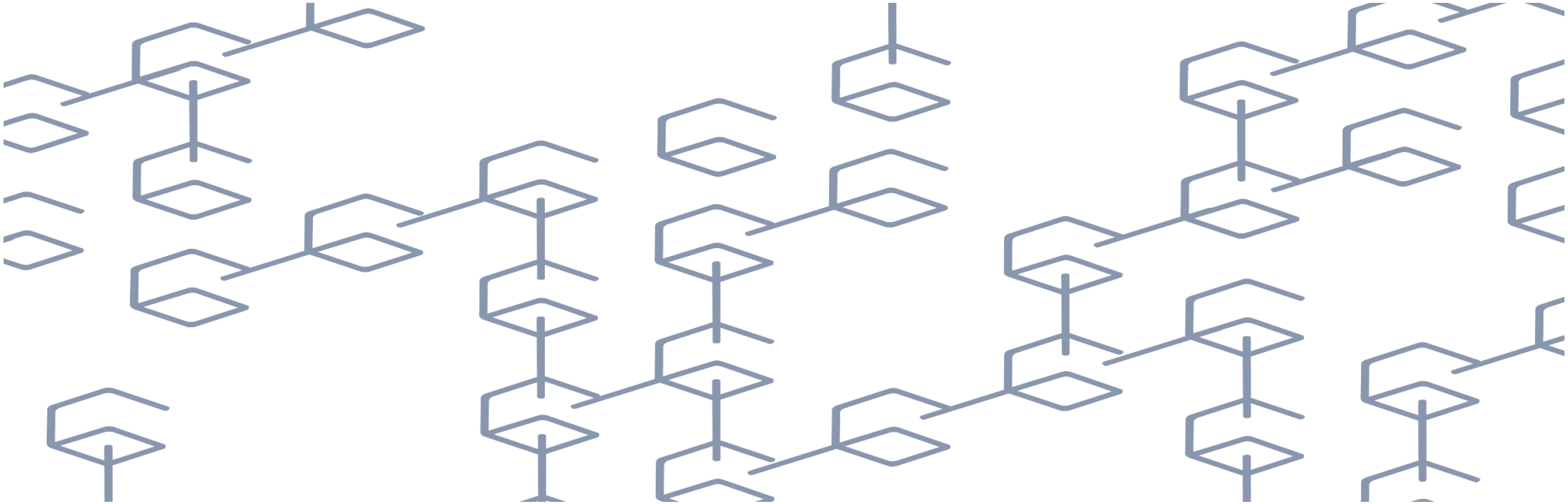
NOVA SQUARE



Blockchain Technology

By Truong Quoc Tuan

Jan, 2022

Index

NOVA SQUARE

Blockchain #1

8 weeks = 16 lessons (2 lessons x 7 weeks + 02 lessions final exam)

1. Blockchain Technology (Basic & Advanced) – 4 lessons
2. **Ethereum Blockchain Development with Solidity – 8 lessons**
3. Ethereum Blockchain Example – 2 lessons
4. Test exam – 02 lessons

Note:

14 lessons = 7 weeks (02 lessons/1 week)

1 lesson = 50 minutes

2. Ethereum Blockchain Development with Solidity

1. What is Ethereum?

2. What is ETH?

3. Smart Contracts with Solidity

4. The Remix/ Smart Contract IDE/ VS/ Hardhat

5. Your First Smart Contract

6. Testing Your Contract

7. Remix and the JavaScript VM

8. The Ganache/ Hardhat Ethereum Client

9. Testing Your Contract in Ganache

10. The Ganache Ethereum Client

11. Ethereum Transactions with MetaMask

https://ethereum.org/en/

2. Ethereum Blockchain Development with Solidity

12. Ethereum ERC20 token deployment
13. Advanced Smart Contracts
14. Building Interactive Front-Ends
15. Projects with Ethereum
16. Deploying Smart Contracts to the Ethereum Network
17. Obtaining Ether for Use in Test Networks

Introduction

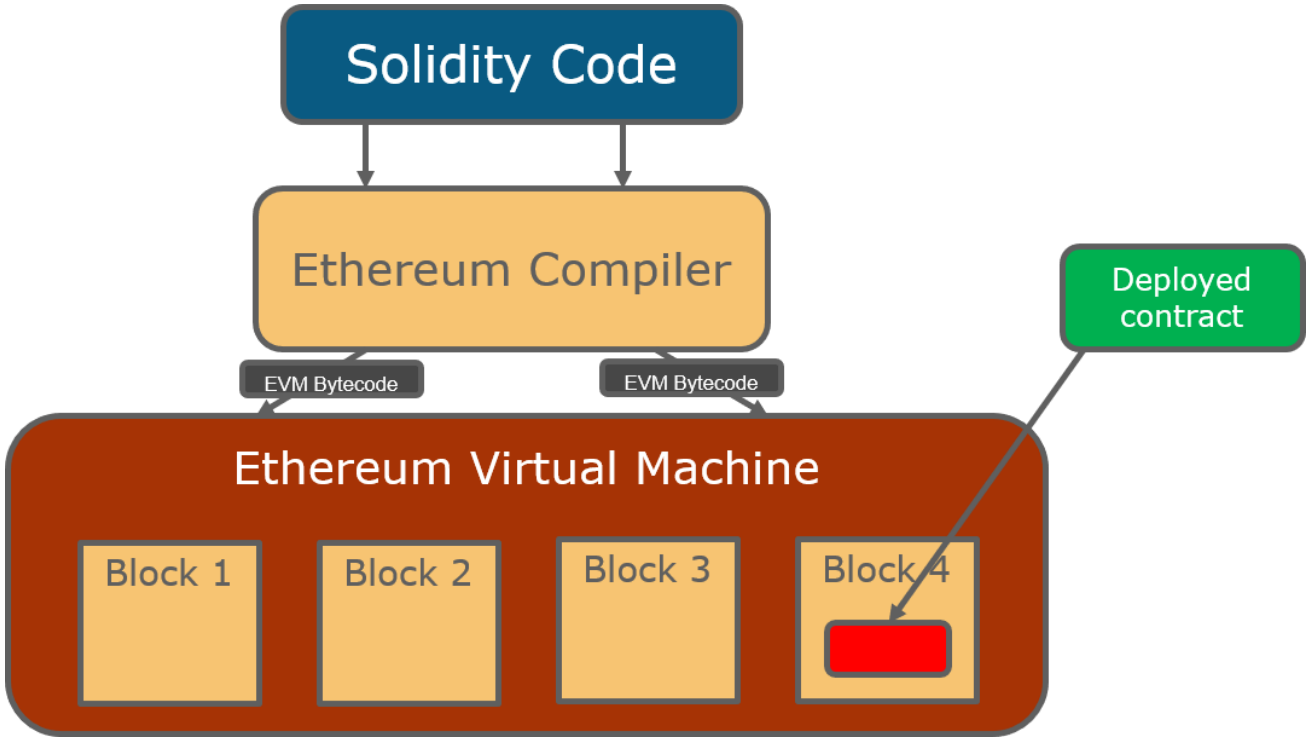
NOVA SQUARE

- Solidity is a javascript like a language used to code smart contracts on the Ethereum platform. It compiles into a bytecode format that is understood by the Ethereum Virtual machine (EVM). It's a strongly typed language with the ability to define custom data structures
- Solidity is an object-oriented, high-level language for implementing smart contracts. Smart contracts are programs which govern the behaviour of accounts within the Ethereum state.

Solidity Code Compiler

NOVA SQUARE

- **Installing the Solidity Compiler**
  - **Hardhat/ Remix:** *We recommend Hardhat/ Remix for small contracts and for quickly learning Solidity.*
  - **npm / Node.js**

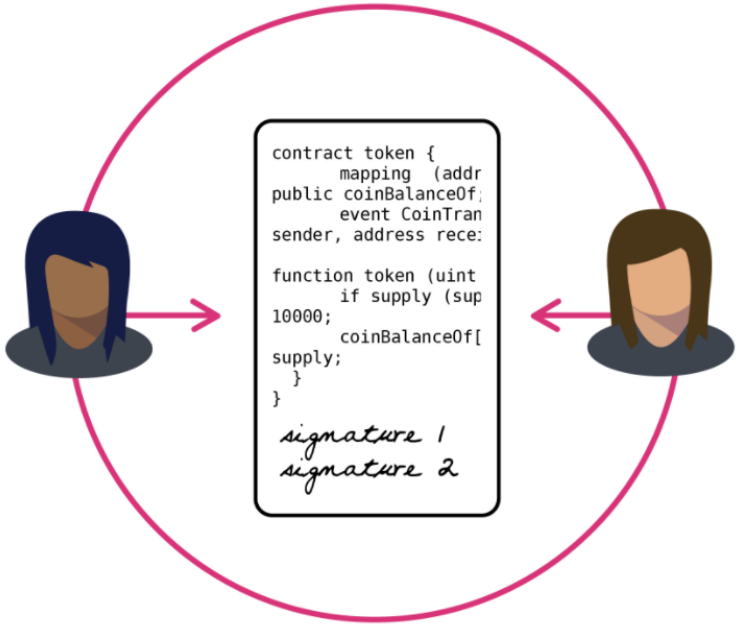


SmartContract with Solidity

NOVA SQUARE

What is SmartContract?

A **smart contract** is a computer protocol intended to digitally facilitate, verify, or enforce the negotiation or performance of a **contract**. **Smart contracts** allow the performance of credible transactions without third parties. These transactions are trackable and irreversible.



Smart contracts were first proposed in the early 1990's by computer scientist, lawyer and cryptographer Nick Szabo, who coined the term. With the present implementations, based on blockchains,"smart contract" is mostly used more specifically in the sense of general purpose computation that takes place on a blockchain or distributed ledger. In this interpretation, used for example by the Ethereum Foundation or IBM,a smart contract is not necessarily related to the classical concept of a contract, but can be any kind of computer program.

A smart contract also can be regarded as a secured stored procedure as its execution and codified effects like the transfer of some value between parties are strictly enforced and can not be manipulated, after a transaction with specific contract details is stored into a blockchain or distributed ledger. That's because the actual execution of contracts is controlled and audited by the platform, not by any arbitrary server-side programs connecting to the platform.

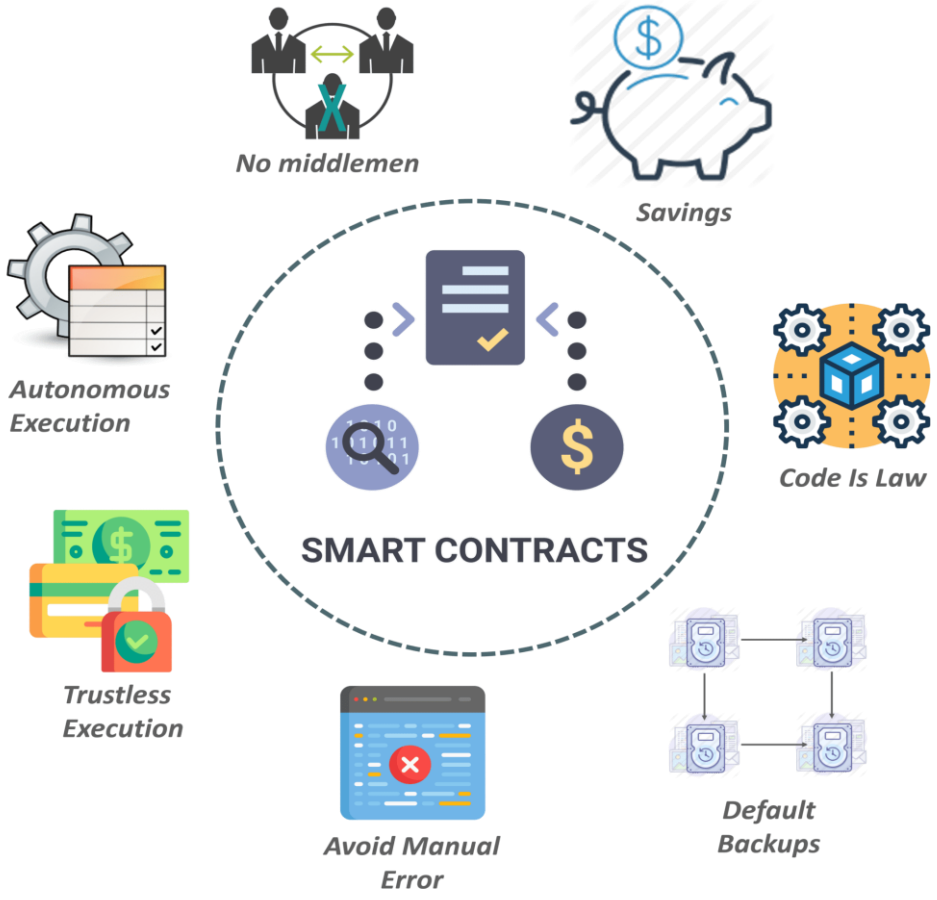
In 2018, a US Senate report said: "While smart contracts might sound new, the concept is rooted in basic contract law. Usually, the judicial system adjudicates contractual disputes and enforces terms, but it is also common to have another arbitration method, especially for international transactions. With smart contracts, a program enforces the contract built into the code."

By implementing the Decree on Development of Digital Economy, Belarus has become the first-ever country to legalize smart contracts. Belarusian lawyer Denis Aleinikov is considered to be the author of a smart contract legal concept introduced by the decree

SmartContract with Solidity

NOVA SQUARE

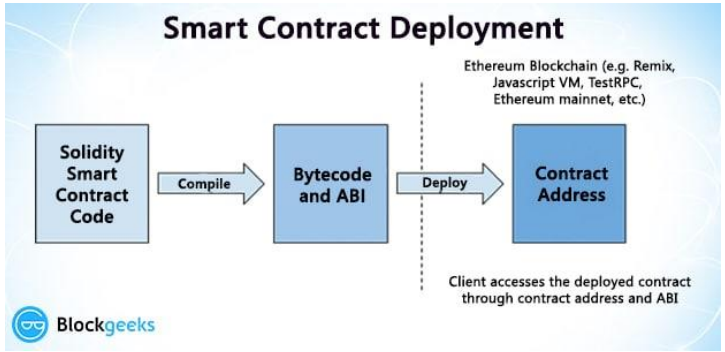
Benefit of SmartContract?

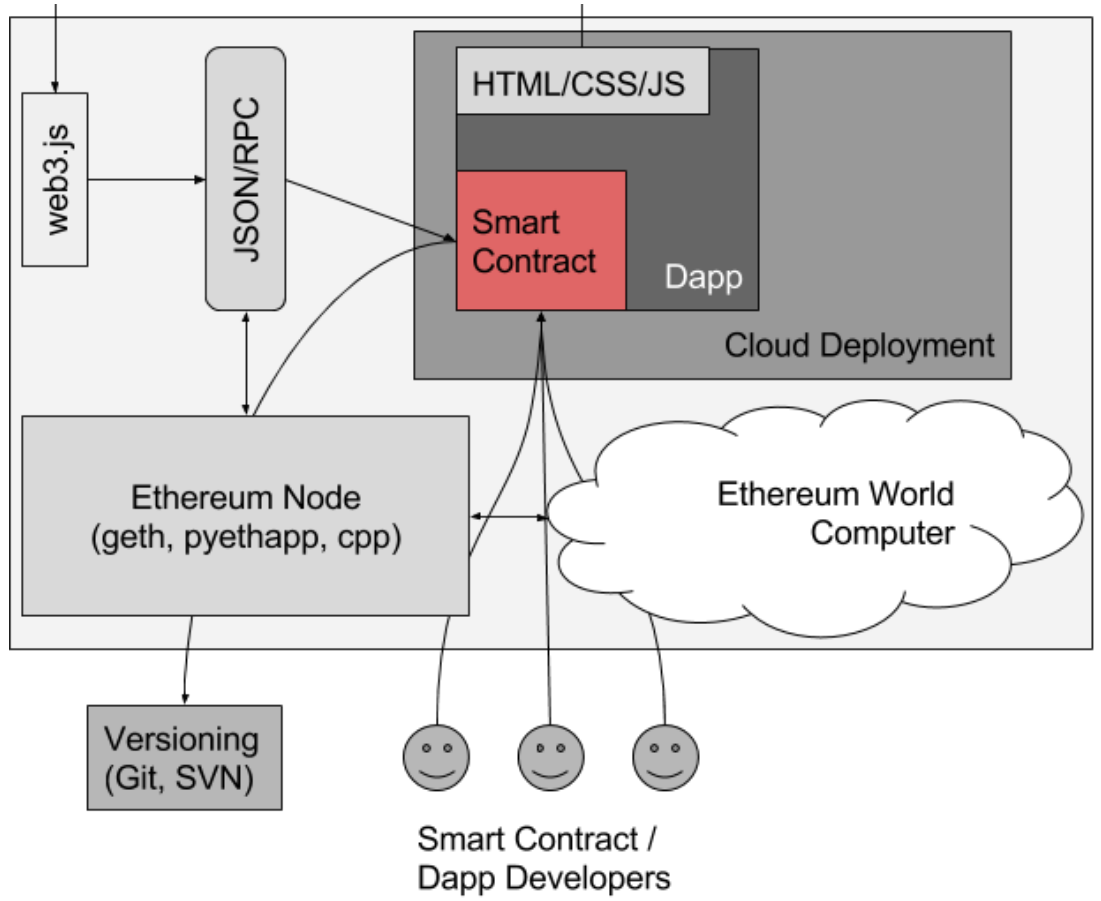


SmartContract with Solidity

NOVA SQUARE

Structure





Aspect: Solidity Env.

Aspect: JavaScript Env.

Aspect: IDE

Aspect: Versioning

Aspect: Collaboration

Aspect: Deployment

SmartContract with Solidity

NOVA SQUARE

- Example

Storage Example

```
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;
    }
}
```

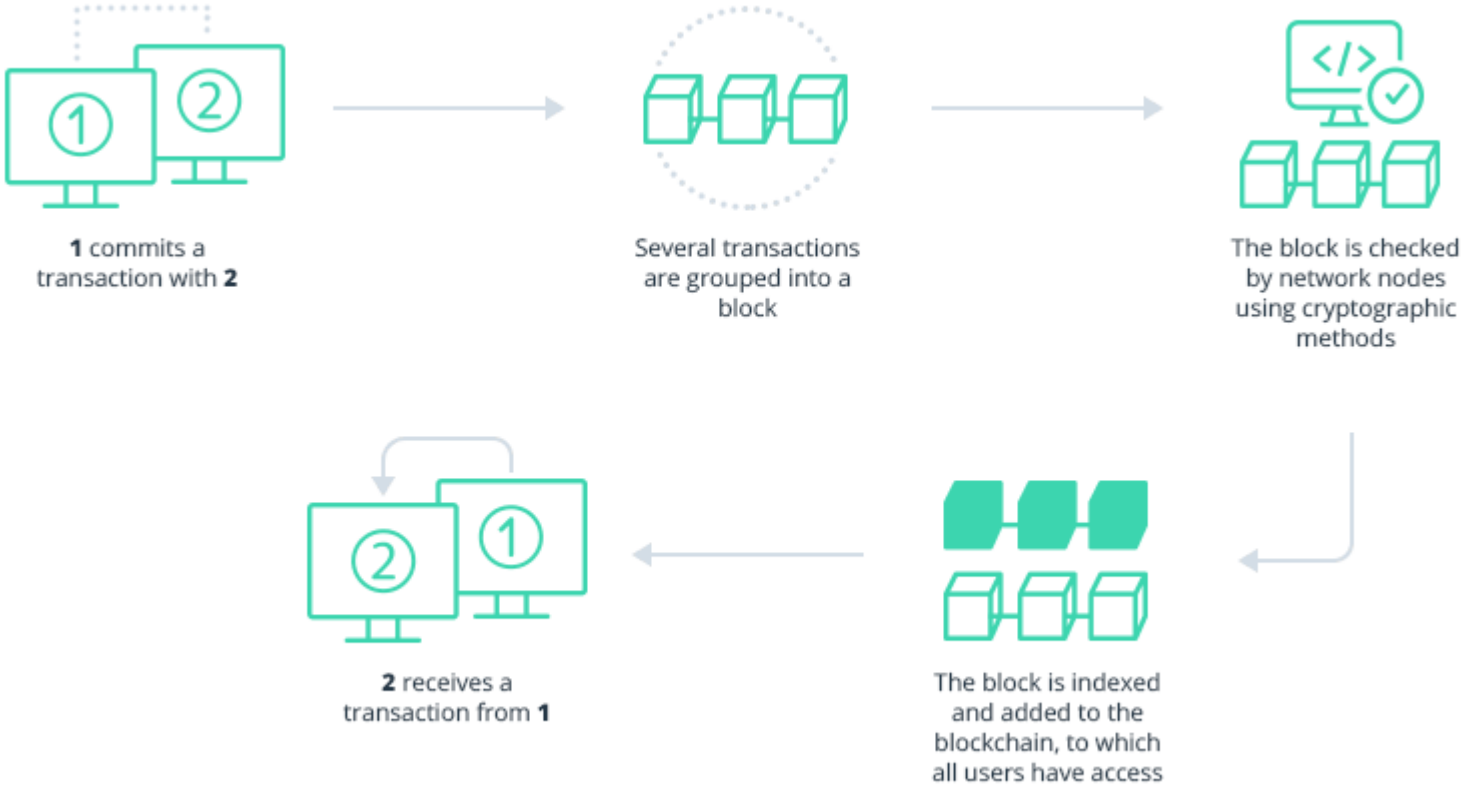
https://solidity-by-example.org/

Subcurrency Example

```
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);
    }
}
```

How smartcontract work?

NOVA SQUARE



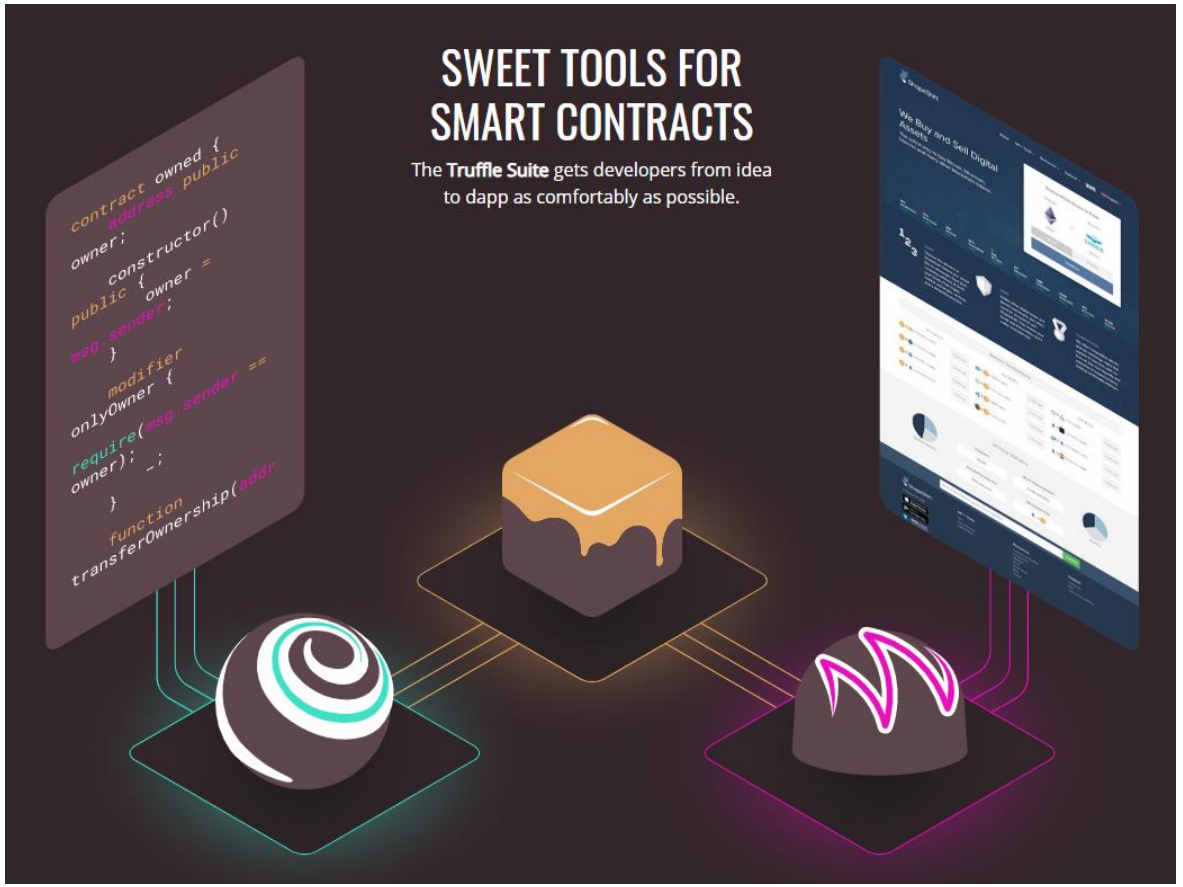
Smart Contract developments

NOVA SQUARE

- Ethereum has a large and growing number of tools to help developers build, test, and deploy their applications. Below are the most popular tools to get you started. If you want to dive deeper, check out this
- Truffle - *A development environment, testing framework, build pipeline, and other tools.*

https://www.trufflesuite.com/

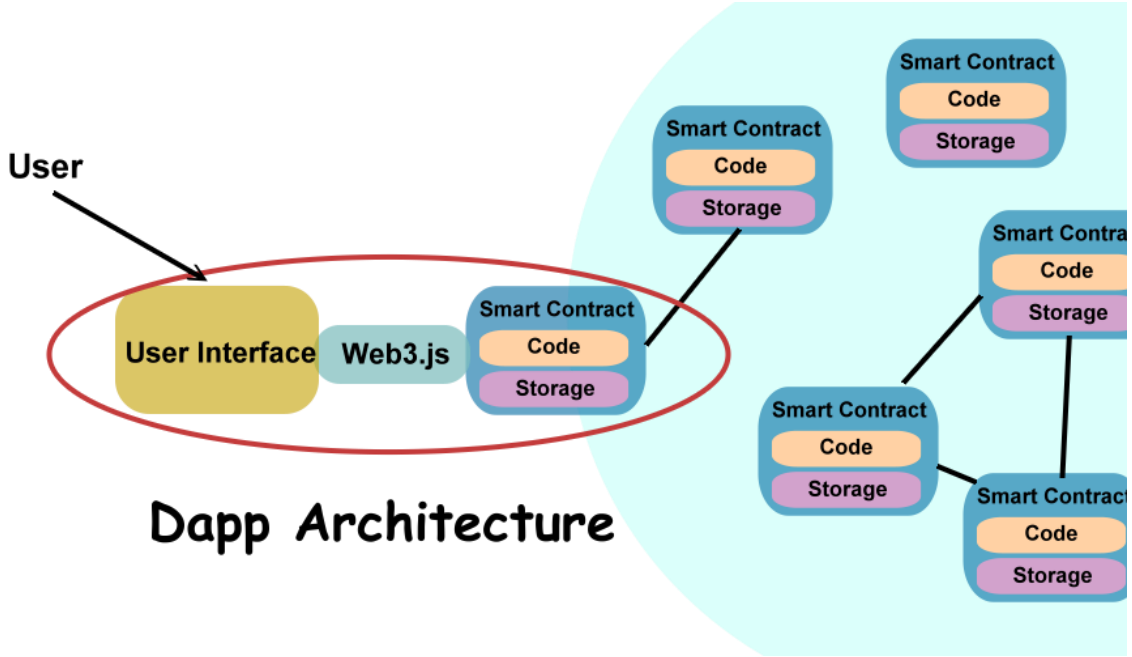
https://www.ethereum.org/developers/



Web3.js

NOVA SQUARE

- web3.js is a collection of libraries which allow you to interact with a local or remote ethereum node, using a HTTP or IPC connection.
  - The web3-eth is for the ethereum blockchain and smart contracts
  - The web3-shh is for the whisper protocol to communicate p2p and broadcast
  - The web3-bzz is for the swarm protocol, the decentralized file storage
  - The web3-utils contains useful helper functions for Dapp developers
- Adding web3.js



https://web3js.readthedocs.io/en/v1.2.2/

Web3.js

NOVA SQUARE

- API Reference example
  - Web3
    - Web3.modules
    - web3 object
  - web3.eth
    - Note on checksum addresses
    - Contract
    - personal
    - accounts
- web3.eth.Contract
  - new contract
  - defaultAccount
  - defaultBlock
  - defaultChain
  - defaultCommon
  - transactionBlockTimeout
  - transactionConfirmationBlocks
  - transactionPollingTimeout

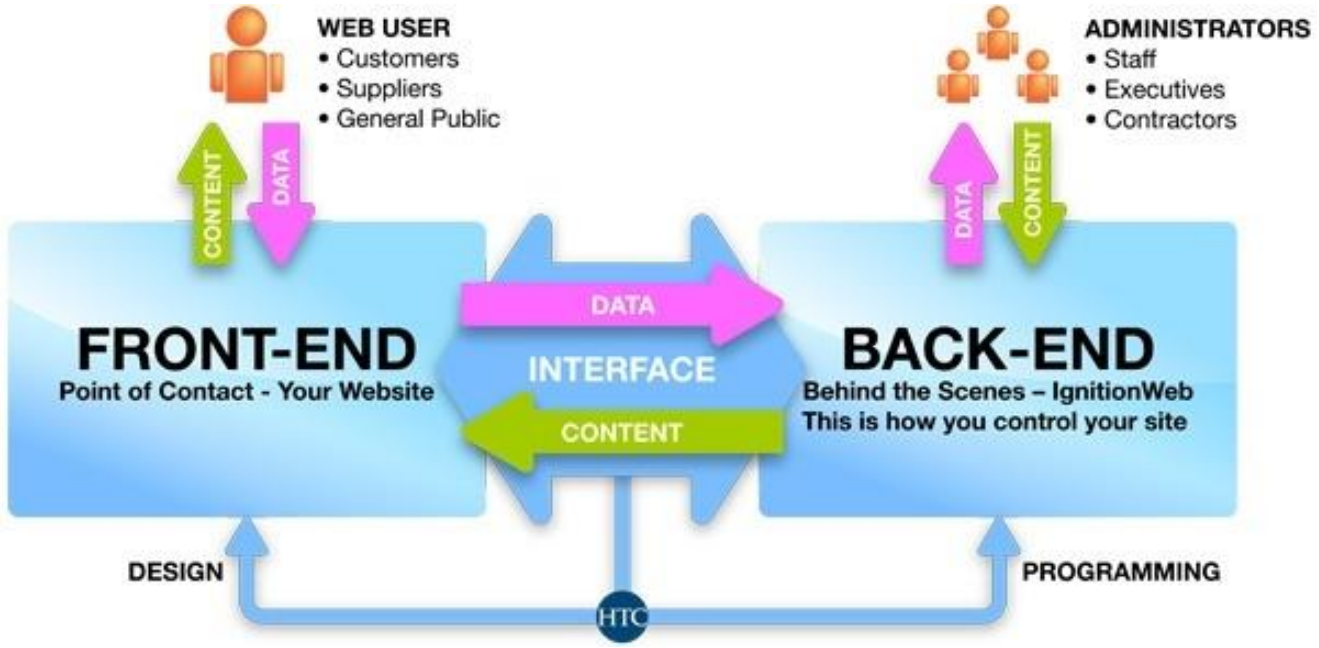
- web3.eth.accoaunts
  - create
  - privateKeyToAccount
  - signTransaction
- web3.eth.personal
  - setProvider
  - givenProvider
  - currentProvider
  - newAccount
  - sign
  - signTransaction
  - sendTransaction
  - unlockAccount
  - lockAccount
  - getAccounts

https://web3js.readthedocs.io/en/v1.2.2/

Building Interactive Front-Ends

NOVA SQUARE

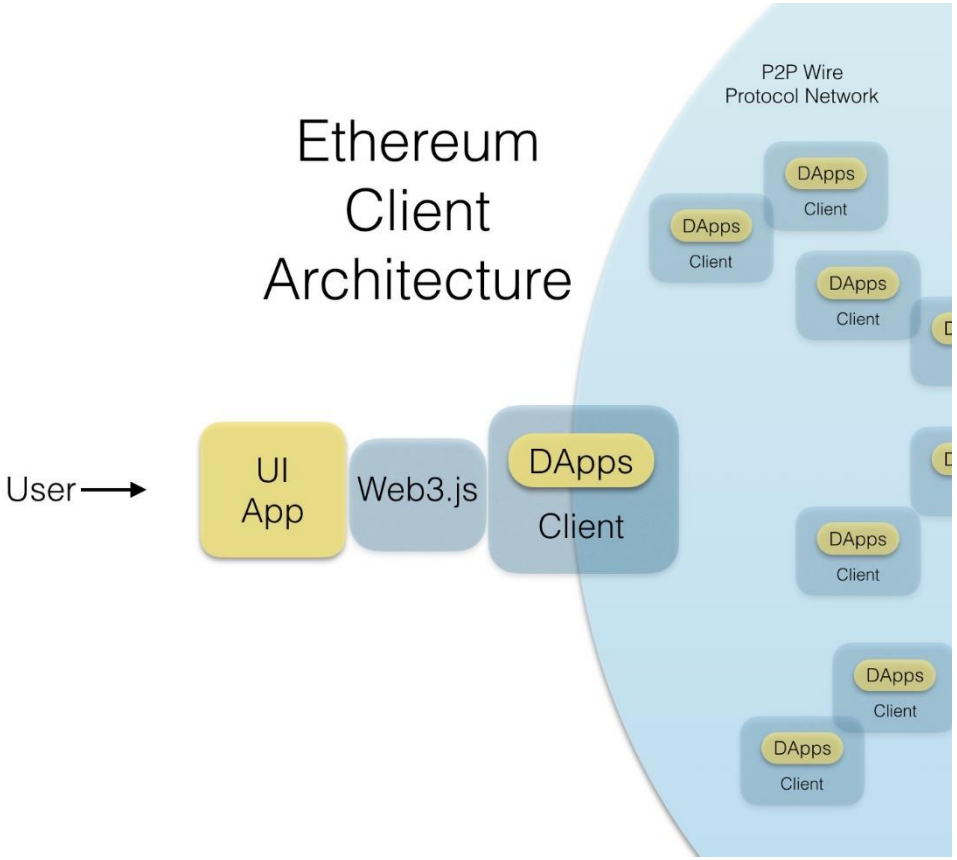
- Front-Ends
  - The front-end is everything involved with what the user sees, including design and some languages like HTML and CSS.
  - Web designer
  - User Interface (UI) Designer
  - User Experience (UX)
  - Front-End Developer
- Back-ends
  - What is back-end web development? The back-end, or the "server-side", is basically how the site works, updates, and changes. This refers to everything the user can't see in the browser, like databases and servers.



Building Interactive Front-Ends

NOVA SQUARE

- Ethereum App Architecture

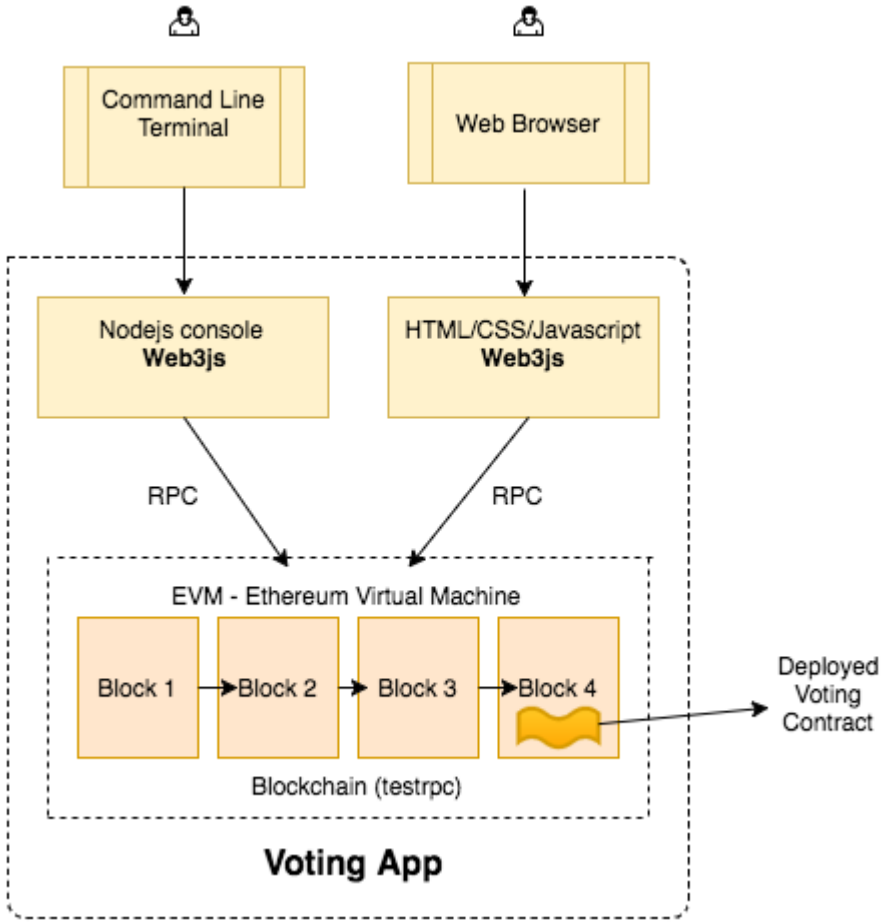


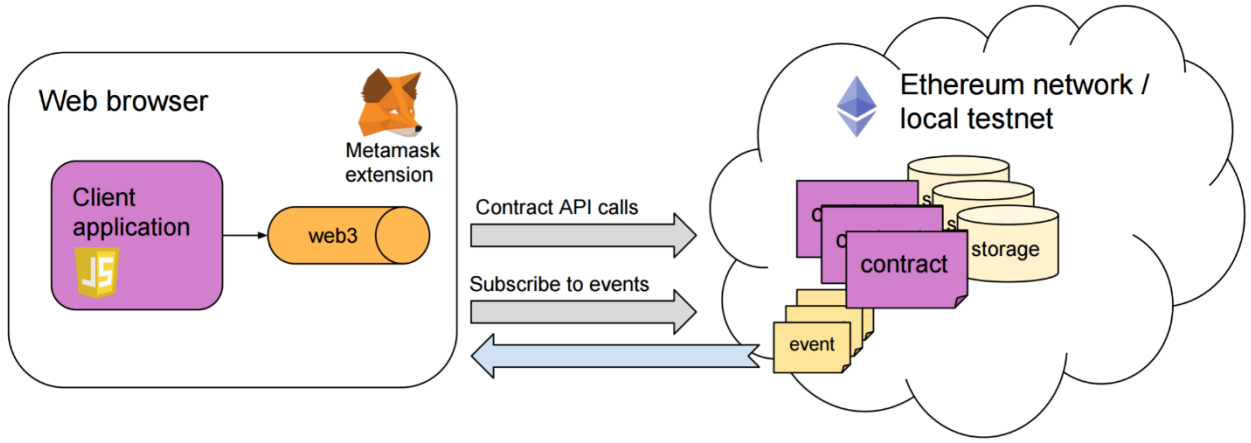

```
var Web3 = require('web3');
var web3 = new Web3();
```

Building Interactive Front-Ends

NOVA SQUARE

- Ethereum App Architecture

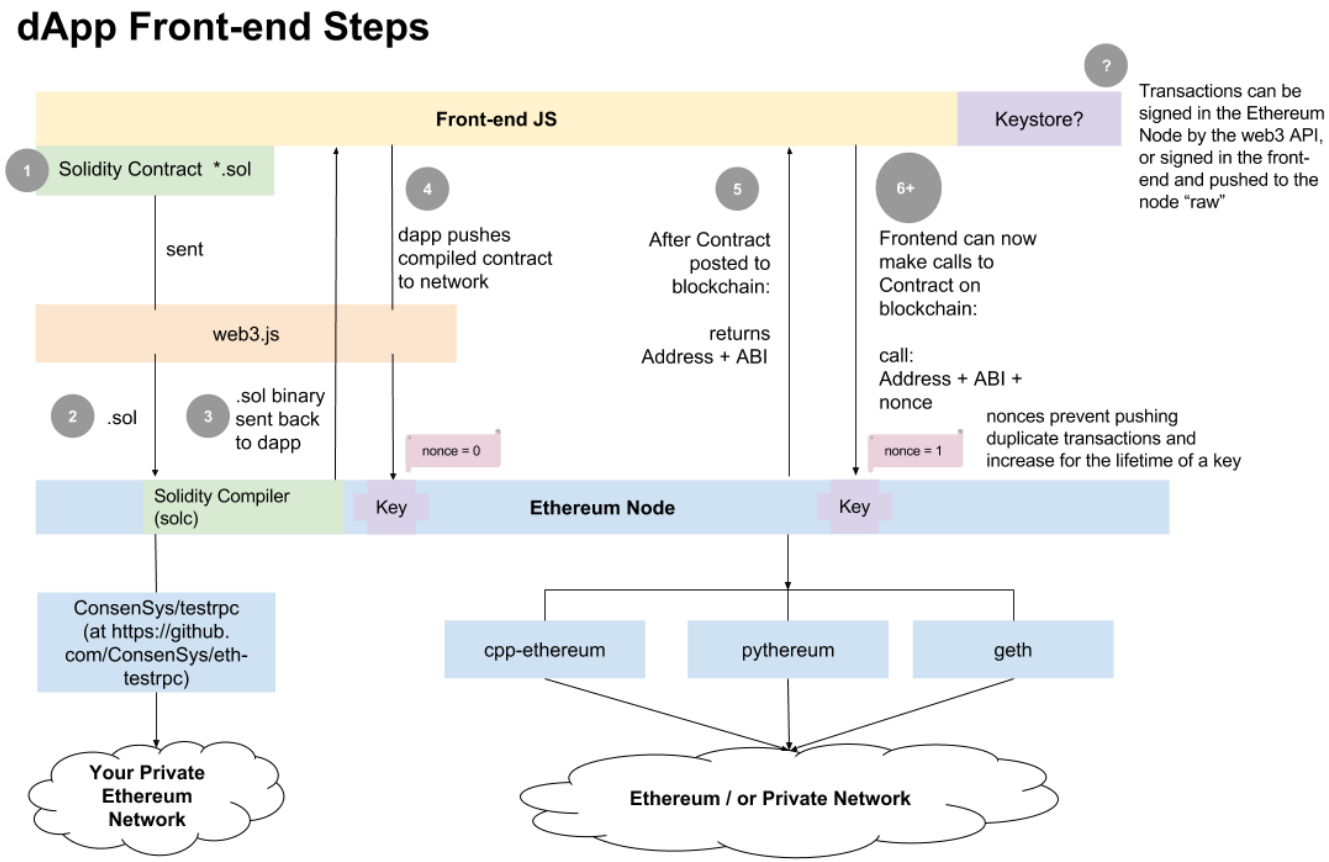




Building Interactive Front-Ends

NOVA SQUARE

- Ethereum App Architecture



A **Contract Creation Transaction** is shown in steps 1-5 at above.

An **Ether Transfer** or **Function Call Transaction** is assumed in step 6.

Ethereum Environments

NOVA SQUARE

- How to set up your Ethereum development environment for PC
- Necessary Tools
  - Powershell
    - Powershell is a very powerful tool that should be installed by default on your windows computer (at least if you're using windows 8 and above). We'll use it mainly as a terminal to accept unix commands and to install packages. As mentioned previously, most of the community uses unix based systems and command prompt won't cut it. Being able to adapt will save a lot of headache in the long run.
  - Visual Studio Code
    - Visual Studio Code is my code editor of choice when it comes to writing smart contracts. It's super lightweight, full of extensions created by the community, and has powerful debugging tools. Obviously opinions may differ when it comes to which code editor or IDE to use. The final decision is yours.
  - Geth (Go Ethreum)
    - Geth is the the command line interface for running a full Ethereum node implemented in Go. It allows you the ability to do practically anything you would need to do on the blockchain
  - Ganache
    - Ganache is a blockchain emulator that allows you to run tests, execute commands, and inspect state while controlling how the blockchain operates. Ganache was called Test RPC in the past, but the developers learned from Android that tasty dessert names are more appealing. The blockchain you create is personal, has user friendly UI, and runs quickly in memory.
  - NPM
    - NPM is a package manager for Node.js files. We'll be using this to download dependencies like Truffle.
  - Truffle
    - Truffle is an awesome tool that makes the developer's job much easier. It's provides a testing framework, smart contract compilation, linking, deployment, and much more. It also handles a lot of boilerplate for you when you get into the realm of using boxes.
- https://medium.com/interfacing-with-a-blockchain/how-to-set-up-your-ethereum-development-environment-for-pc-f6caf6eb8017

Ethereum Environments

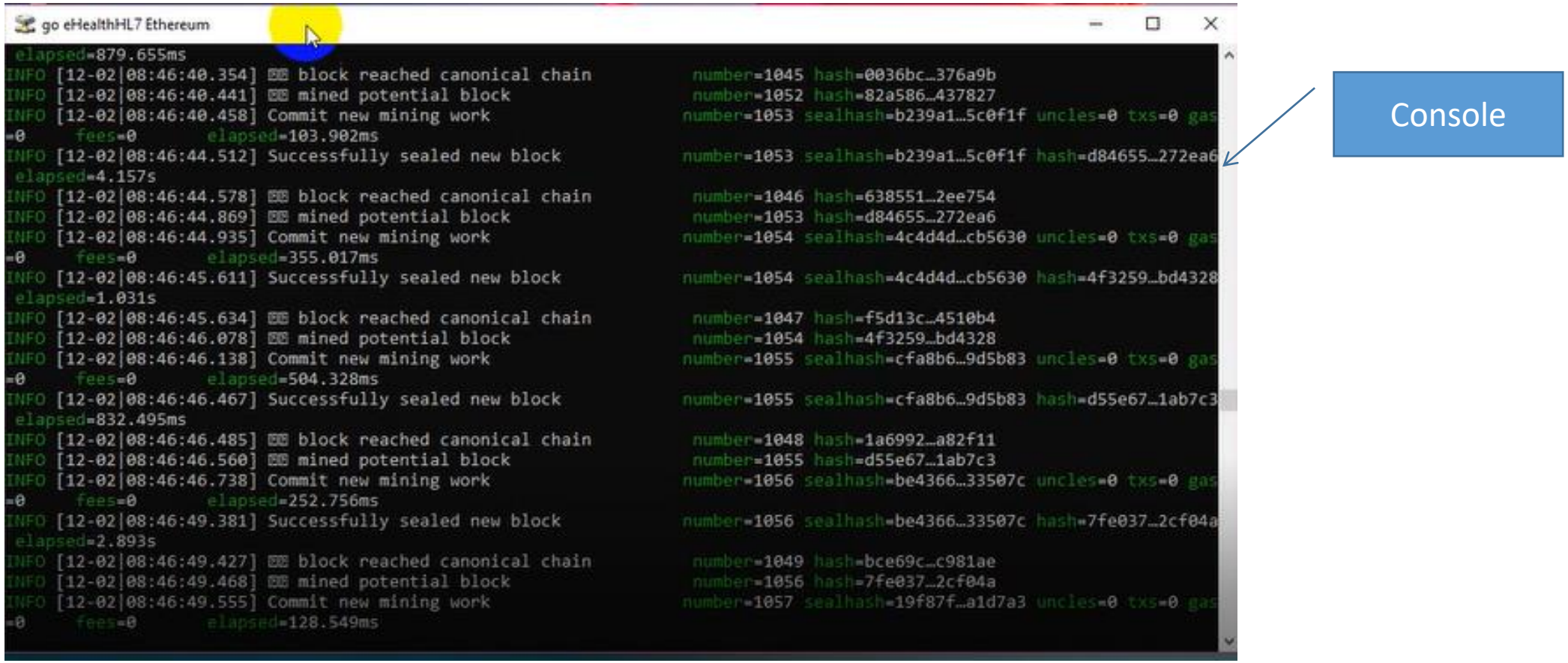
NOVA SQUARE

- Step by step
  - 1. Download eth
    - https://geth.ethereum.org/downloads/ -> download for windows/ macos,...
  - 2. Setup geth and Wallet
  - 3. Run geth
  - 4. Init Ethereum network
  - 5. View Console
  - 6. Check/ create account
    - eth.accounts
  - 7. Miner
  - 8. Deploy the contract
  - 9. Deploy the webapp

Ethereum Environments

NOVA SQUARE

- Example



Projects with Ethereum

NOVA SQUARE

- Project for Healthcare
- Project for Real Estate
- Project for Coffee
- Project for Learning Online
- Project for Seek an Origin

Projects discussion

NOVA SQUARE

- How to get an idea?
- How to wireframe?
- Show prototype?
- Setup an Ethereum Env
- Deploy Webapp
- How to transfers money, data and demo

THANK YOU

Email: tuan.truongquoc@gmail.com
tuan@novasquare.vn
Mobile: 0913-111-576