# Solidity Tutorial
# How to launch your token

Joshi

# Problems we face in the open source universe with no central for profit entity

Ports Closed by Firewall for Ether Wallet

No Peer to download the blockchain for Rinkby on Ether Wallet

Faucet not giving fake ether

Suggestion: Use metamask

Smart Contracts
Ether currency
Dapps (decentralized applications)
Blockchain infrastructure and principles
Secure, decentralized, tamper-proof
Understanding cryptography
Introduction to blockchain programming
Solidity variables,  Solidity control structure, Solidity functions
Solidity inheritance, Solidity modifiers
Proxy contracts, Solidity events
Development frameworks
Truffle Framework
Web3 JavaScript API
DAO (decentralized autonomous organization)

Overview of decentralized platforms
Ethereum Blockchain App Platform
Creating the Smart Contract
Writing the Smart Contract functions
Storing dept registries
Deploying the Smart Contract
Ethereum Wallet
Holding and securing ether
Managing other crypto-assets
Creating your own cryptocurrency
Overview of tradeable digital token and coin APIs
Design & Issuing the cryptocurrency
Kickstarting a blockchain project
Initiating a trustless crowdsale
Building your own virtual organization

# Implementing a new contract

This takes time so we will do this first and then practice solidity

Getting fake ether

Deploy using fake ether

https://www.ethereum.org/token

```
pragma solidity ^0.4.2;
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 */
     function MyToken(
        uint256 initialSupplybyjoshi              ) public {
            initialSupplybyjoshi =121;
        balanceOf[msg.sender] = initialSupplybyjoshi;              // Give the creator all initial
tokens
    }    /* Send coins */
    function transfer(address _to, uint256 _value) public {
        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    }}
```

# Contents

Understanding the two tools: Metamask / Ether Wallet

Creating Wallet at metamask and receiving coin from individual and from Faucet

Creating a simple solidity program on remix

# Two Tools to launch your coin

**MetaMask** is a bridge that allows you to visit the distributed web of tomorrow in your browser today.

**Ethereum Wallet**: Run Ethereum dApps right in your browser without running a full Ethereum node

# MetaMask & Ether Wallet

Installing Metamask

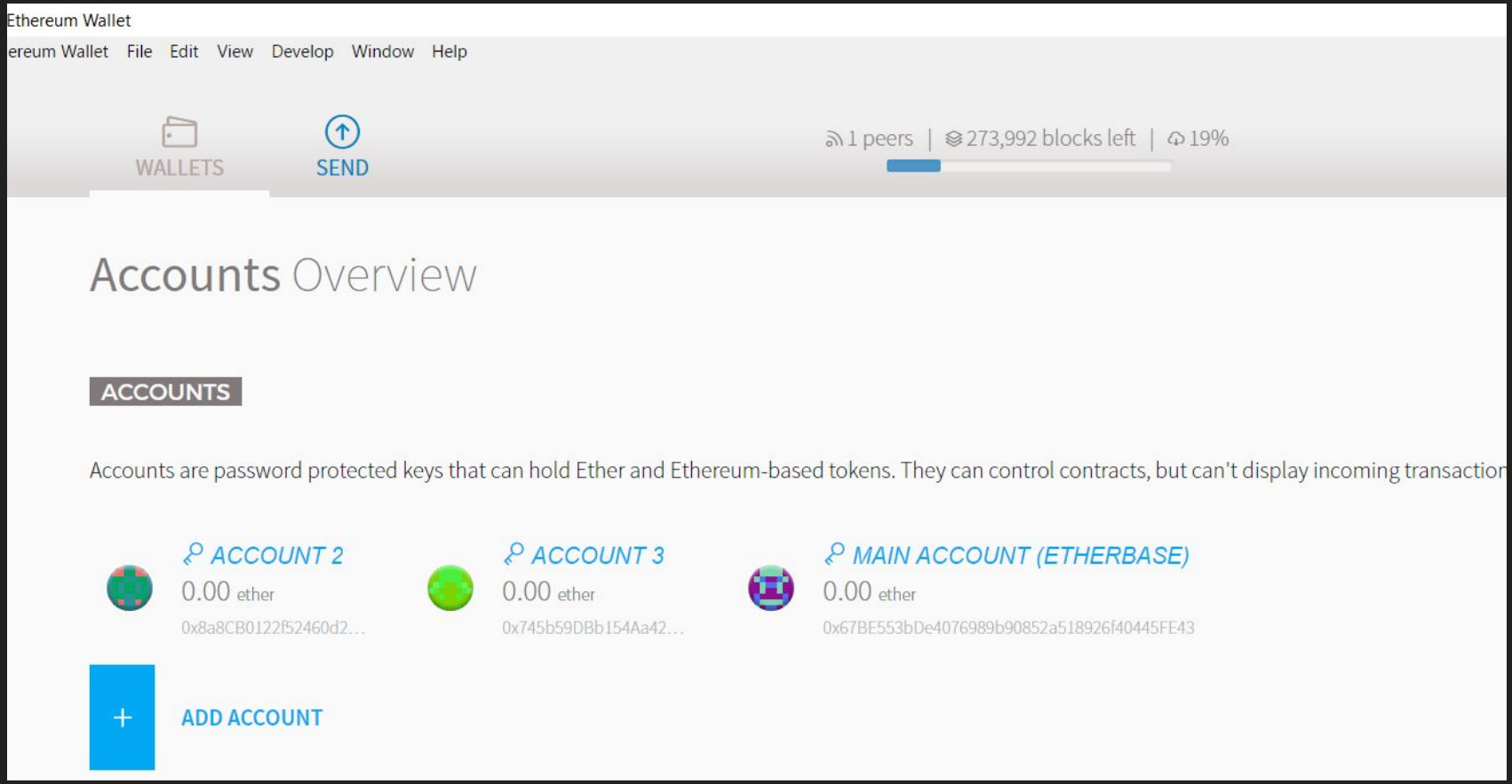
An Extension

Getting fake ether from faucet

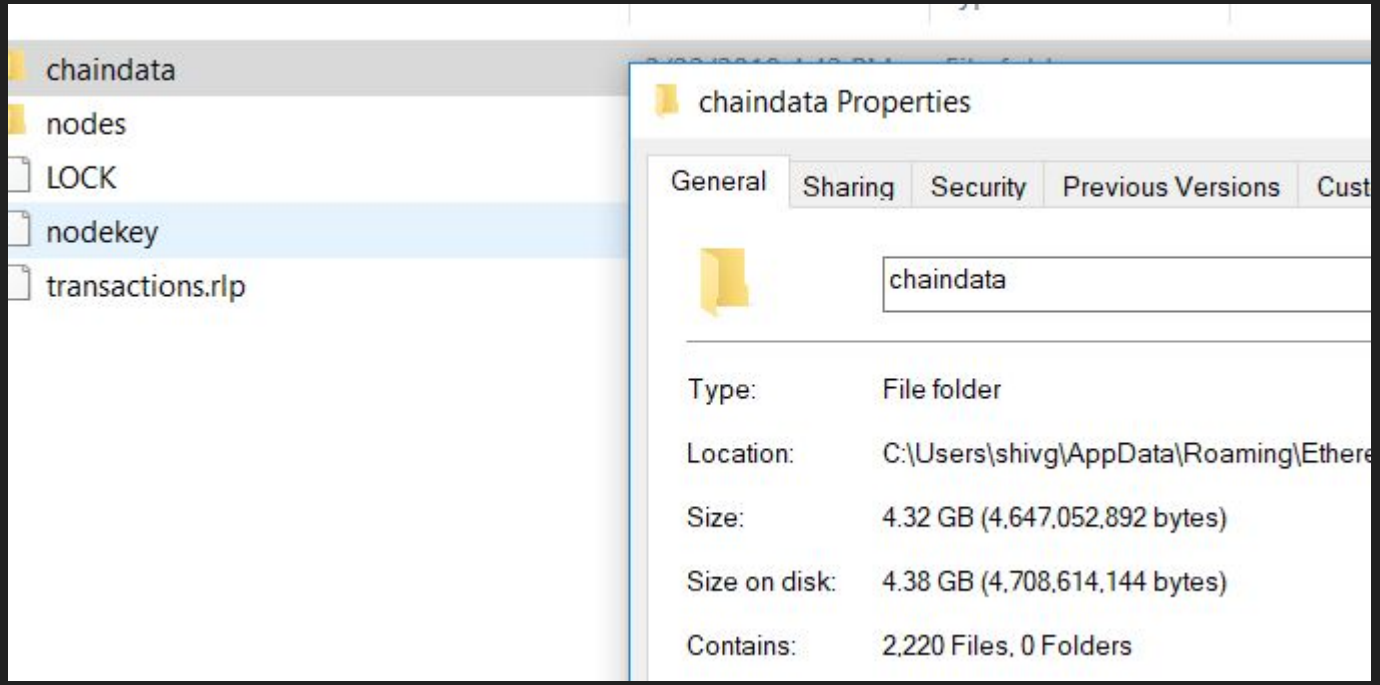
Installing Ether Wallet

Understanding the contract implementation

# Ether Wallet



# Chain Data


chaindata
nodes
LOCK
nodekey
transactions.rlp
chaindata Properties
General
Sharing
Security
Previous Versions
chaindata
Type:
File folder
Location:
C:\Users\shivg\AppData\Roaming\Ethere
Size:
4.32 GB (4,647,052,892 bytes)
Size on disk:
4.38 GB (4,708,614,144 bytes)
Contains:
2,220 Files, 0 Folders

# Public Constants in Solidity

```
pragma solidity ^0.4.0;

contract publiconst {

    string public constant symbol = "MFT";

    string public constant name = "My First Token";

    uint8 public constant decimals = 18; }
```

# First Ethereum Program to set age and name

Understanding Data types

Creating Functions

Identifying the arguments and return values

Set age vs get age

```
// What are the arguments in the functions below?

pragma solidity ^0.4.0;
contract MyFirstContract {
    string private name;
    uint private age;
    function setName(string newName) { name = newName;  }
    function getName() returns (string) {  return name; }
    function setAge(uint newAge) {     age = newAge;   }
    function getAge() returns (uint) {return age;  } }
```

# Program 2: Bank

Understanding Inheritance

# Launching your Token

Link: https://www.ethereum.org/token

# Keywords

Interface

Contract X is interface

# Libraries

## Libraries

Libraries are similar to contracts, but their purpose is that they are deployed only once at a specific address and their code is reused using the DELEGATECALL (CALLCODE until Homestead) feature of the EVM.
This means that if library functions are called, their code is executed in the context of the calling contract, i.e. this points to the calling contract, and especially the storage from the calling contract can be accessed.
As a library is an isolated piece of source code, it can only access state variables of the calling contract if they are explicitly supplied (it would have no way to name them, otherwise).
Library functions can only be called directly (i.e. without the use of DELEGATECALL) if they do not modify the state (i.e. if they are view or pure functions), because libraries are assumed to be stateless.
In particular, it is not possible to destroy a library unless Solidity’s type system is circumvented.