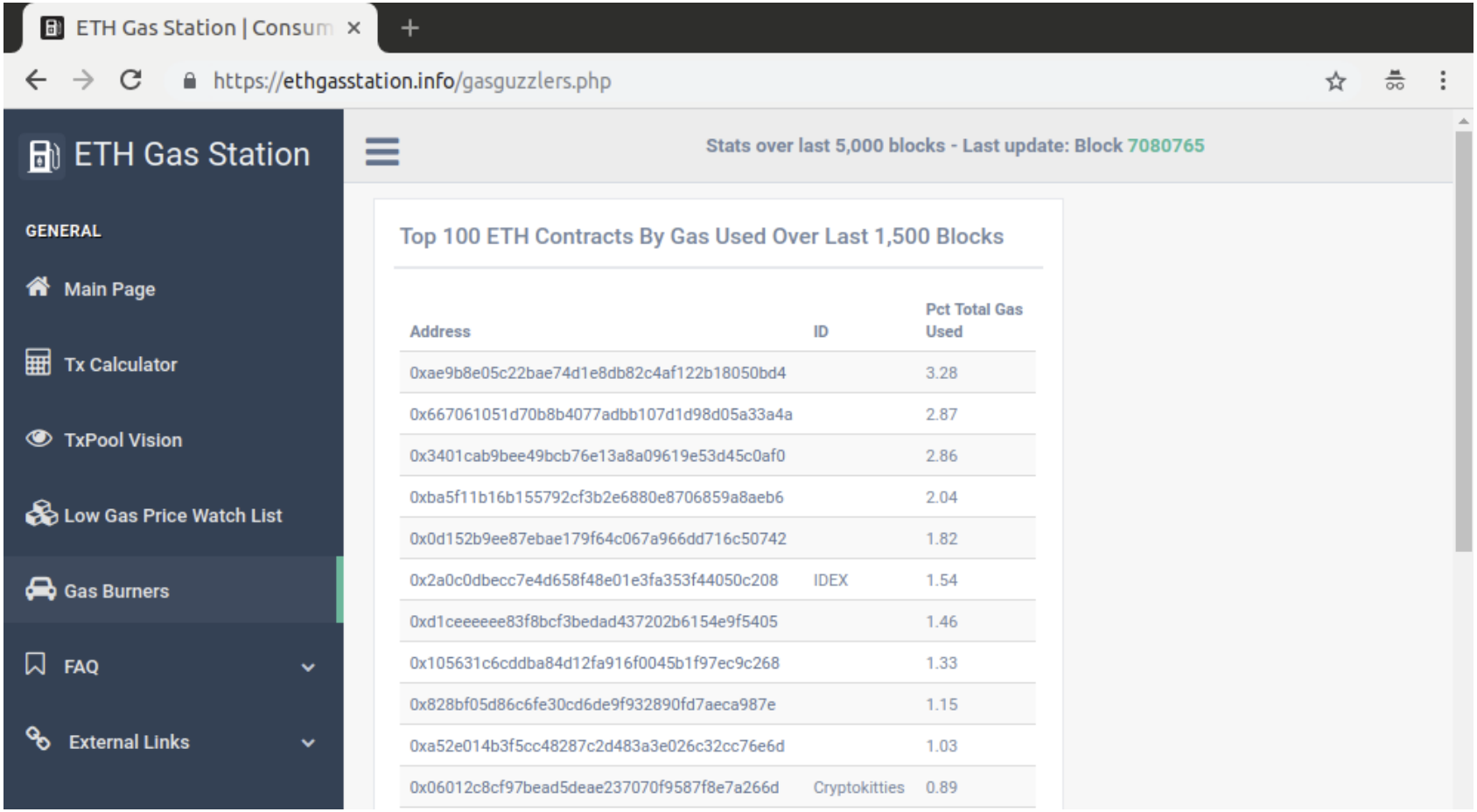
```
1022    function giveBirth(uint256 _matronId)
1023        external
1024        whenNotPaused
1025        returns(uint256)
1026    {
1027        // Grab a reference to the matron in storage.
1028        Kitty storage matron = kitties[_matronId];
1029
1030        // Check that the matron is a valid cat.
1031        require(matron.birthTime != 0);
1032
1033        // Check that the matron is pregnant, and that its time has come!
1034        require(_isReadyToGiveBirth(matron));
1035
1036        // Grab a reference to the sire in storage.
1037        uint256 sireId = matron.siringWithId;
1038        Kitty storage sire = kitties[sireId];
1039
1040        // Determine the higher generation number of the two parents
1041        uint16 parentGen = matron.generation;
1042        if (sire.generation > matron.generation) {
1043            parentGen = sire.generation;
1044        }
1045
1046        // Call the sooper-sekret gene mixing operation.
1047        uint256 childGenes = geneScience.mixGenes(matron.genes, sire.genes, matron.cooldownEndBlock - 1);
1048
1049        // Make the new kitten!
1050        address owner = kittyIndexToOwner[_matronId];
1051        uint256 kittenId = _createKitty(_matronId, matron.siringWithId, parentGen + 1, childGenes, owner);
1052
```



Cryptokitties use (secret) genetic algorithms



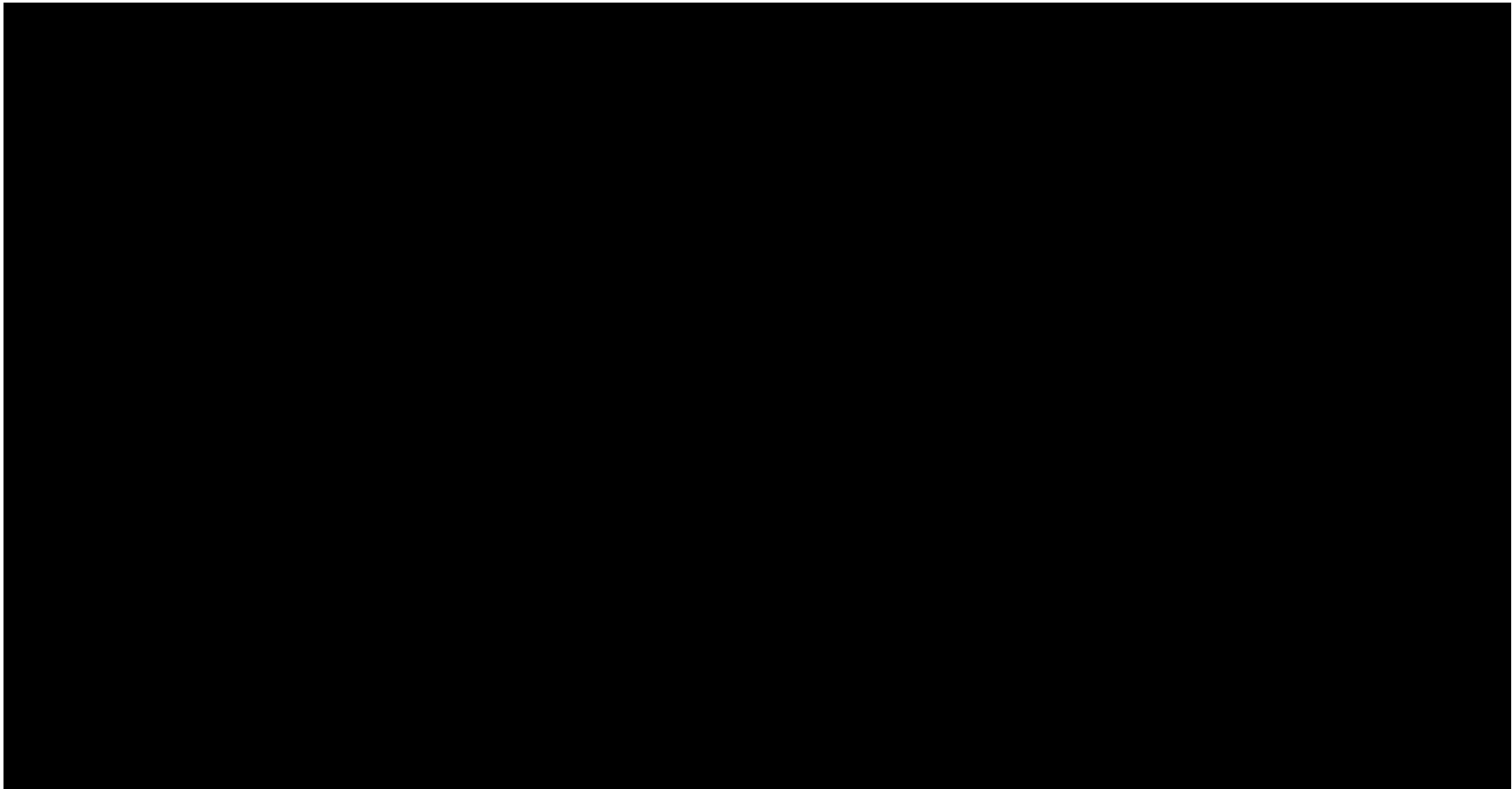
Cryptokitties are greedy for gas



https://ethgasstation.info/gasguzzlers.php (Jan 2019)

Utrecht University

Hello Token



HelloToken (cyptoeconomic version of Hello World)

```
1  // SPDX-License-Identifier: CC-BY-SA-4.0
2  pragma solidity >=0.4.0 <0.9.0;
3
4  contract HelloToken {
5      address public minter;
6      mapping (address => uint) public balance;
7      uint public constant PRICE = 2 * 1e15;
8      // uint public constant PRICE = 2 finney;
9      // finney is no longer a supported denomination since Solidity v.0.7.0
10
11     constructor() {
12         minter = msg.sender;
13     }
14
15     function mint() public payable {
16         require(msg.value >= PRICE, "Not enough value for a token!");
17         balance[msg.sender] += msg.value / PRICE;
18         // Guess guess, where does the remainder of the msg.value end?
19     }
20
21     function transfer(uint amount, address to) public {
22         require(balance[msg.sender] >= amount, "Not enough tokens!");
23         balance[msg.sender] -= amount;
24         balance[to] += amount;
25     }
26
27     function terminate() public {
28         require(msg.sender == minter, "You cannot terminate the contract!");
29         selfdestruct(payable(minter));
30     }
31 }
```

Solidity, bytecode and bytes

```sol
// SPDX-License-Identifier: CC-BY-SA-4.0
pragma solidity >=0.4.0 <0.9.0;

contract HelloToken {
    address public minter;
    mapping (address => uint) public balance;
    uint public constant PRICE = 2 * 1e15;
    // uint public constant PRICE = 2 finney;
    // finney is no longer a supported denomination since Solidity v.0.7.0

    constructor() {
        minter = msg.sender;
    }

    function mint() public payable {
        require(msg.value >= PRICE, "Not enough value for a token!");
        balance[msg.sender] += msg.value / PRICE;
        // Guess guess, where does the remainder of the msg.value end?
    }

    function transfer(uint amount, address to) public {
        require(balance[msg.sender] >= amount, "Not enough tokens!");
        balance[msg.sender] -= amount;
        balance[to] += amount;
    }

    function terminate() public {
        require(msg.sender == minter, "You cannot terminate the contract!");
        selfdestruct(payable(minter));
    }
}
```

The rules of the game

- The owner of the Smart Contract is the creator
  - The owner writes the code
  - The owner sends a transaction to the blockchain with that code attached
    - The Smart Contract now has a unique address & balance

- What are the governance rules?
  - Every HelloToken costs 2 finney ($^{2}/_{1000}$ ETH $= 2 \times 10^{15}$ wei)
    - Minimum threshold: 2 finney
  - If I want HelloTokens, I send ETH to the contract
    - If I send the contract 11 finney, I get 5 HelloTokens in my balance
    - If I send 1 finney, my request is refused

Your brand new token in 5 minutes or less

```sol
// SPDX-License-Identifier: CC-BY-SA-4.0
pragma solidity >=0.8.0 <0.9.0;

contract HelloToken {
    address public minter;
    mapping (address => uint) public balance;
    uint public constant PRICE = 2 * 1e15;
    // uint public constant PRICE = 2 finney;
    // finney is no longer a supported denomination since Solidity v.0.7.0

    constructor() {
        minter = msg.sender;
    }

    function mint() public payable {
        require(msg.value >= PRICE, "Not enough value for a token!");
        balance[msg.sender] += msg.value / PRICE;
        // Guess guess, where does the remainder of the msg.value end?
    }

    function transfer(uint amount, address to) public {
        require(balance[msg.sender] >= amount, "Not enough tokens!");
        balance[msg.sender] -= amount;
        balance[to] += amount;
    }
}
```

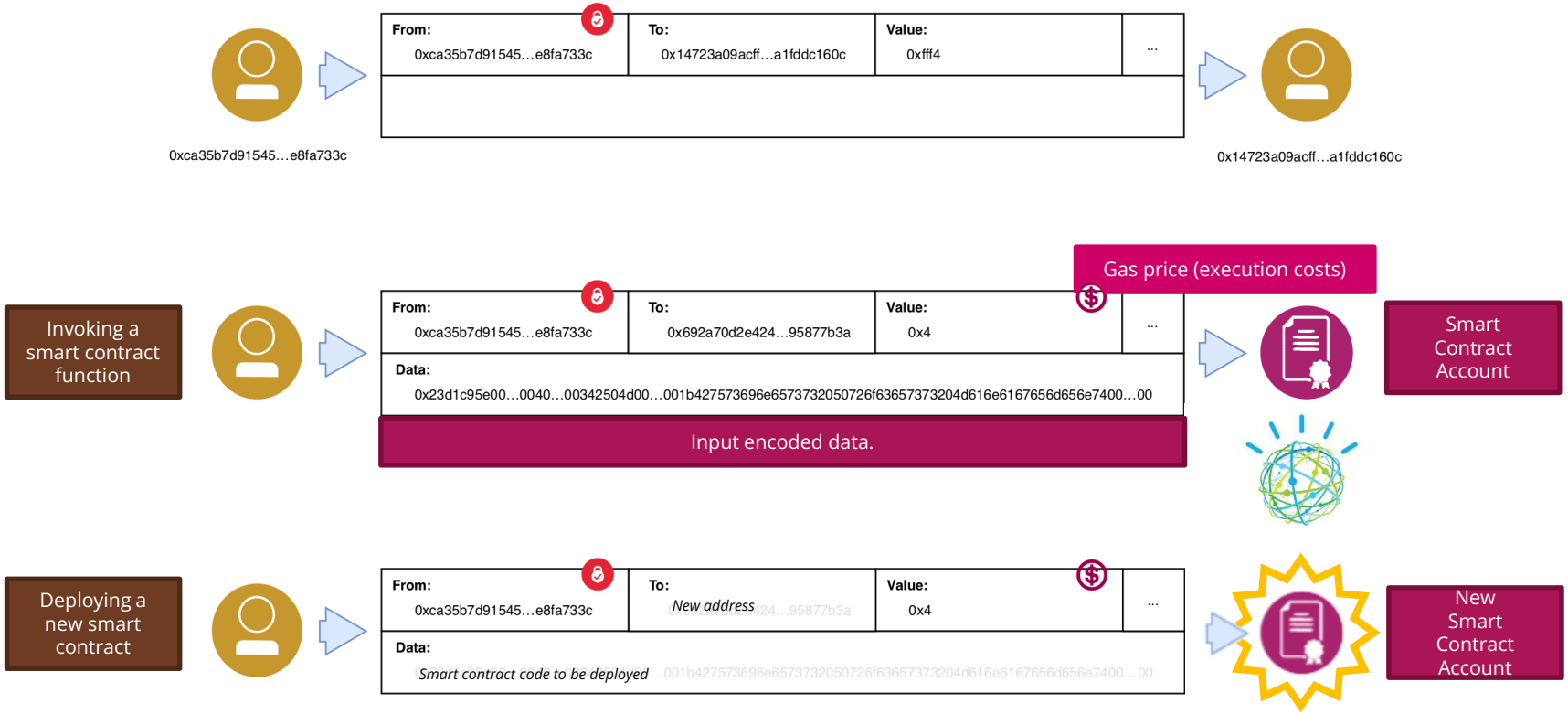
The name of the contract and the token coincide: “HelloToken”

msg.sender is the identity (address) of the one who calls the function

msg.value is the amount of wei (a fraction of ETH) attached

! Do not do this at home. This code is simple yet prone to basically any well-known attack.

A programmable distributed environment



The rules of the game (updated)

- The owner of the Smart Contract is the creator
  - The owner writes the code
  - The owner sends a transaction to the blockchain with that code attached
    - The Smart Contract now has a unique address & balance

- What are the governance rules?
  - Every HelloToken costs 2 finney (1/1000 ETH)
    - Minimum threshold: 2 finney
  - If I want HelloTokens, I send ETH to the contract
    - If I send the contract 11 finney, I get 5 HelloTokens in my balance
    - If I send 1 finney, my request is refused

  - Who gets the ETH of the contract?
    - The ETH are stored in the Smart Contract account
    - The creator can get the ETH upon the self-destruction of the contract

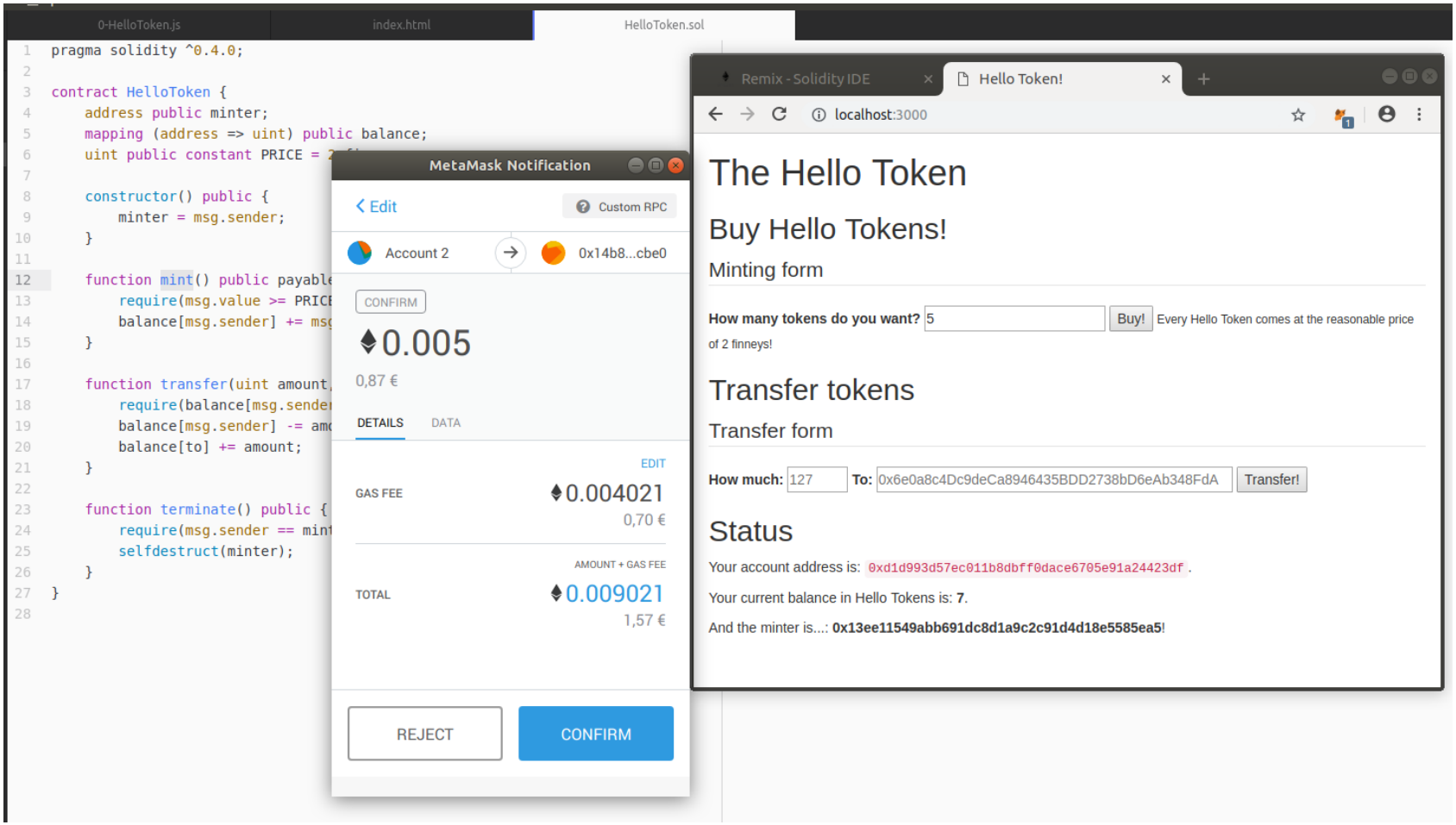
Your brand new token in 5 minutes or less

```solidity
// SPDX-License-Identifier: CC-BY-SA-4.0
pragma solidity >=0.8.0 <0.9.0;

contract HelloToken {
    address public minter;
    mapping (address => uint) public balance;
    uint public constant PRICE = 2 * 1e15;
    // uint public constant PRICE = 2 finney;
    // finney is no longer a supported denomination since Solidity v.0.7.0

    constructor() {
        minter = msg.sender;
    }

    function mint() public payable {
        require(msg.value >= PRICE, "Not enough value for a token!");
        balance[msg.sender] += msg.value / PRICE;
        // Guess guess, where does the remainder of the msg.value end?
    }

    function transfer(uint amount, address to) public {
        require(balance[msg.sender] >= amount, "Not enough tokens!");
        balance[msg.sender] -= amount;
        balance[to] += amount;
    }

    function terminate() public {
        require(msg.sender == minter, "You cannot terminate the contract!");
        selfdestruct(payable(minter));
    }
}
```

At self-destruction, the contract sends its belongings to the account that deployed it

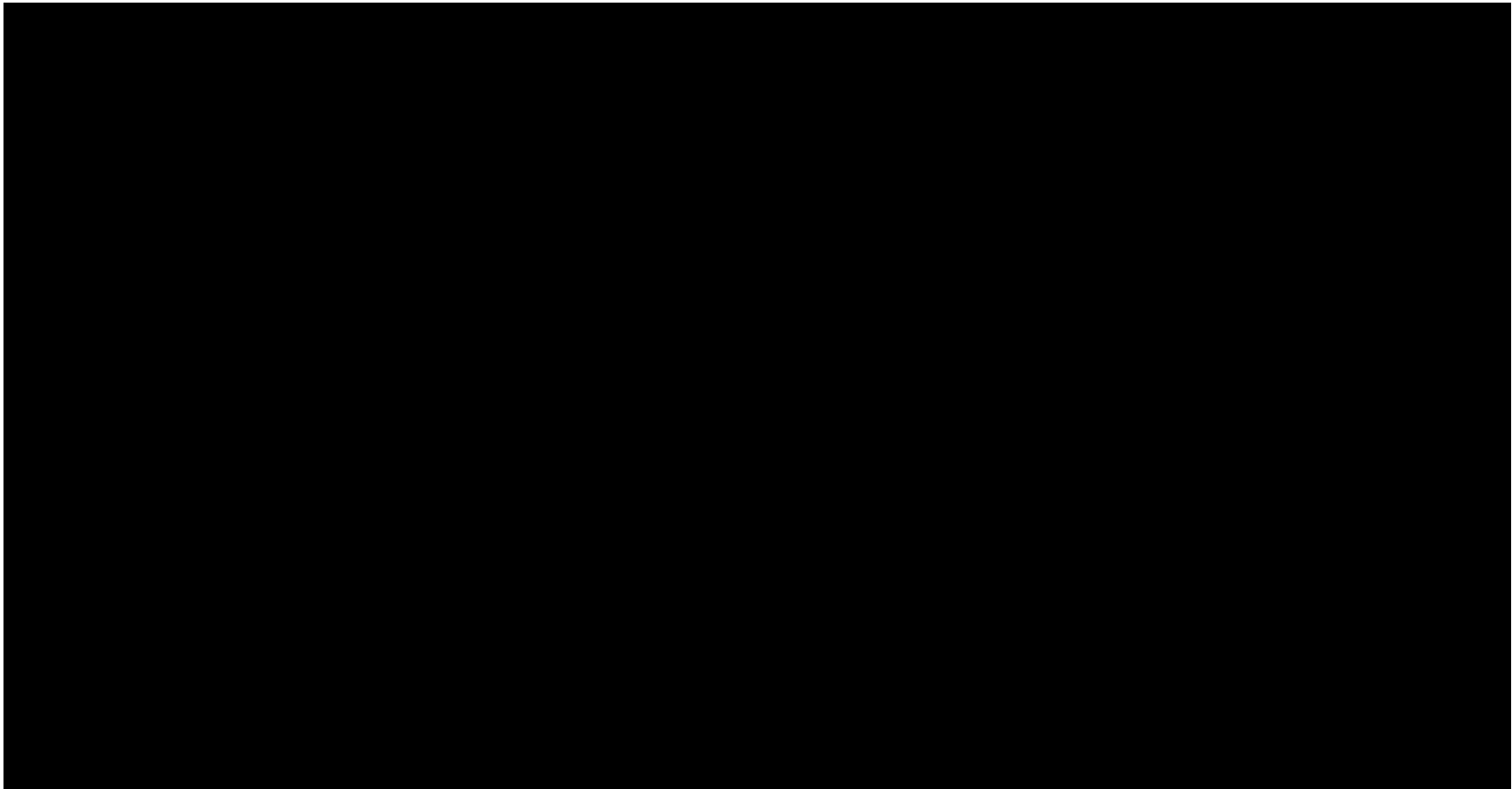
! Do not do this at home. This code is simple yet prone to basically any well-known attack.

Your brand new token in 5 minutes or less



Utrecht University

Announcement



Challenges in building Billion User Cloud Applications (Summer School, Venice, 6-11 October 2024)

- Team of instructors:
  - Dan Ardelean (Distinguished Software Engineer in Google Cloud)
  - Amer Diwan (Distinguished Software Engineer at Google),
  - JJ Furman (founder and creator of Megastore, the system behind Gmail and Drive)
- Organised by Alessandro Panconesi and Emanuele Panizzi (Sapienza University of Rome) for BICI
- Generous fellowships are provided
  - Full room and board for the duration of the school

Interested? Contact me at c.diciccio@uu.nl and Emanuele Panizzi at panizzi@di.uniroma1.it

Agenda for today

13:15 - 13:20:
Recap

13:20 - 14:00:
Smart Contracts and DApps

14:15 - 15:00:
Tokens and hands on Solidity

15:00 - 15:30:
Feedback

15:30 - 17:00:
Architectural debate

Kudos to Jan Martijn for the picture

Utrecht University

DISCLAIMER

The information in this presentation has been compiled with the utmost care,
but no rights can be derived from its contents.