Workshop on
Blockchain Technologies and Applications

Smart Contracts & Tools for Decentralized Applications(DApps)

Arjun Singh Kushwaha
B. Tech (Honors)
IIT Bhilai

Susil Kumar Mohanty
Phd. Scholar
IIT Patna

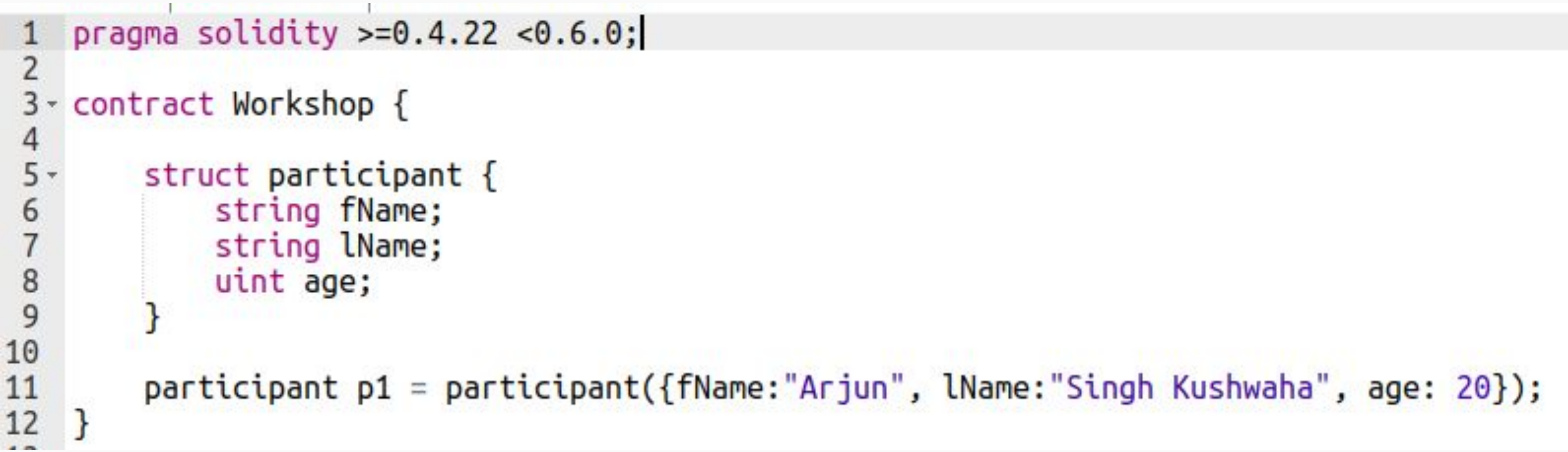
Overview

- **Solidity - Some advanced features**
  - Structs
  - Mappings
  - Hashes
- **Secure and Fair MPC on Blockchain**
  - Coin Toss
- **What Blockchain brings?**
- **Code - Smart Contract for Coin Toss**

Solidity - Some advanced features

Structs

- **struct** in solidity is custom data type.
- Member access operator (.)

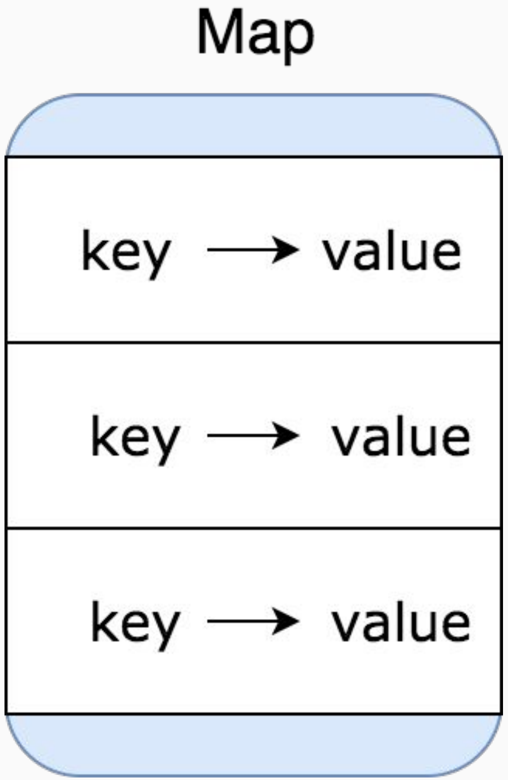


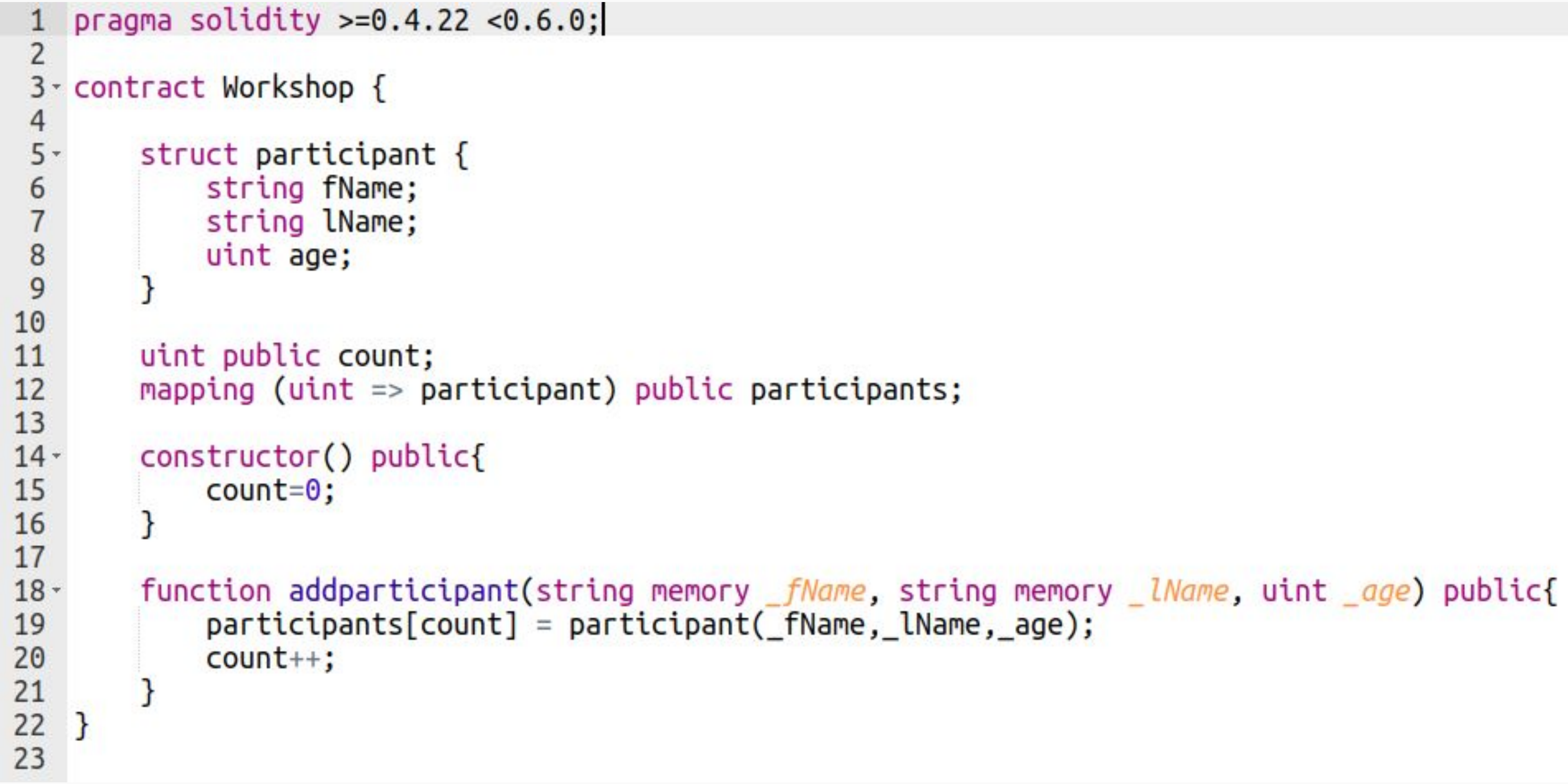
```
pragma solidity >=0.4.22 <0.6.0;

contract Workshop {

    struct participant {
        string fName;
        string lName;
        uint age;
    }

    participant p1 = participant({fName:"Arjun", lName:"Singh Kushwaha", age: 20});
}
```

Mappings

- **Mapping** is used to **structure value types**, such as booleans, integers, addresses, and structs according to key types.
- Two main parts: _KeyType and _ValueType
- Syntax:

```
mapping (_KeyType => _ValueType) mapName;
```

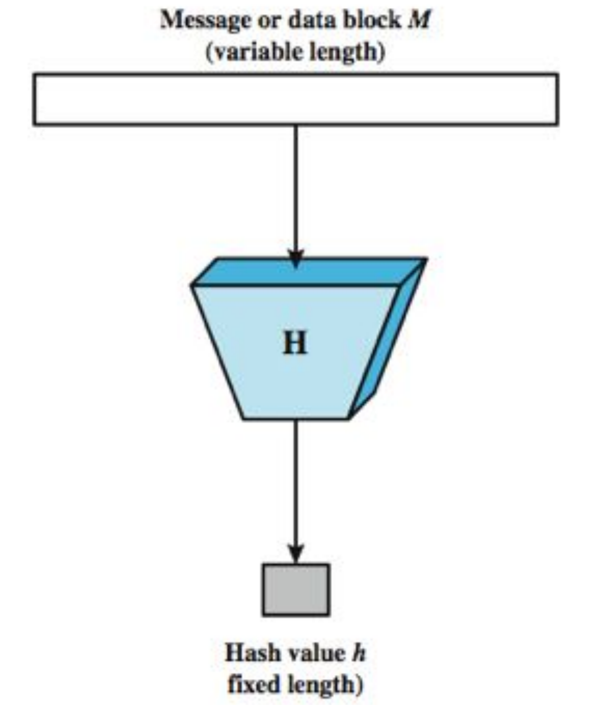




```solidity
pragma solidity >=0.4.22 <0.6.0;

contract Workshop {

    struct participant {
        string fName;
        string lName;
        uint age;
    }

    uint public count;
    mapping (uint => participant) public participants;

    constructor() public{
        count=0;
    }

    function addparticipant(string memory _fName, string memory _lName, uint _age) public{
        participants[count] = participant(_fName,_lName,_age);
        count++;
    }
}

```

Hashes

- Solidity provides **three hash functions**:
  - keccak256()
  - sha256()
  - ripemd()
- **keccak256()**: Native one and most efficient.
- keccak256() has its own EVM opcode.
- They expect an argument of type *bytes memory*, and return an array of *bytes32* (Keccak and SHA) or *bytes20*

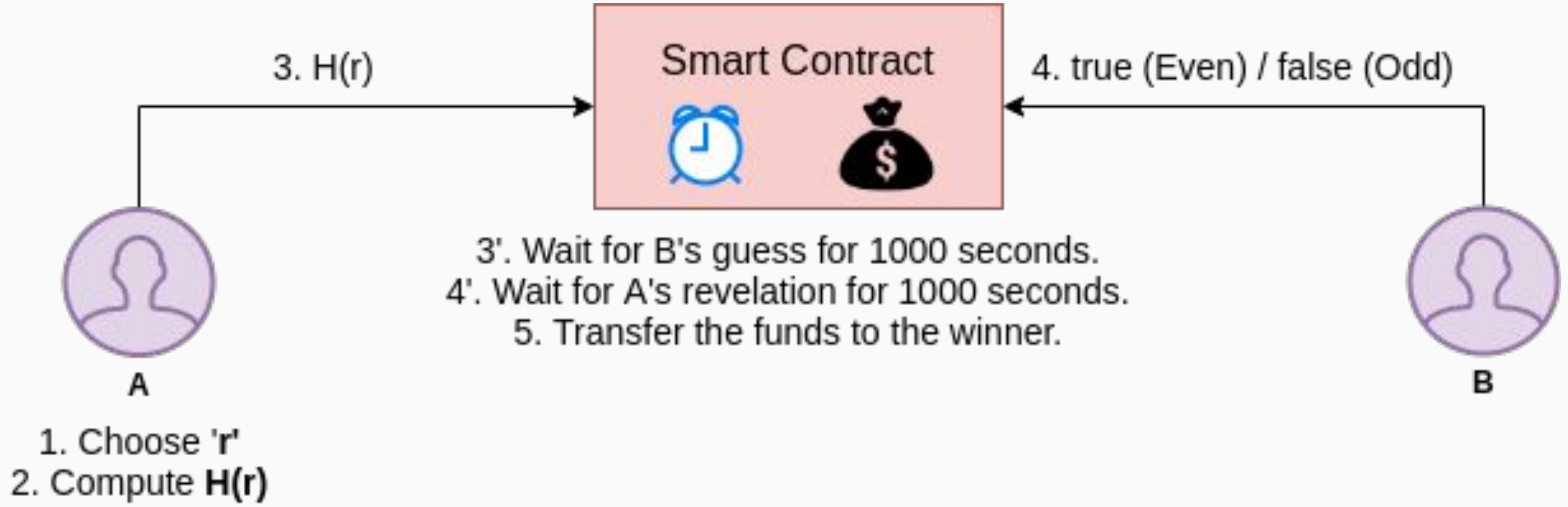


Secure and Fair MPC

Coin Toss

Protocol

Online Coin Toss


3. H(r)
Smart Contract
4. true (Even) / false (Odd)
3'. Wait for B's guess for 1000 seconds.
4'. Wait for A's revelation for 1000 seconds.
5. Transfer the funds to the winner.
A
B
1. Choose 'r'
2. Compute **H(r)**

What Blockchain brings?

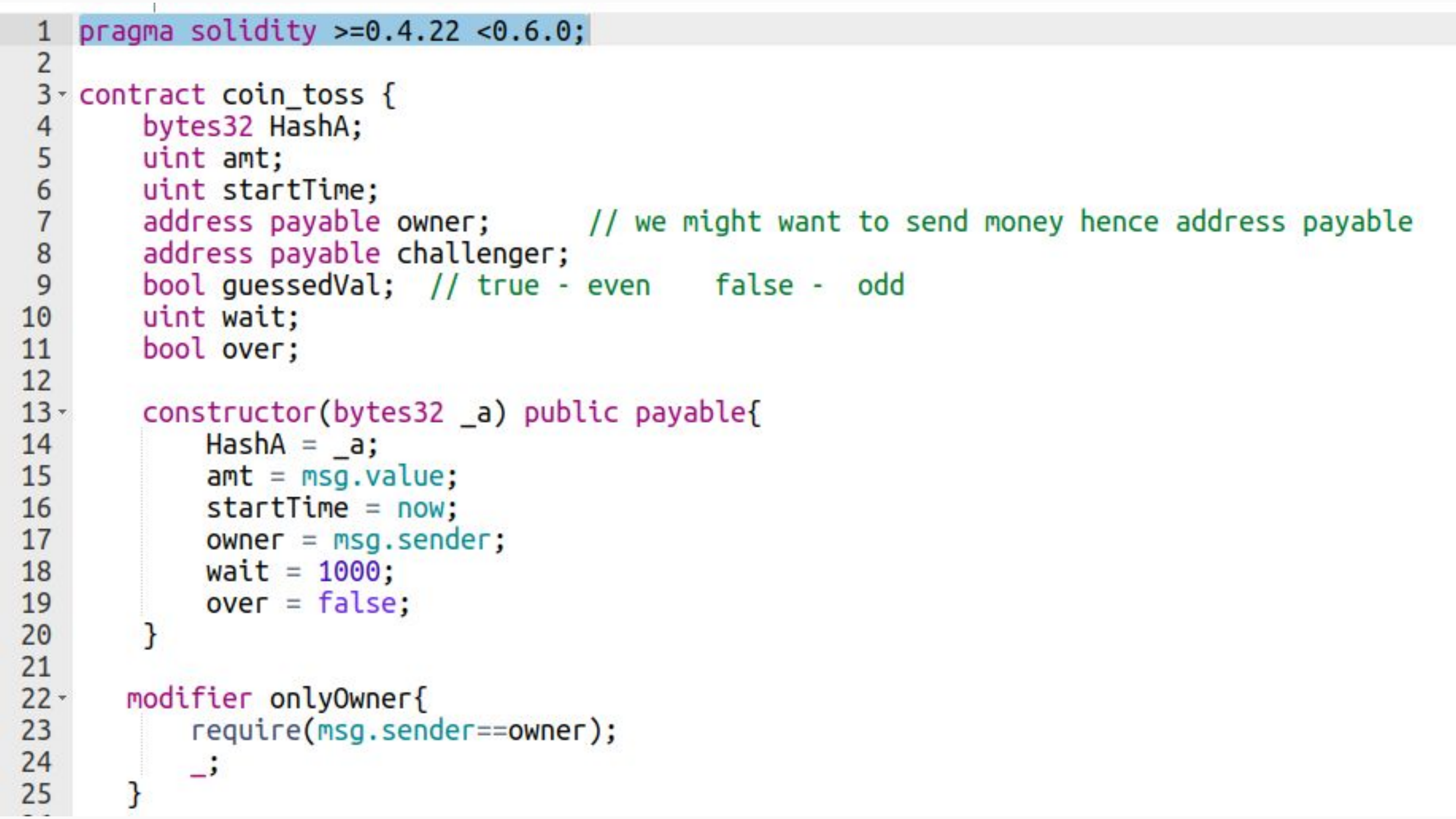
- Immutability
- Public Trust
- Non-repudiation
- Availability

Can you write a smart contract to do it?

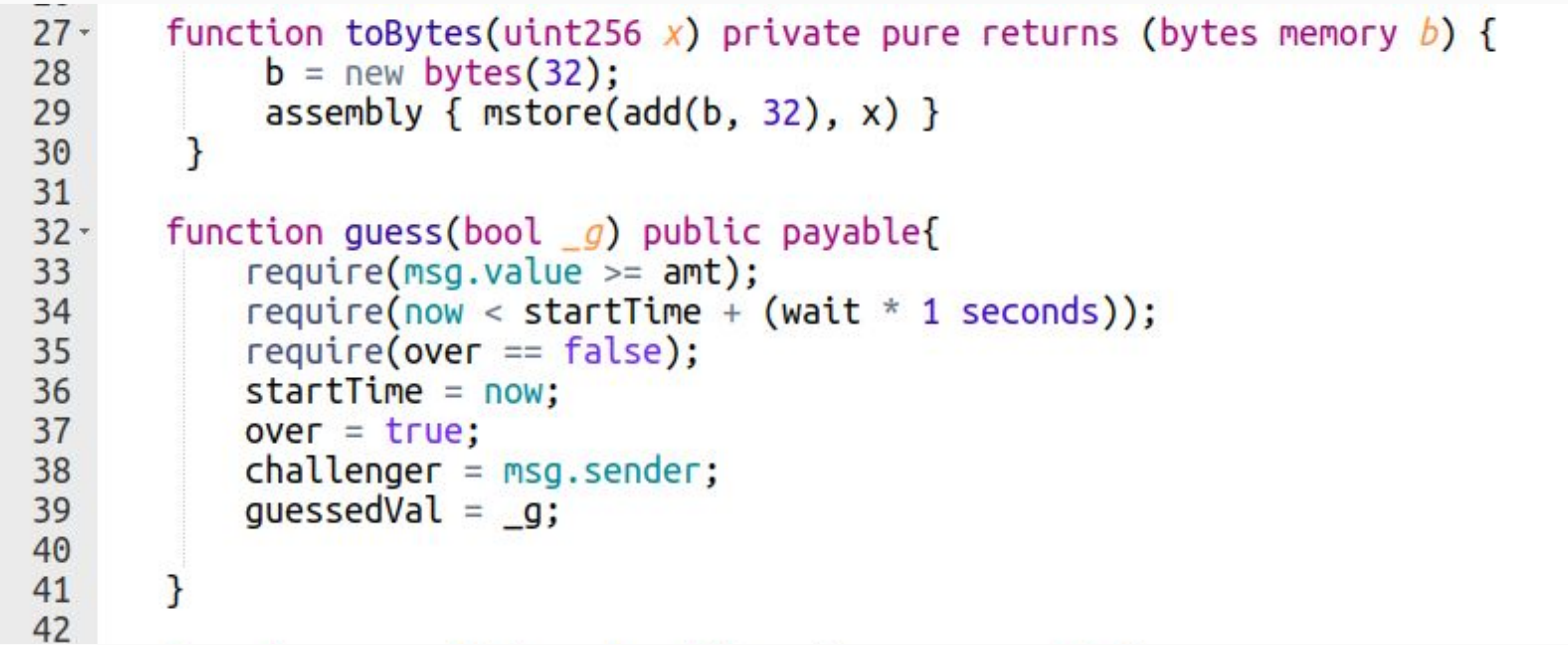
Demo

Find the solution file named ‘**coin_toss.sol**’

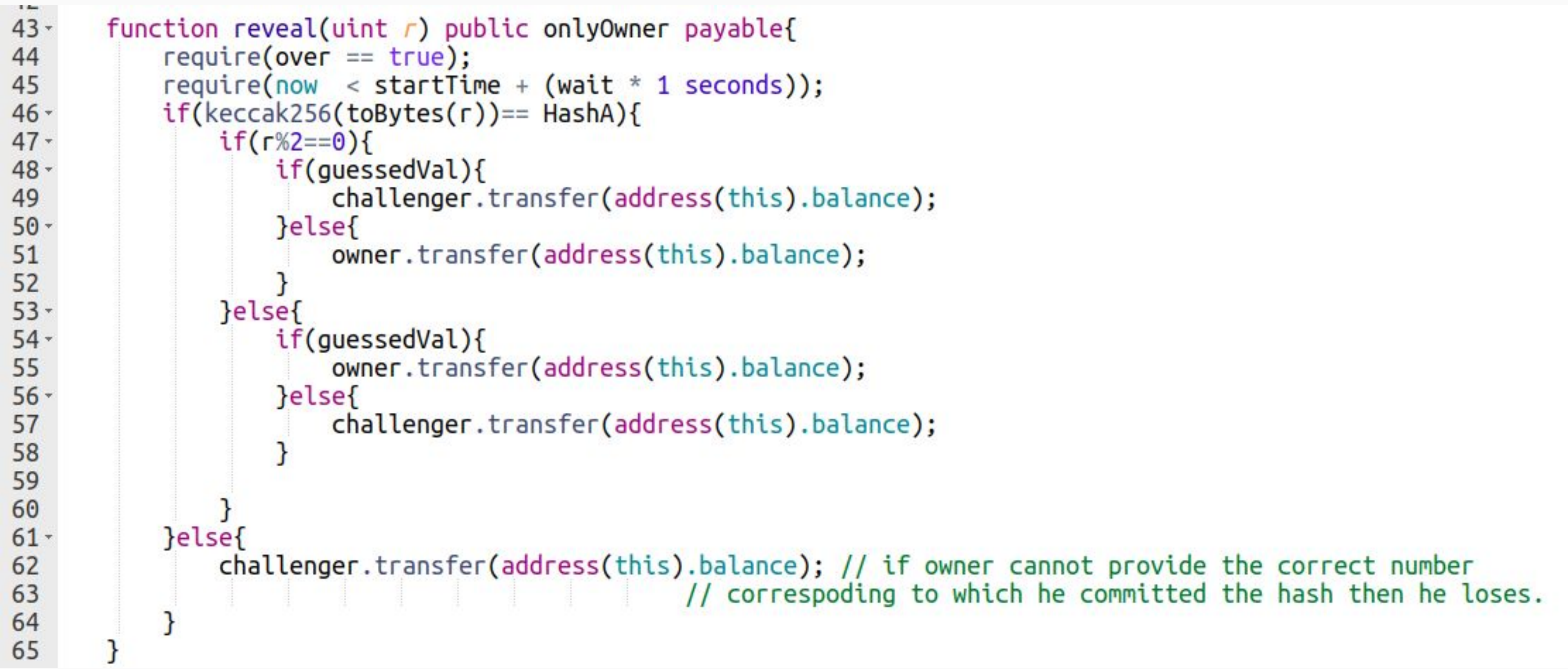
shorturl.at/iEMS2



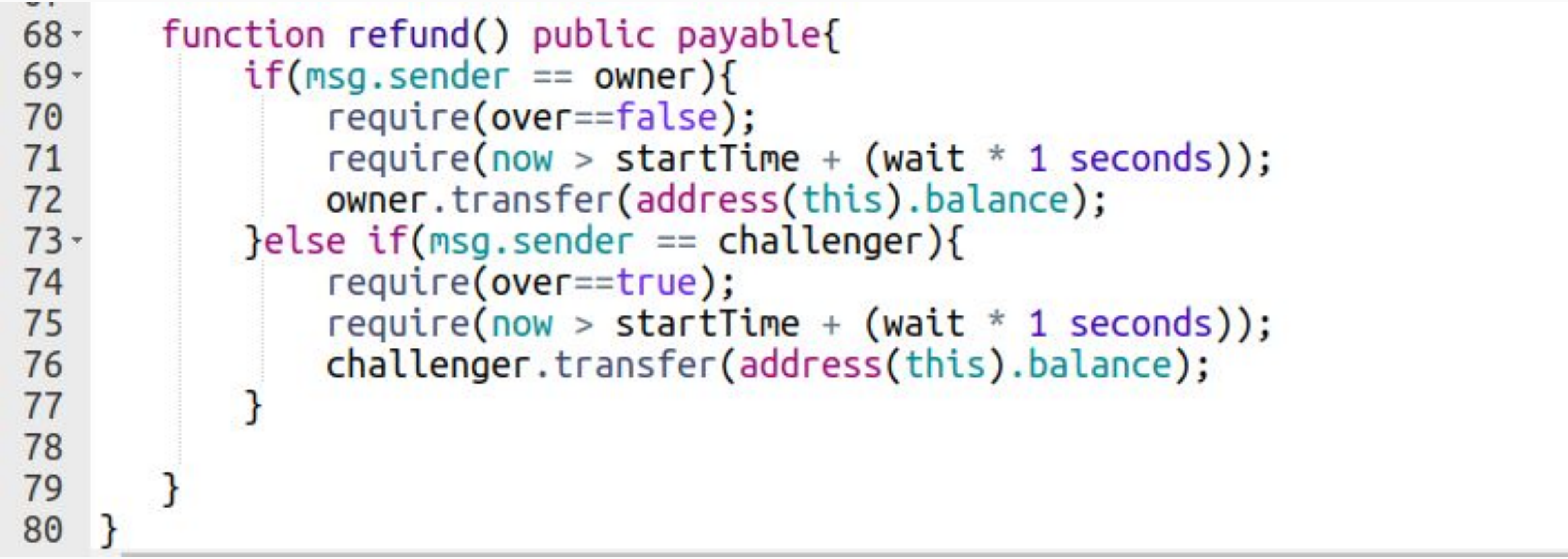
```
pragma solidity >=0.4.22 <0.6.0;

contract coin_toss {
    bytes32 HashA;
    uint amt;
    uint startTime;
    address payable owner;      // we might want to send money hence address payable
    address payable challenger;
    bool guessedVal;  // true - even    false -  odd
    uint wait;
    bool over;

    constructor(bytes32 _a) public payable{
        HashA = _a;
        amt = msg.value;
        startTime = now;
        owner = msg.sender;
        wait = 1000;
        over = false;
    }

   modifier onlyOwner{
       require(msg.sender==owner);
       _;
   }
```



```
    function toBytes(uint256 x) private pure returns (bytes memory b) {
         b = new bytes(32);
         assembly { mstore(add(b, 32), x) }
     }

    function guess(bool _g) public payable{
        require(msg.value >= amt);
        require(now < startTime + (wait * 1 seconds));
        require(over == false);
        startTime = now;
        over = true;
        challenger = msg.sender;
        guessedVal = _g;

    }

```



```
    function reveal(uint r) public onlyOwner payable{
        require(over == true);
        require(now  < startTime + (wait * 1 seconds));
        if(keccak256(toBytes(r))== HashA){
            if(r%2==0){
                if(guessedVal){
                    challenger.transfer(address(this).balance);
                }else{
                    owner.transfer(address(this).balance);
                }
            }else{
                if(guessedVal){
                    owner.transfer(address(this).balance);
                }else{
                    challenger.transfer(address(this).balance);
                }

            }
        }else{
            challenger.transfer(address(this).balance); // if owner cannot provide the correct number
                                             // correspoding to which he committed the hash then he loses.
        }
    }
```

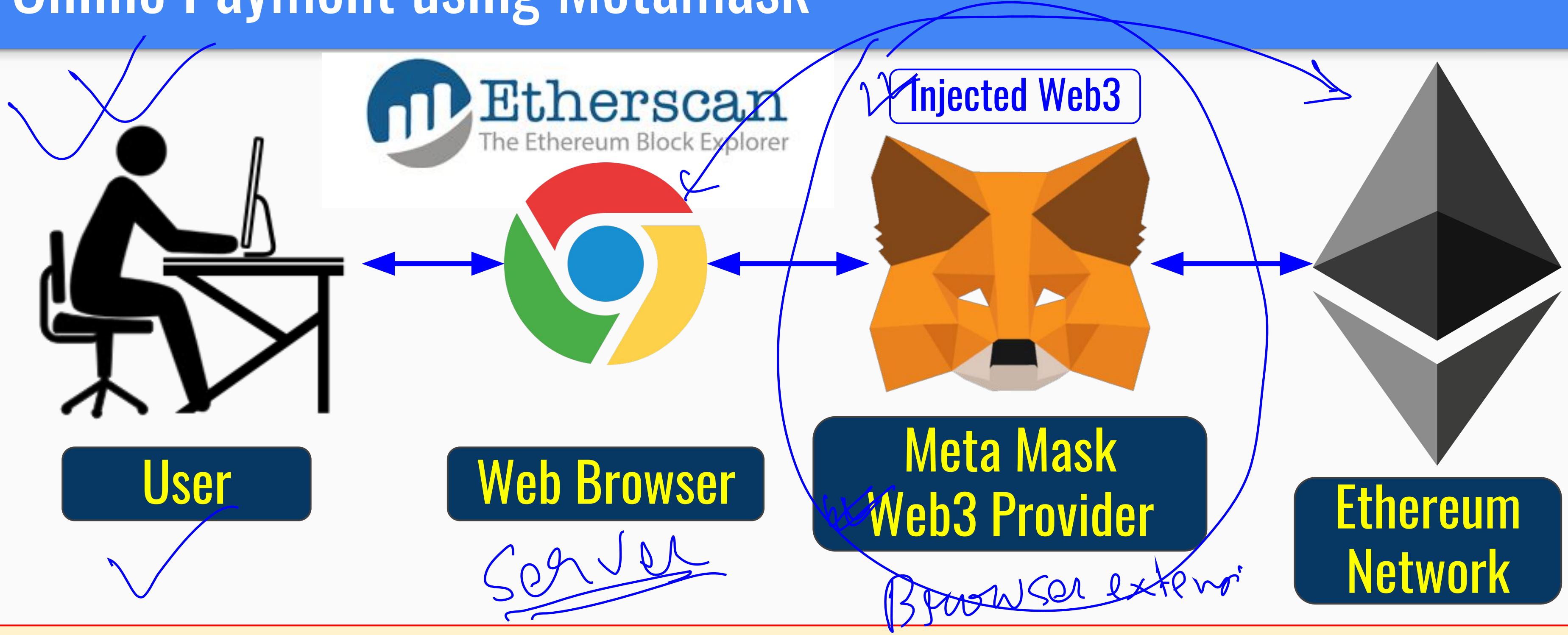


```
    function refund() public payable{
        if(msg.sender == owner){
            require(over==false);
            require(now > startTime + (wait * 1 seconds));
            owner.transfer(address(this).balance);
        }else if(msg.sender == challenger){
            require(over==true);
            require(now > startTime + (wait * 1 seconds));
            challenger.transfer(address(this).balance);
        }

    }
}
```

Overview

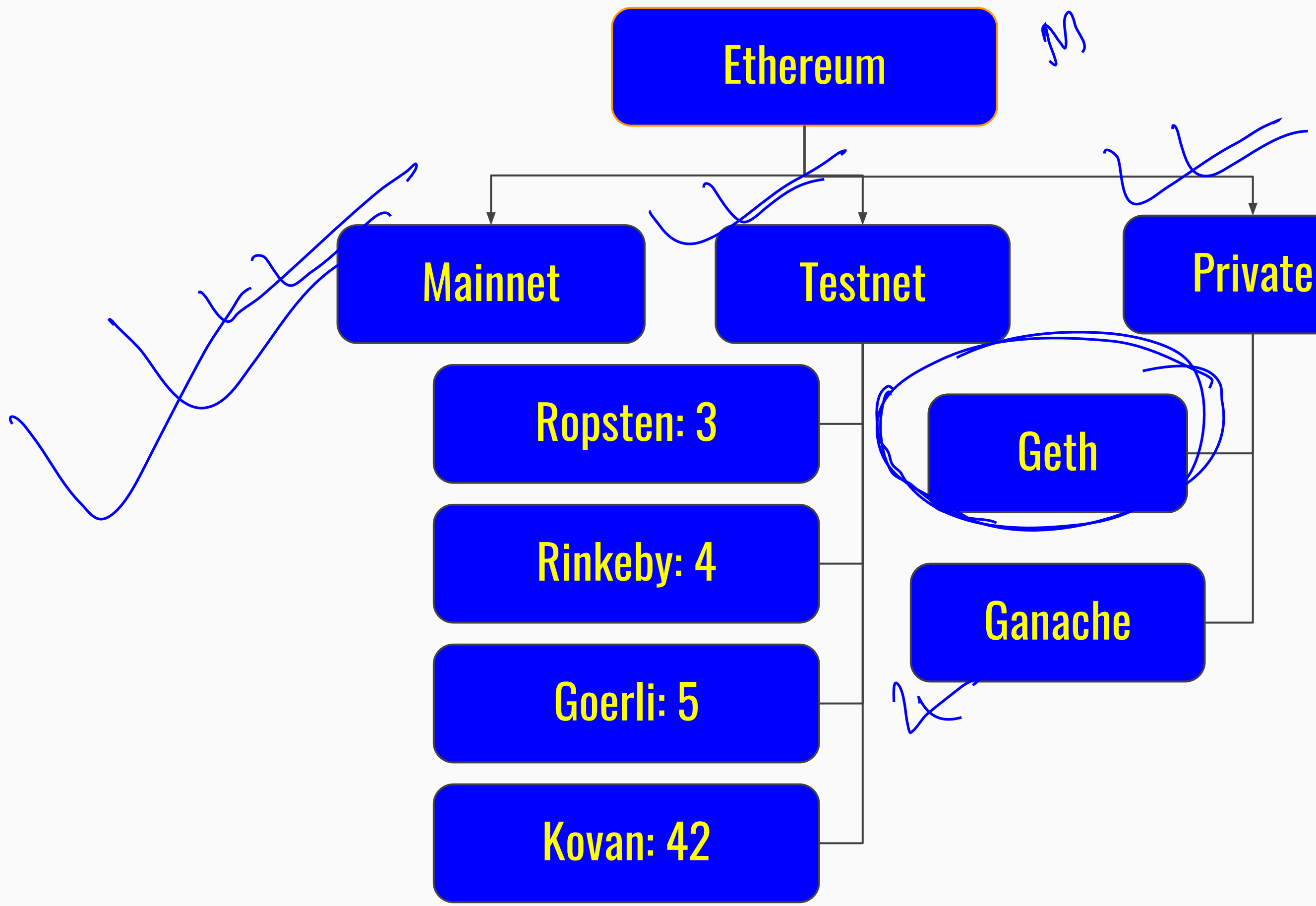
- **Online Payment using Metamask**
- **Types of Ethereum Networks**
- **Smart Contract Deployment using Metamask**
- **Smart Contract Deployment using Ganache**
- **Offline Development**

Online Payment using Metamask

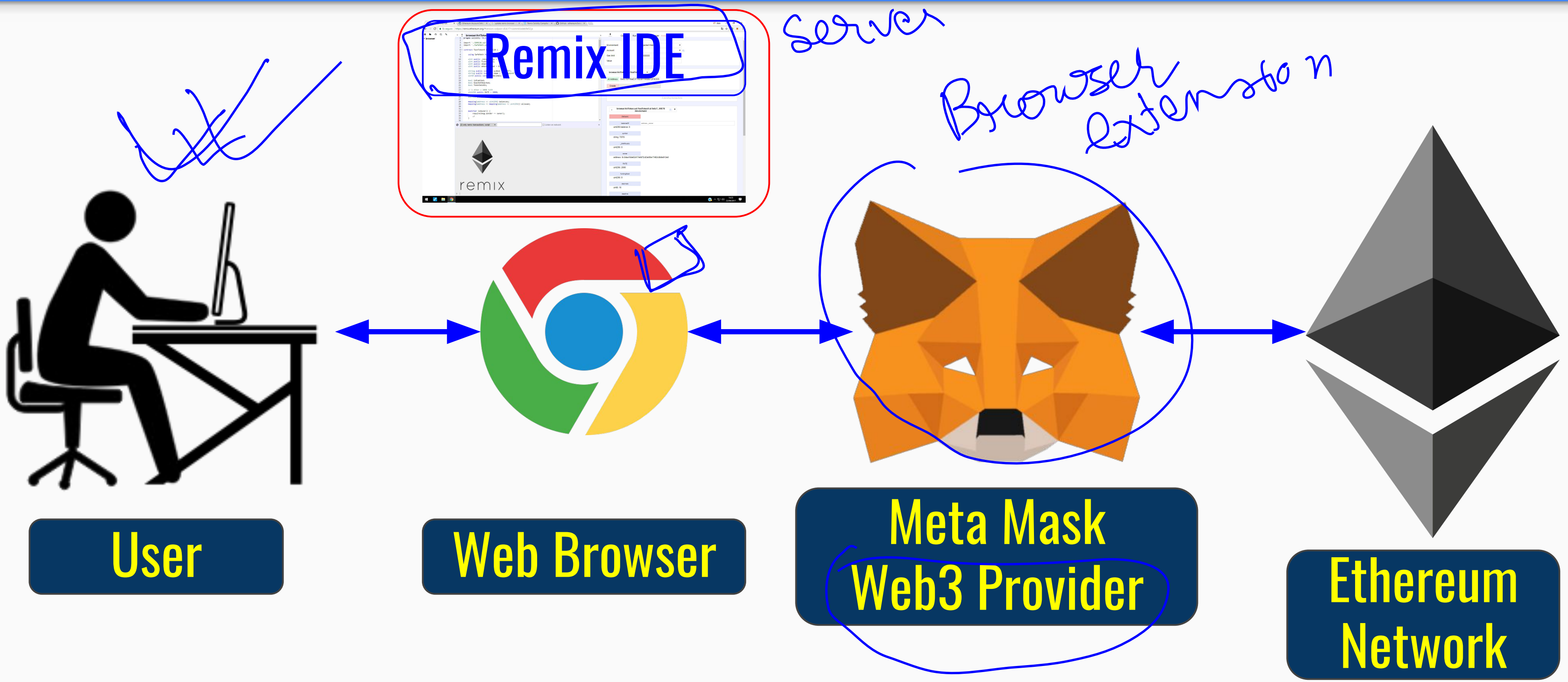


- MetaMask is a browser extension that acts as a bridge between browsers and Ethereum.
- It enables users to execute Ethereum dApps in their browser directly without running a full Ethereum node.
- MetaMask allows users to store, send, receive, and facilitate interactions with the Ethereum network.

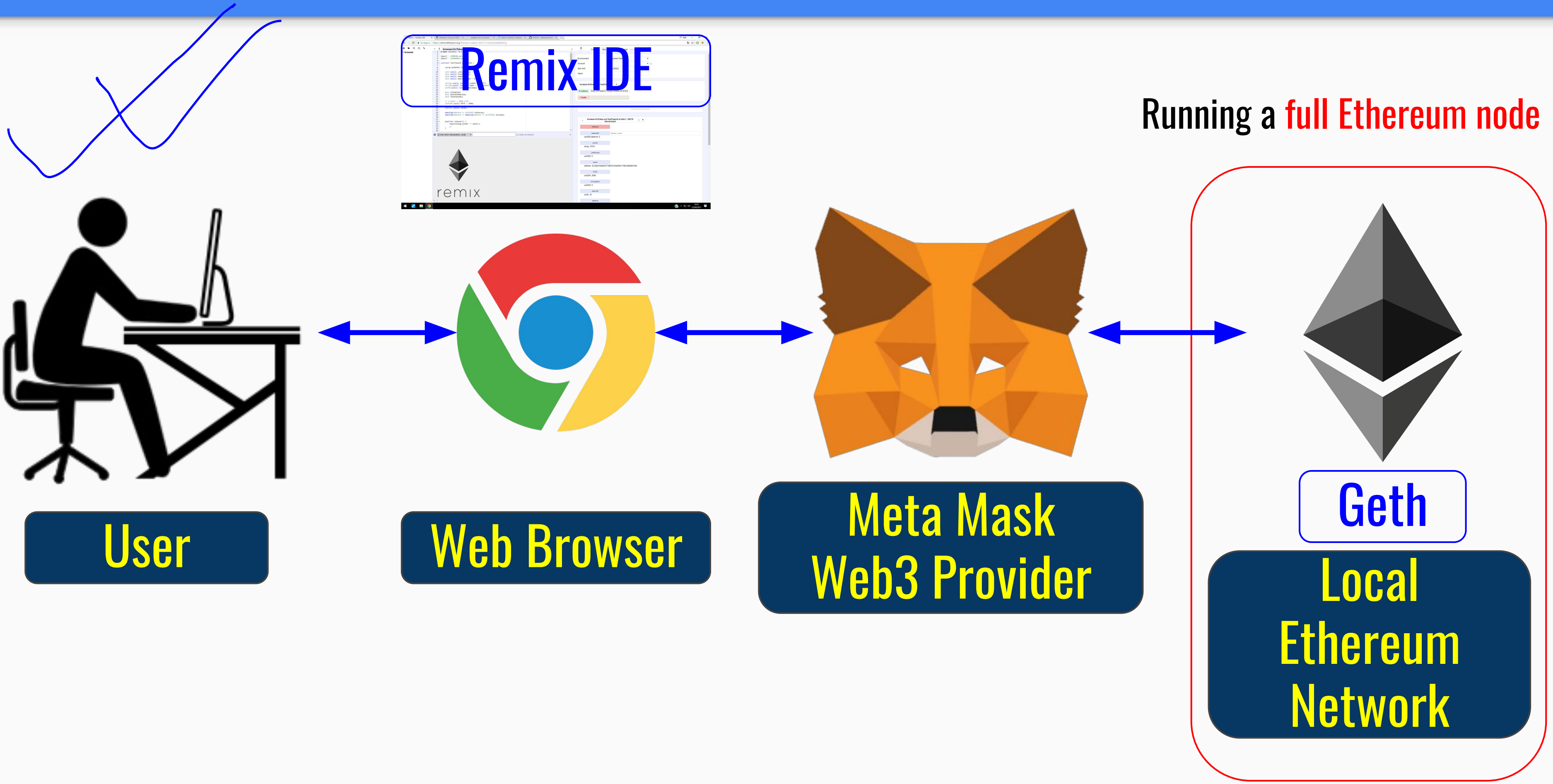
Ethereum Test Networks



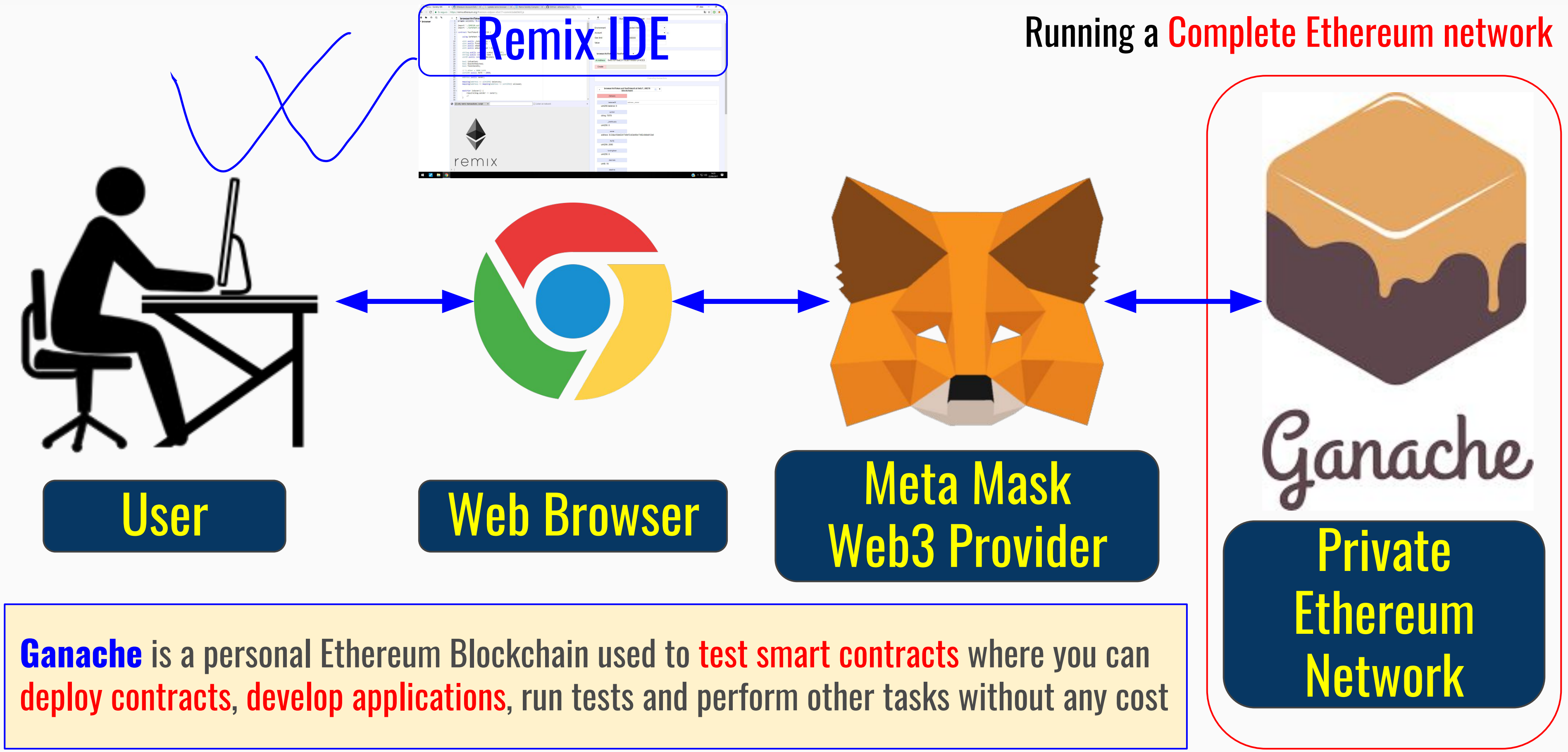
Smart Contract Deployment using Metamask



Smart Contract Deployment using Geth



Smart Contract Deployment using Ganache

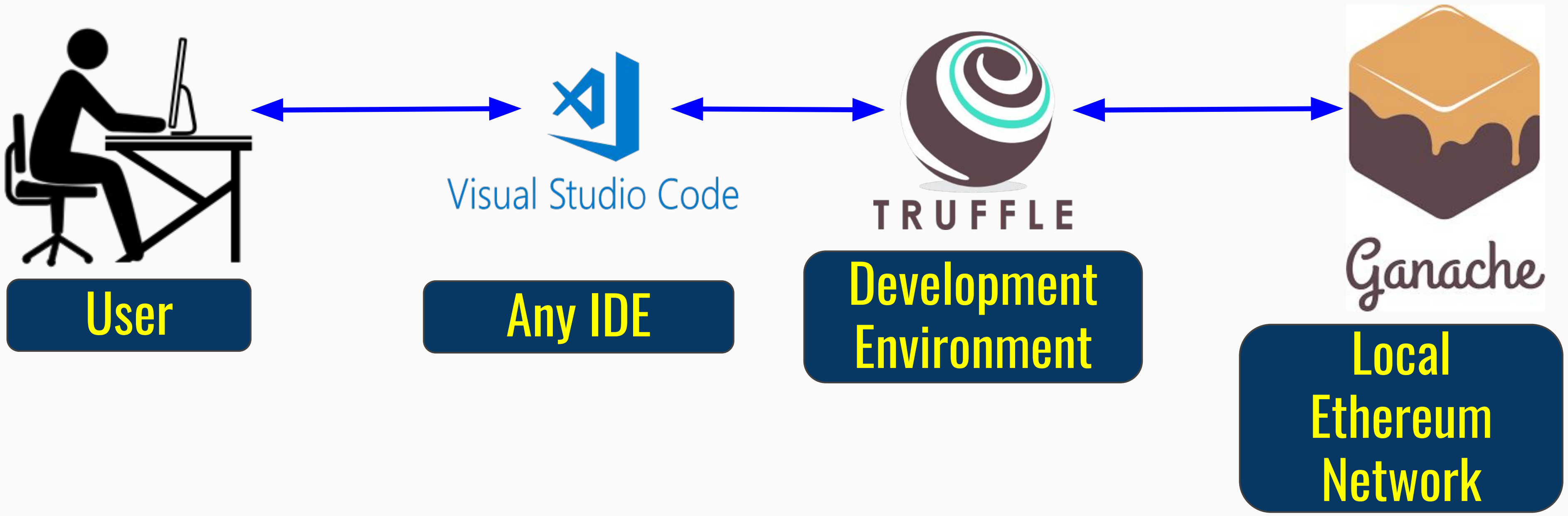


Ganache is a personal Ethereum Blockchain used to test smart contracts where you can deploy contracts, develop applications, run tests and perform other tasks without any cost

Online vs Offline Development

	Online	Offline
Solidity IDE	Remix IDE	Visual Studio Code (Any IDE)
Solidity Compiler Contract Interface	Remix Etherscan	Truffle
Blockchain	JavaScript VM Injected Web3 (Metamask)	Ganache

Offline Development



Truffle Suite is a development environment (Compiling Contracts, Deploying Contracts, Creating front-end for DApps and Testing) based on Ethereum Blockchain, used to develop DApps (Distributed Applications).

Offline Development

```
$ sudo apt-get install curl
$ curl -sL https://deb.nodesource.com/setup_12.x | sudo -E bash -
$ sudo apt-get install nodejs
$ node -v
$ npm -v
```

1. https://tecadmin.net/install-latest-nodejs-npm-on-ubuntu/
2. https://metamask.io/
3. https://www.trufflesuite.com/truffle
4. https://www.trufflesuite.com/ganache

METAMASK + Ganache + TRUFFLE

Kenneth Hu

Tools

TABLE I: Tools used for Ethereum Blockchain Developement

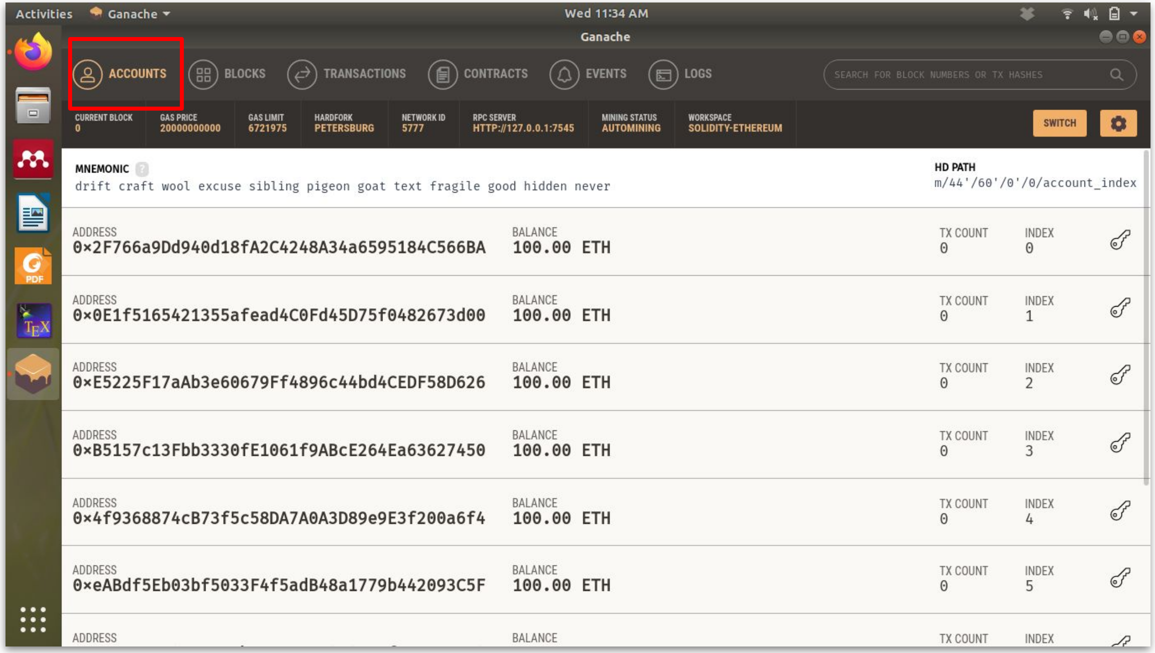
Tool Name	**Description**
Solidity	Solidity is the most popular programming language used to write smart contracts to run on the Ethereum blockchain. It is a high level language which when compiled gets converted to EVM (Ethereum Virtual Machine) byte code.
Truffle	Truffle is the gold standard for providing the building blocks to quickly create, compile, deploy, and test blockchain apps. It is a development environment, testing framework and asset pipeline for blockchains using the Ethereum Virtual Machine (EVM).
Metamask	Metamask is a dedicated way to allows you to run Ethereum Apps right in your browser without running a full Ethereum node. Users securely manage their Ethereum accounts and private keys, and use these accounts to interact with websites that are using Web3.js. **Note**: Metamask uses Infuras servers under the hood as a Web3 provider but it also gives the user the option to choose their own provider.
Influra	An IaaS (Infrastructure-as-a-Service) product offering developers a suite of tools to connect their apps to the Ethereum network and other decentralized platforms. Metamask,CryptoKitties, UJO, uPort - all utilize Infuras APIs to connect their applications to the Ethereum network. It includes an easy to use API and developer tools to provide secure, reliable, and scalable access to Ethereum and IPFS.
Web3	Web3 is an interface you use to interact with blockchain through JSON-RPC. It is a library which can be used to interact with an Ethereum node from your web based DApp. Remember each node on the network contains a copy of the blockchain. When you want to call a function on a smart contract, you need to query one of these nodes and tell it the address of the smart contract and the function you want to call.

Introduction to Ganache

Ganache

- ➔ Ganache is a test blockchain network for Ethereum development used to deploy contracts, develop your applications, and run tests.
- ➔ Download: https://www.trufflesuite.com/ganache

Ganache



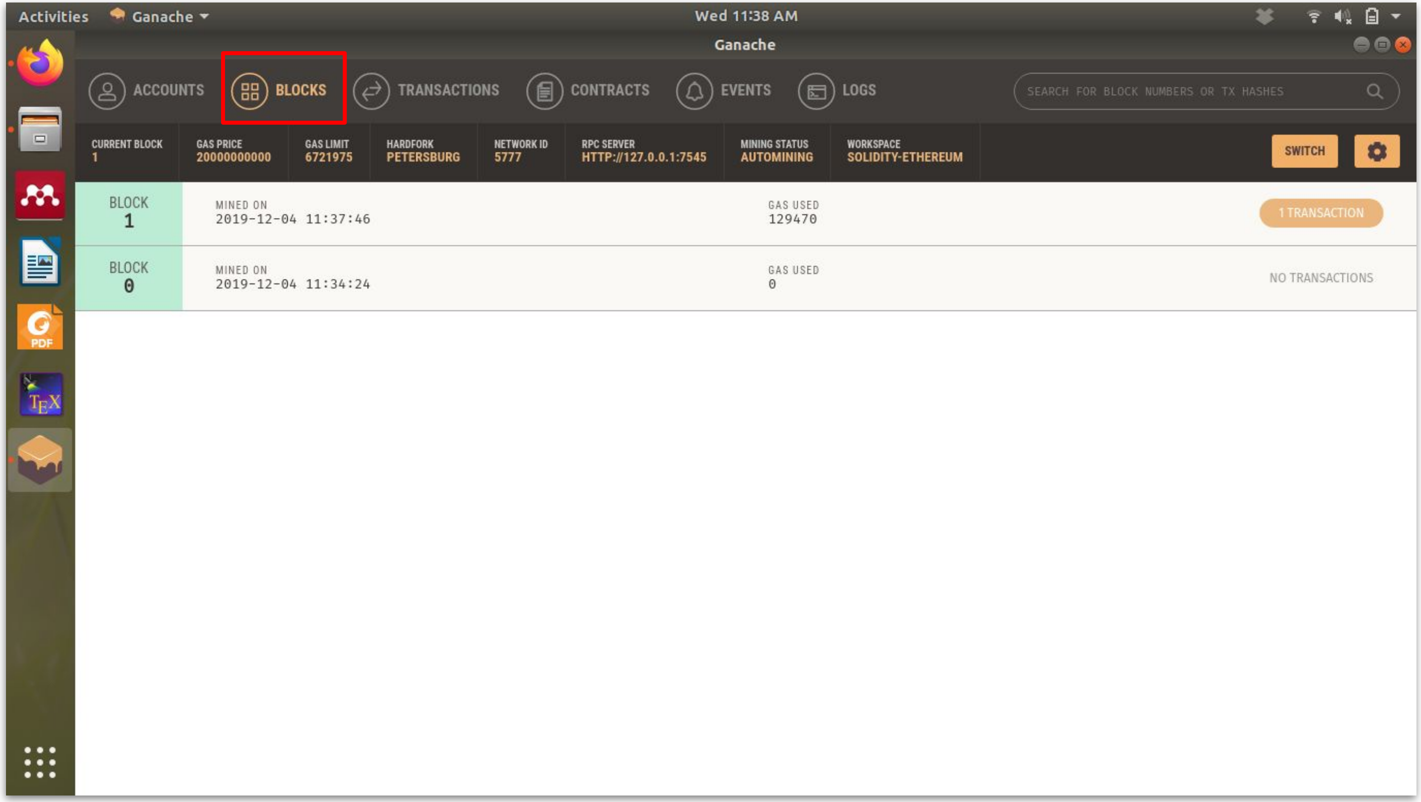
CURRENT BLOCK	GAS PRICE	GAS LIMIT	HARDFORK	NETWORK ID	RPC SERVER	MINING STATUS	WORKSPACE
0	20000000000	6721975	PETERSBURG	5777	HTTP://127.0.0.1:7545	AUTOMINING	SOLIDITY-ETHEREUM

MNEMONIC
drift craft wool excuse sibling pigeon goat text fragile good hidden never

HD PATH
m/44'/60'/0'/0/account_index

ADDRESS	BALANCE	TX COUNT	INDEX
0×2F766a9Dd940d18fA2C4248A34a6595184C566BA	100.00 ETH	0	0
0×0E1f5165421355afead4C0Fd45D75f0482673d00	100.00 ETH	0	1
0×E5225F17aAb3e60679Ff4896c44bd4CEDF58D626	100.00 ETH	0	2
0×B5157c13Fbb3330fE1061f9ABcE264Ea63627450	100.00 ETH	0	3
0×4f9368874cB73f5c58DA7A0A3D89e9E3f200a6f4	100.00 ETH	0	4
0×eABdf5Eb03bf5033F4f5adB48a1779b442093C5F	100.00 ETH	0	5

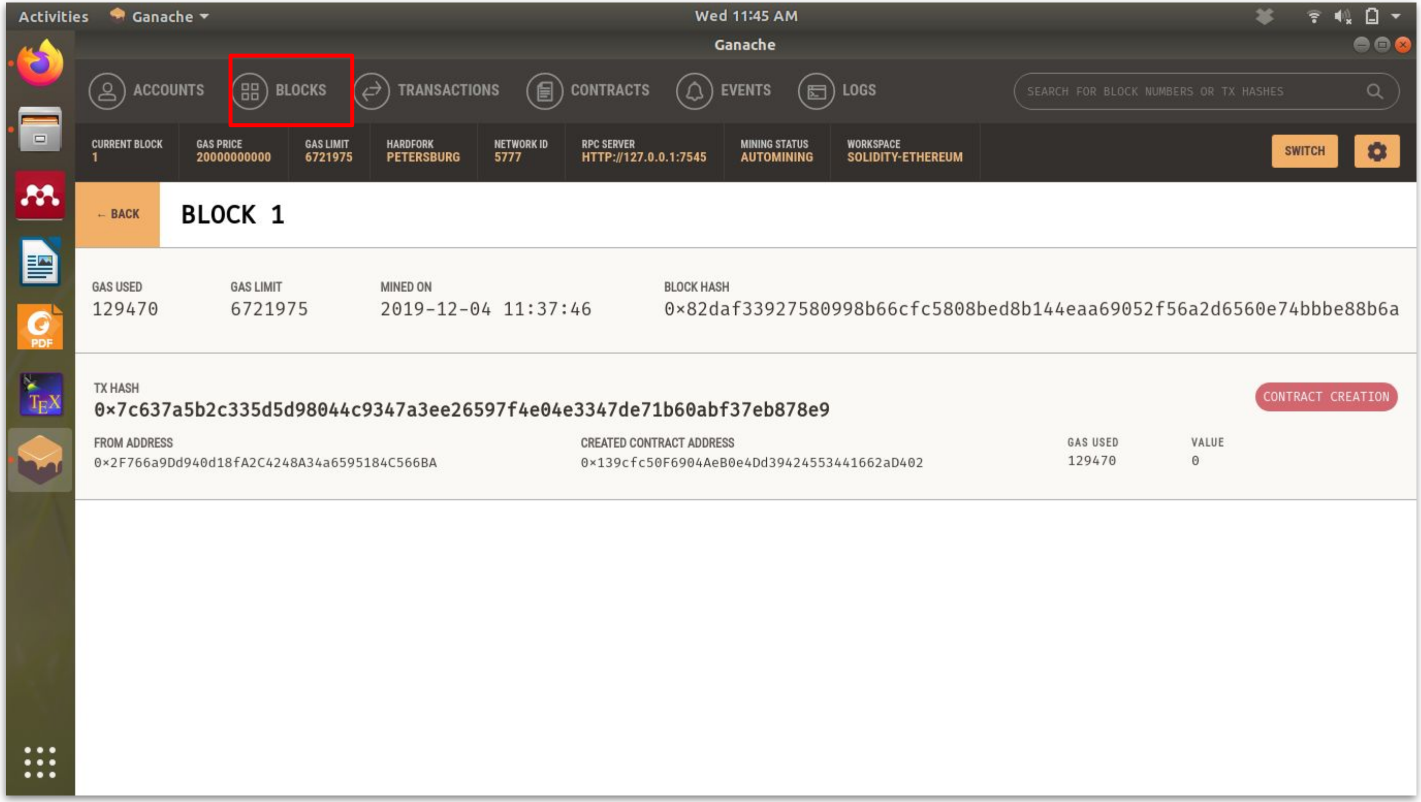
Ganache



CURRENT BLOCK 1 | GAS PRICE 20000000000 | GAS LIMIT 6721975 | HARDFORK PETERSBURG | NETWORK ID 5777 | RPC SERVER HTTP://127.0.0.1:7545 | MINING STATUS AUTOMINING | WORKSPACE SOLIDITY-ETHEREUM

BLOCK 1	MINED ON 2019-12-04 11:37:46	GAS USED 129470	1 TRANSACTION
BLOCK 0	MINED ON 2019-12-04 11:34:24	GAS USED 0	NO TRANSACTIONS

Ganache



CURRENT BLOCK 1
GAS PRICE 20000000000
GAS LIMIT 6721975
HARDFORK PETERSBURG
NETWORK ID 5777
RPC SERVER HTTP://127.0.0.1:7545
MINING STATUS AUTOMINING
WORKSPACE SOLIDITY-ETHEREUM

BLOCK 1

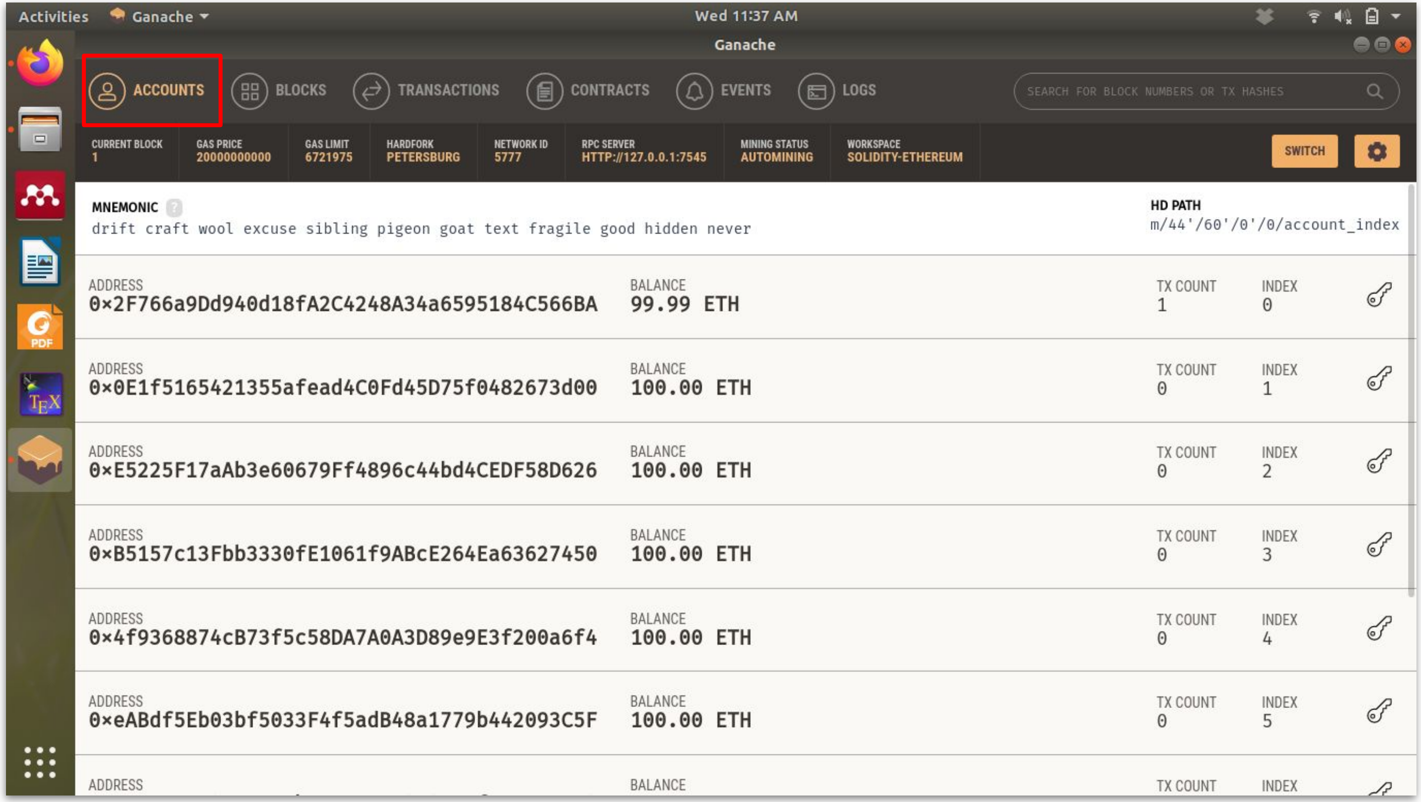
GAS USED	GAS LIMIT	MINED ON	BLOCK HASH
129470	6721975	2019-12-04 11:37:46	0×82daf33927580998b66cfc5808bed8b144eaa69052f56a2d6560e74bbbe88b6a

TX HASH
0×7c637a5b2c335d5d98044c9347a3ee26597f4e04e3347de71b60abf37eb878e9

CONTRACT CREATION

FROM ADDRESS	CREATED CONTRACT ADDRESS	GAS USED	VALUE
0×2F766a9Dd940d18fA2C4248A34a6595184C566BA	0×139cfc50F6904AeB0e4Dd39424553441662aD402	129470	0

Ganache



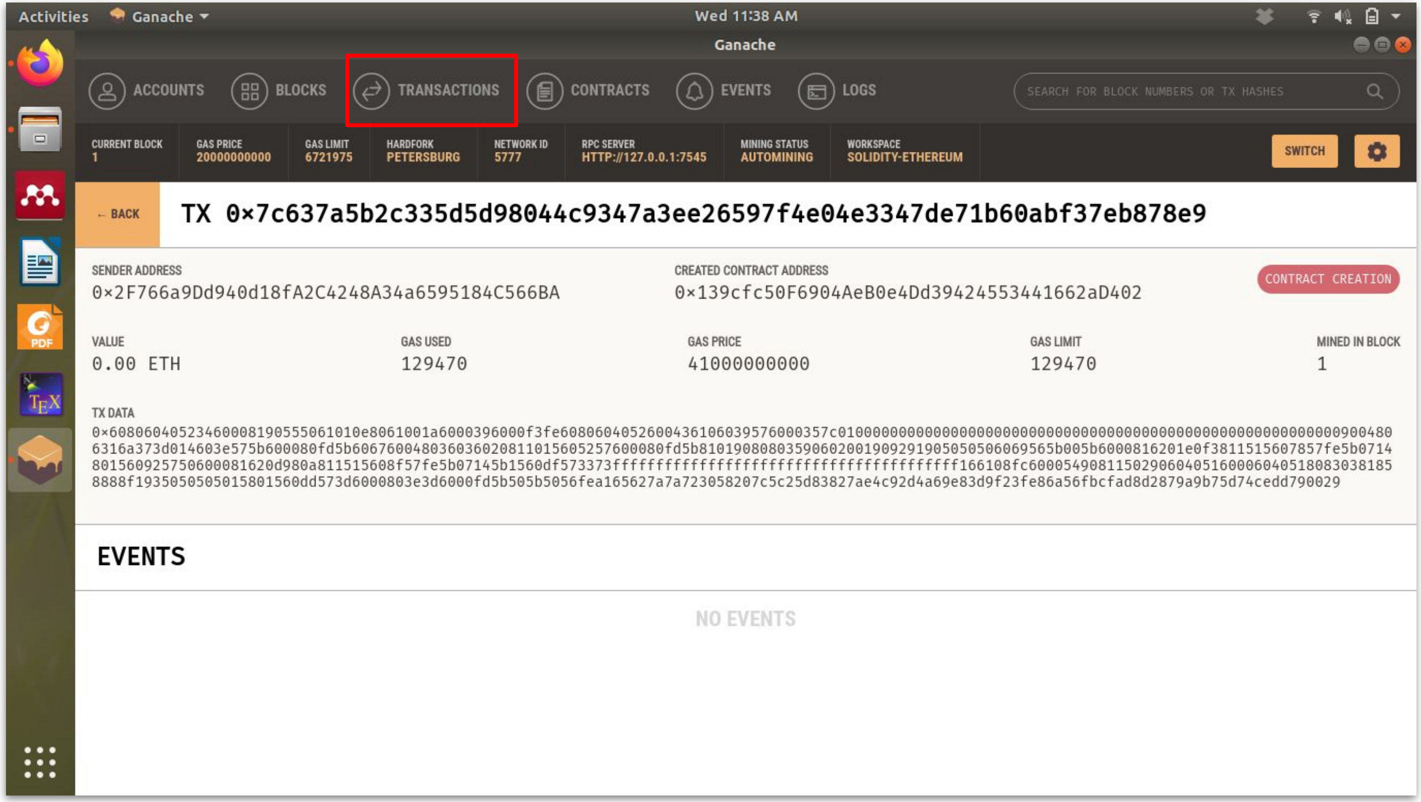
CURRENT BLOCK	GAS PRICE	GAS LIMIT	HARDFORK	NETWORK ID	RPC SERVER	MINING STATUS	WORKSPACE
1	20000000000	6721975	PETERSBURG	5777	HTTP://127.0.0.1:7545	AUTOMINING	SOLIDITY-ETHEREUM

MNEMONIC
drift craft wool excuse sibling pigeon goat text fragile good hidden never

HD PATH
m/44'/60'/0'/0/account_index

ADDRESS	BALANCE	TX COUNT	INDEX
0×2F766a9Dd940d18fA2C4248A34a6595184C566BA	99.99 ETH	1	0
0×0E1f5165421355afead4C0Fd45D75f0482673d00	100.00 ETH	0	1
0×E5225F17aAb3e60679Ff4896c44bd4CEDF58D626	100.00 ETH	0	2
0×B5157c13Fbb3330fE1061f9ABcE264Ea63627450	100.00 ETH	0	3
0×4f9368874cB73f5c58DA7A0A3D89e9E3f200a6f4	100.00 ETH	0	4
0×eABdf5Eb03bf5033F4f5adB48a1779b442093C5F	100.00 ETH	0	5
ADDRESS	BALANCE	TX COUNT	INDEX

Ganache



CURRENT BLOCK 1 | GAS PRICE 20000000000 | GAS LIMIT 6721975 | HARDFORK PETERSBURG | NETWORK ID 5777 | RPC SERVER HTTP://127.0.0.1:7545 | MINING STATUS AUTOMINING | WORKSPACE SOLIDITY-ETHEREUM

TX 0×7c637a5b2c335d5d98044c9347a3ee26597f4e04e3347de71b60abf37eb878e9

SENDER ADDRESS
0×2F766a9Dd940d18fA2C4248A34a6595184C566BA

CREATED CONTRACT ADDRESS
0×139cfc50F6904AeB0e4Dd39424553441662aD402

CONTRACT CREATION

VALUE	GAS USED	GAS PRICE	GAS LIMIT	MINED IN BLOCK
0.00 ETH	129470	41000000000	129470	1

TX DATA

```
0×608060405234600081905550610 10e8061001a6000396000f3fe6080604052600436106039576000357c0100000000000000000000000000000000000000000000000000000000900480
6316a373d014603e575b600080fd5b606760048036036020811015605257600080fd5b8101908080359060200190929190505050606956 5b005b6000816201e0f3811515607857fe5b0714
8015609257506000816 20d980a811515608f57fe5b07145b1560df573373ffffffffffffffffffffffffffffffffffffffff166108fc600054908115029060405160006040518083038185
8888f19350505050158015 60dd573d6000803e3d6000fd5b505b5056fea165627a7a723058207c5c25d83827ae4c92d4a69e83d9f23fe86a56fbcfad8d2879a9b75d74cedd790029
```

EVENTS

NO EVENTS

Full Implementation of Ethereum

- ➔ Open source code
- ➔ Available in C++/Go/Python
- ➔ Download: https://github.com/ethereum/go-ethereum

Thank you