Smart Contracts

September 23, 2019

guha.jayachandran@sjsu.edu

What’s a Contract?

What’s a Contract?

“A legally binding agreement which recognizes and governs the rights and duties of the parties to the agreement”

Pacta sunt servanda

What’s a Contract?

If this, then that

Smart Contract

If this, then that

Smart Contract

First proposed by Nick Szabo in 1994

Smart Contract

Self-executing contract

Computerized transaction protocols that execute terms of a contract

How Does it Work?

How Does it Work?

Everyone evaluate contract, consensus reached on outcome

How Does it Work?

Bitcoin - Bitcoin script

Ethereum - EVM

Some cryptocurrencies support, some don’t

Examples

Transfer value if signed

Transfer if multi-signature provided

“Tipping point” funding

Provable casino

Prediction markets

…

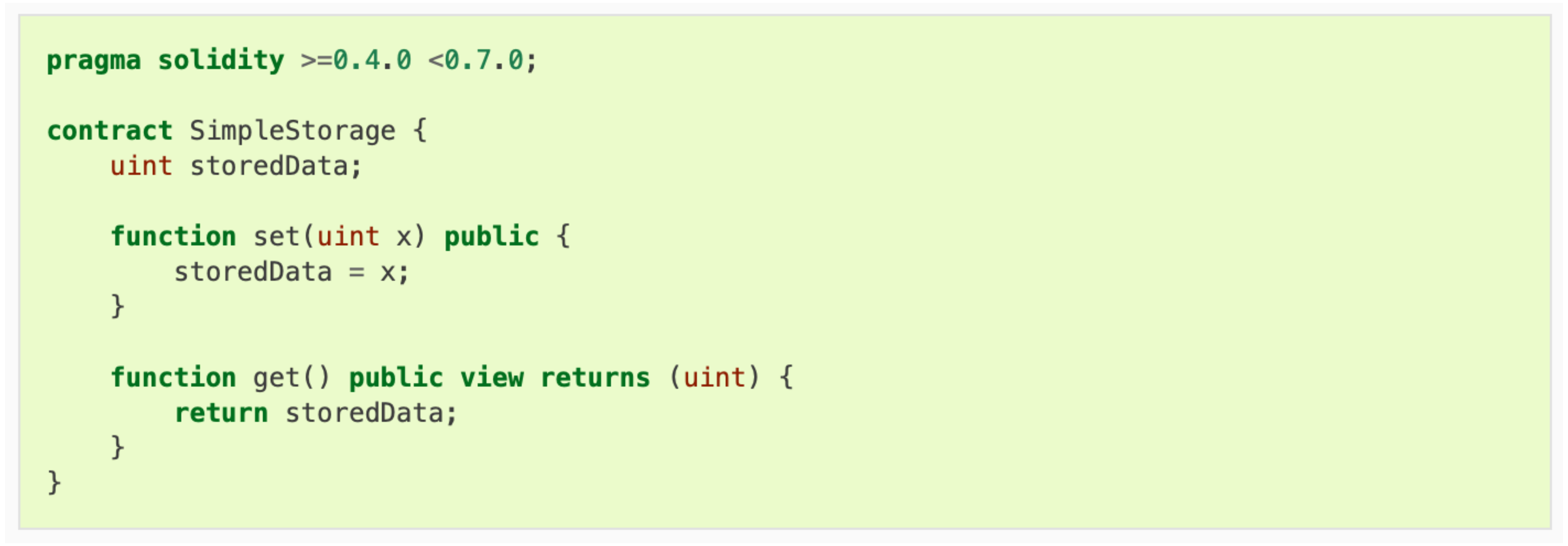
Ethereum Smart Contracts

Turing complete

Write in high-level language and compile to EVM bytecode

Solidity

Sample Solidity



```
pragma solidity >=0.4.0 <0.7.0;

contract SimpleStorage {
    uint storedData;

    function set(uint x) public {
        storedData = x;
    }

    function get() public view returns (uint) {
        return storedData;
    }
}
```

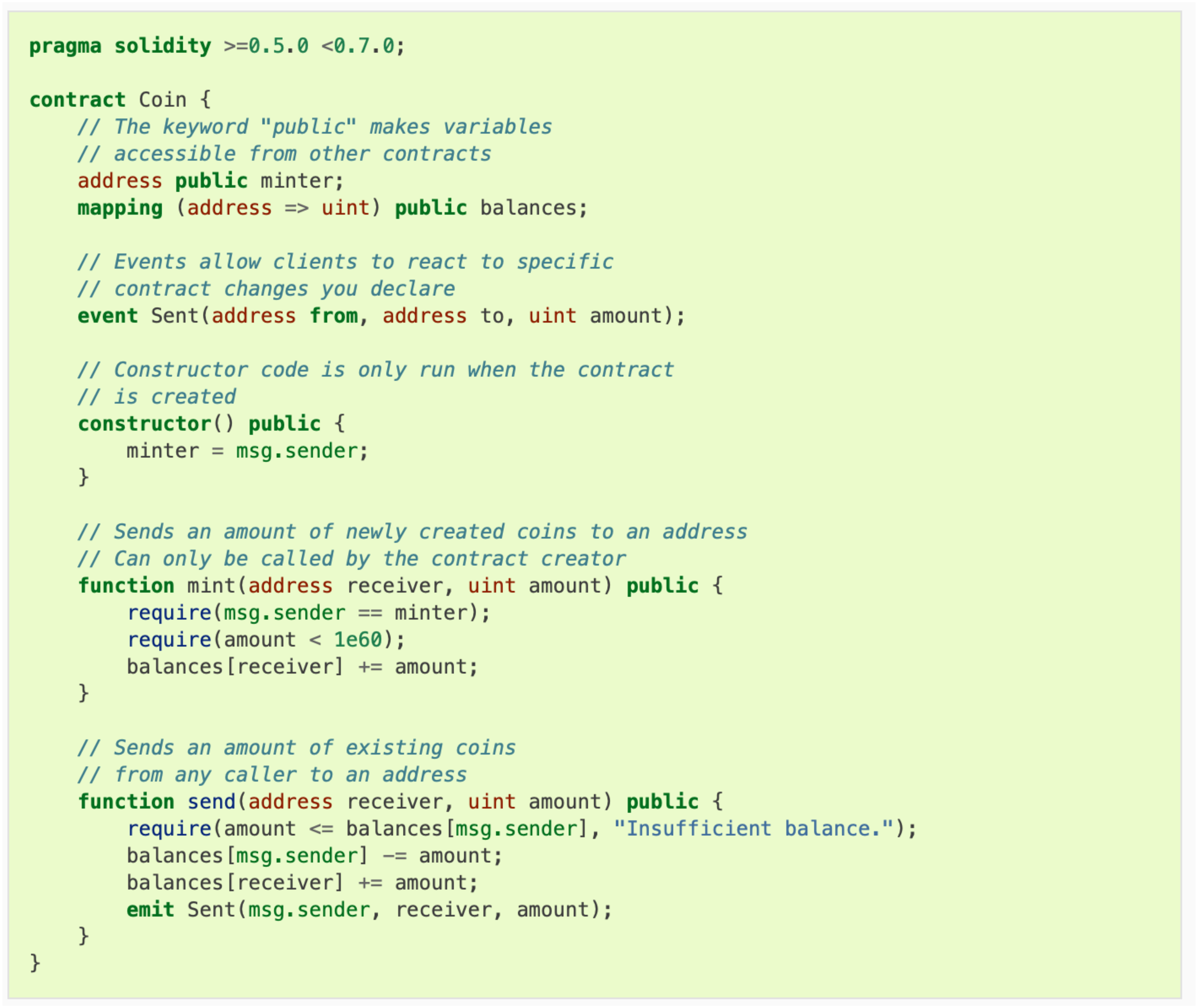
Source: solidity.readthedocs.io

Digression: UTXO vs. Account-Based

- UTXO = Unspent transaction output
  - Graph of transactions like in Bitcoin from before.
  - Your money is spread across your (potentially) many UTXOs

- Account-based
  - There is a specific account associated with your key on the blockchain
  - When you spend or receive, your account is updated

Sample Solidity



```
pragma solidity >=0.5.0 <0.7.0;

contract Coin {
    // The keyword "public" makes variables
    // accessible from other contracts
    address public minter;
    mapping (address => uint) public balances;

    // Events allow clients to react to specific
    // contract changes you declare
    event Sent(address from, address to, uint amount);

    // Constructor code is only run when the contract
    // is created
    constructor() public {
        minter = msg.sender;
    }

    // Sends an amount of newly created coins to an address
    // Can only be called by the contract creator
    function mint(address receiver, uint amount) public {
        require(msg.sender == minter);
        require(amount < 1e60);
        balances[receiver] += amount;
    }

    // Sends an amount of existing coins
    // from any caller to an address
    function send(address receiver, uint amount) public {
        require(amount <= balances[msg.sender], "Insufficient balance.");
        balances[msg.sender] -= amount;
        balances[receiver] += amount;
        emit Sent(msg.sender, receiver, amount);
    }
}
```

Source: solidity.readthedocs.io

Questions

How do you prevent DOS?

Everyone runs the contract?

Bugs?

Gas

Transaction creator charged `gas_price * gas`

If you run out of gas, transaction does not complete

Dapp

Decentralized application

DAO

Decentralized Autonomous Organization

The DAO

2016

Investor-directed venture capital fund

No human directors/managers

Cross-border

Raised $150 million in crowdsale

Legality?

The DAO

Bug resulted in $50 million hack

What would you do?

Code = Law ?

Is the right thing to do to respect the transparent smart contract or the human intentions?

The DAO

Outcome:

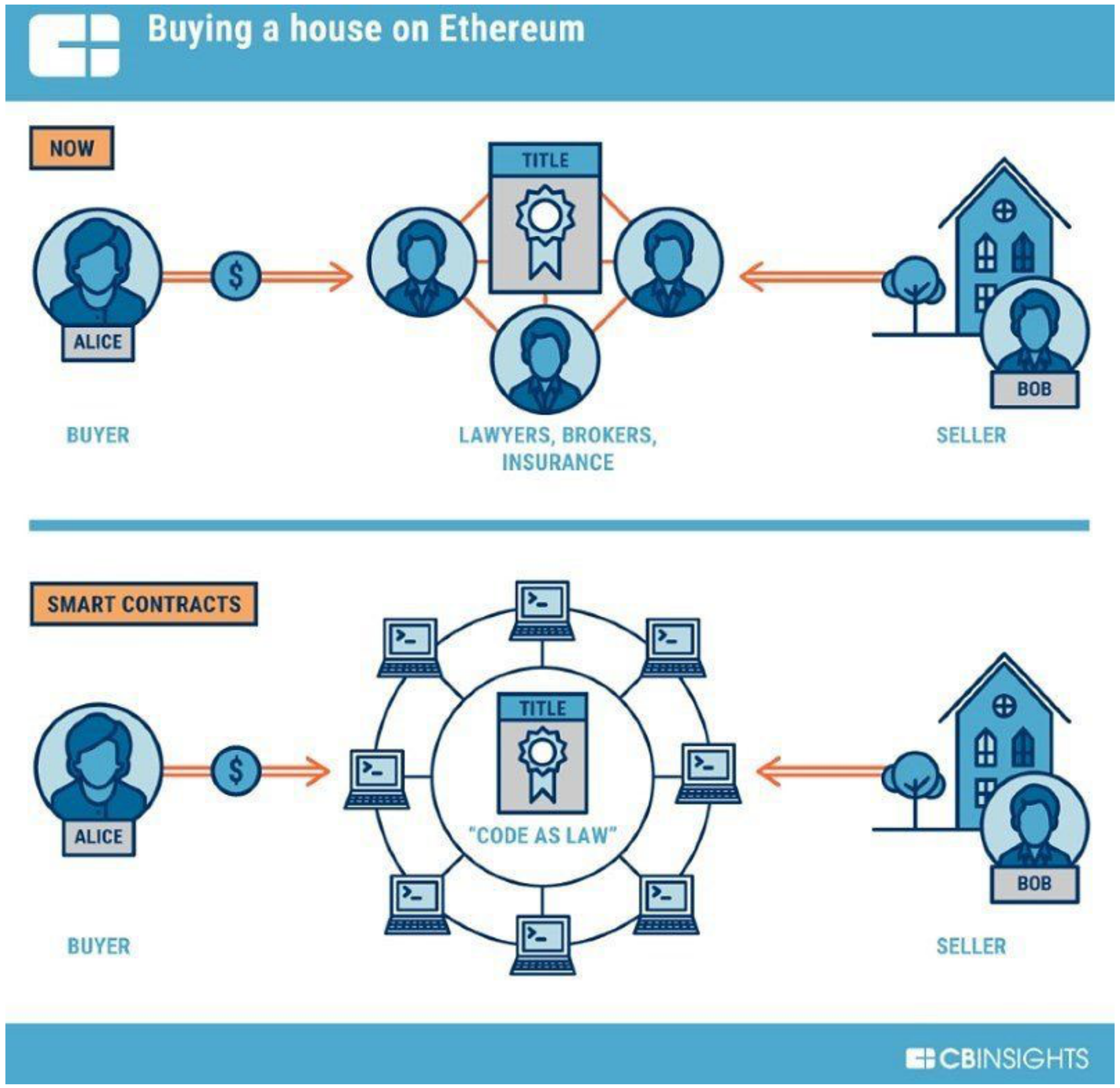
Fork of Ethereum into Ethereum (restore hacked funds) and Ethereum Classic (go along with hack)

Flaws Continued

Many hundreds of millions of dollars in smart contract hacks

Prompting efforts in formal verification

Real-World Interaction



But how would you know if the seller actually delivered possession?