Blockstack

Innovating With Bitcoin

Joe Bender - Developer Evangelist

ExpoLab
Creativity Unfolded

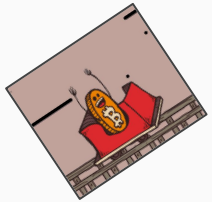
October 2020

Introduction

I’m over four months into my **Developer Evangelist** journey at **Blockstack**. My focus is empowering our community to build robust tools & useful applications.

Favorite Quarantine Activity?

Browsing Bitcoin memes!



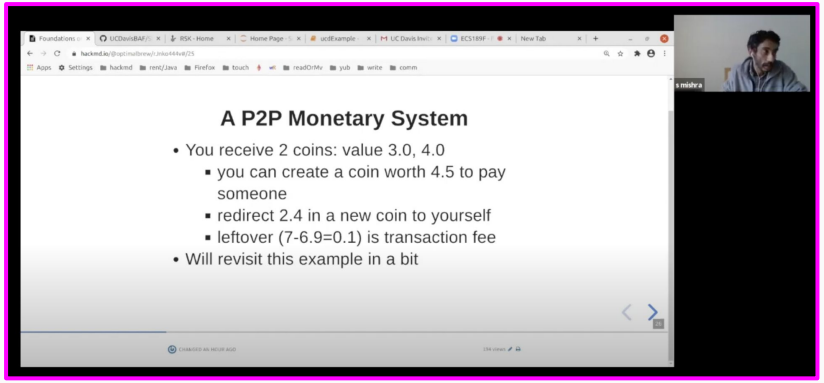
Overview

During this talk, I'll explain:

1. Blockchain Basics
2. Bitcoin & Lightning
3. Blockstack
4. The Stacks 2.0 Testnet
5. Clarity Smart Contracts
6. Overview of Running a Node & Miner
7. Why Run a Node?
8. How to Get Involved

BLOCKSTACK

Catch-Up



Principle Foundations of Bitcoin
Shreemoy Mishra

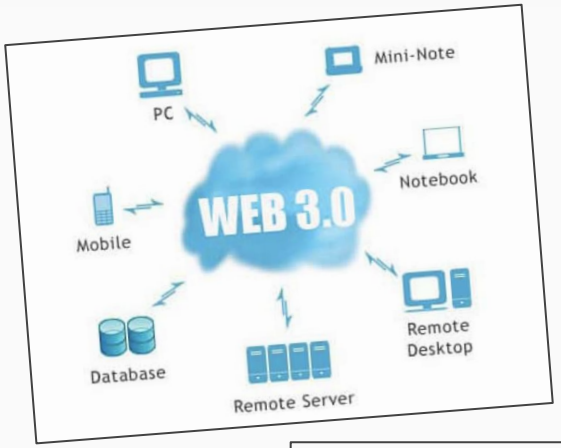
How Ethereum Works
John Long

“No company on the internet should have so much power that they get to debate if they should be evil today or not..”

- Muneeb Ali, Co-Founder of Blockstack

BLOCKCHAIN BASICS

What the heck is a Web 3.0?



Early Internet Decentralization

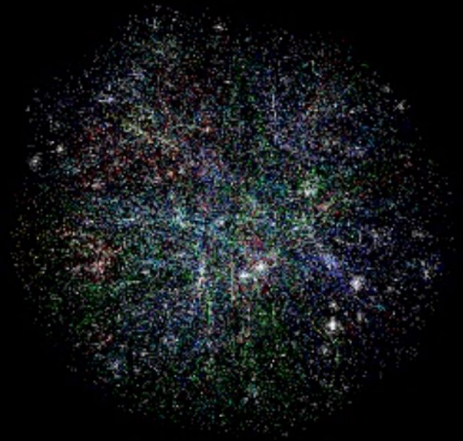
Lime wire

KaZaA

napster

BitTorrent™

Tor



Internet

wires, network

Web 1.0

read-only
static

Web 2.0

read-write
dynamic

Web 3.0

read-write-trust
verifiable

The best place to get started with crypto

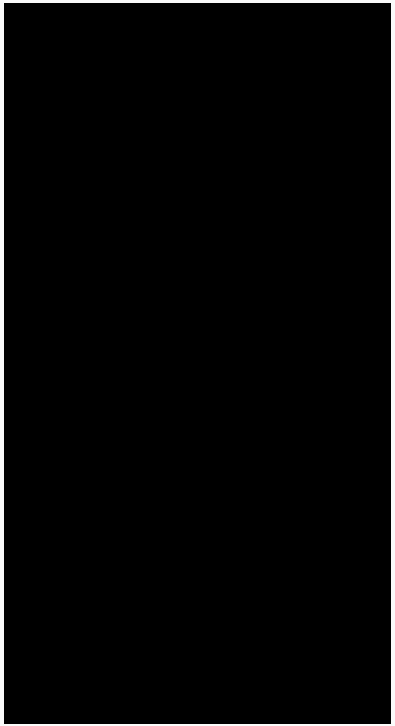
coinbase

METAMASK

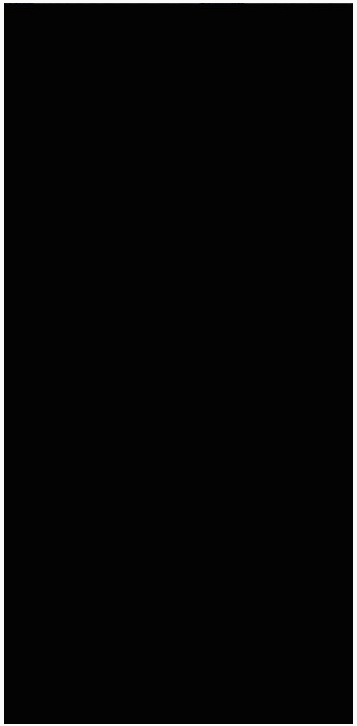
A crypto wallet & gateway to blockchain apps

First-Movers

coinbase



METAMASK



First-Movers

coinbase

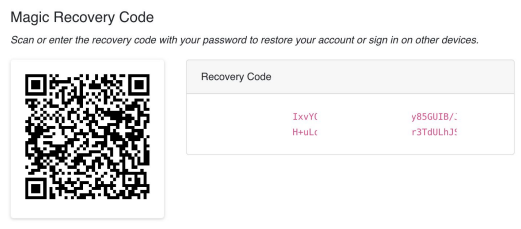
- 30+ Million Users
- Password Login, Coinbase has Private Key
- Buy Crypto directly within app
- Implemented Staking Rewards (Tezos)
- Coinbase Pro & Coinbase Wallet
- Verification Levels
  - Level 1 - Phone #
  - Level 2 - Personal Info
  - Level 3 - Verify Photo ID

METAMASK

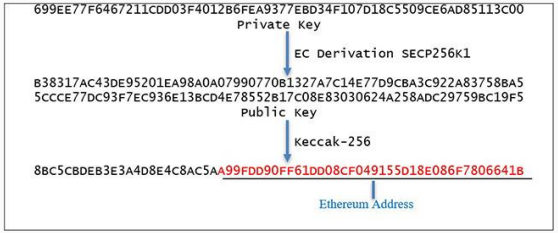
- 1+ Million Extension Downloads
- Password Login, User stores seed phrase
- Recommends Wyre or CoinSwitch to purchase crypto
- Allows user to connect to testnets or local networks
- Mobile app added dApp browser

User Experience

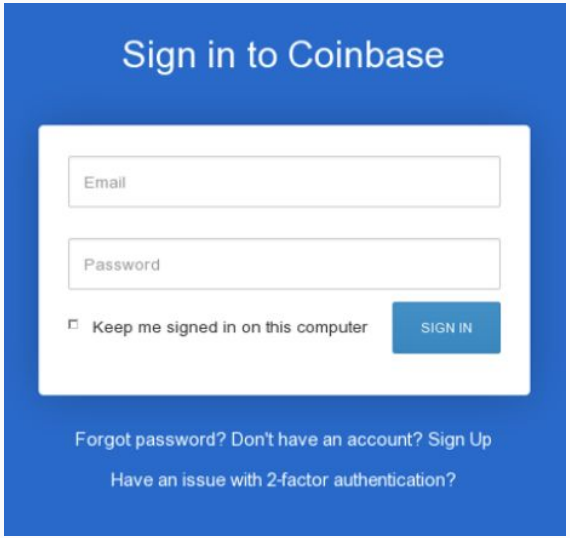
Secret Key



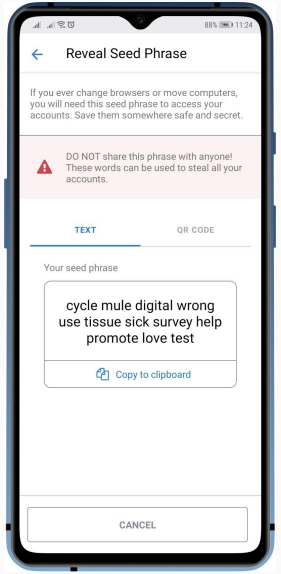
Private Key



Password



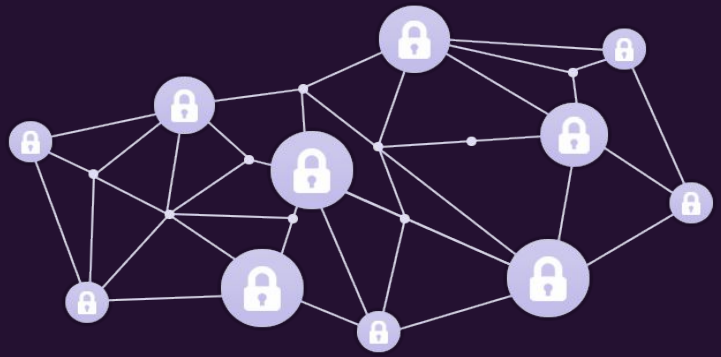
Seed Phrase



Networks & Nodes

- In telecommunication networks, **nodes** act as redistribution points or as a communication endpoints.
- Full node, master node, light node, miner, super node.
- Download particular software to your machine that outlines rules for connecting to **network**.

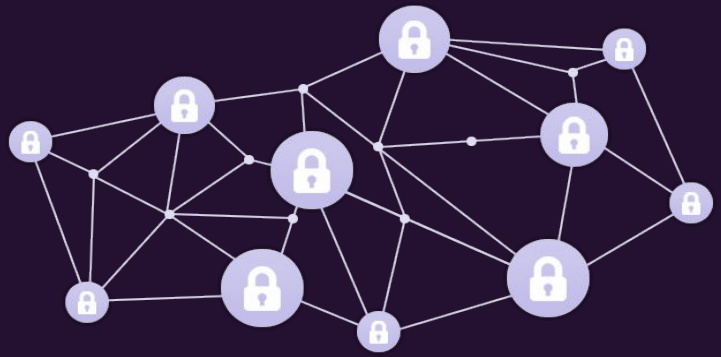
- Node = Building blocks of blockchain.
- Blockchain = blocks of data.
  - Data is stored in nodes.
- All **nodes** on a **blockchain** are connected to each other and they constantly exchange the latest blockchain data with each other so all nodes stay up to date.



Networks & Nodes

- In telecommunication networks, **nodes** act as redistribution points or as a communication endpoints.
- Full node, master node, light node, miner, super node.
- Download particular software to your machine that outlines rules for connecting to **network**.

- Node = Building blocks of blockchain.
- Blockchain = blocks of data.
  - Data is stored in nodes.
- All **nodes** on a **blockchain** are connected to each other and they constantly exchange the latest blockchain data with each other so all nodes stay up to date.



Enter, Bitcoin

Bitcoin Genesis Block

Raw Hex Version

```
00000000   01 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00   ................
00000010   00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00   ................
00000020   00 00 00 00 3B A3 ED FD   7A 7B 12 B2 7A C7 2C 3E   ....;£íýz{.²zÇ,>
00000030   67 76 8F 61 7F C8 1B C3   88 8A 51 32 3A 9F B8 AA   gv.a.È.Ã^ŠQ2:Ÿ¸ª
00000040   4B 1B 5E 4A 29 AB 5F 49   FF FF 00 1D 1D AC 2B 7C   K.^J)«_Iÿÿ...¬+|
00000050   01 01 00 00 00 01 00 00   00 00 00 00 00 00 00 00   ................
00000060   00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00   ................
00000070   00 00 00 00 00 00 FF FF   FF FF 4D 04 FF FF 00 1D   ......ÿÿÿÿM.ÿÿ..
00000080   01 04 45 54 68 65 20 54   69 6D 65 73 20 30 33 2F   ..EThe Times 03/
00000090   4A 61 6E 2F 32 30 30 39   20 43 68 61 6E 63 65 6C   Jan/2009 Chancel
000000A0   6C 6F 72 20 6F 6E 20 62   72 69 6E 6B 20 6F 66 20   lor on brink of
000000B0   73 65 63 6F 6E 64 20 62   61 69 6C 6F 75 74 20 66   second bailout f
000000C0   6F 72 20 62 61 6E 6B 73   FF FF FF FF 01 00 F2 05   or banksÿÿÿÿ..ò.
000000D0   2A 01 00 00 00 43 41 04   67 8A FD B0 FE 55 48 27   *....CA.gŠý°þUH'
000000E0   19 67 F1 A6 71 30 B7 10   5C D6 A8 28 E0 39 09 A6   .gñ¦q0·.\Ö¨(à9.¦
000000F0   79 62 E0 EA 1F 61 DE B6   49 F6 BC 3F 4C EF 38 C4   ybàê.aÞ¶Iö¼?Lï8Ä
00000100   F3 55 04 E5 1E C1 12 DE   5C 38 4D F7 BA 0B 8D 57   óU.å.Á.Þ\8M÷º..W
00000110   8A 4C 70 2B 6B F1 1D 5F   AC 00 00 00 00            ŠLp+kñ._¬....
```

- Launched 03 January 2009 by “Satoshi Nakamoto”

THE TIMES

Max 5C, min -5C

Saturday January 3 2009 timesonline.co.uk No 69523

£1.50

Eat Out from £5

More than 900 great restaurants, including four **Gordon Ramsay** favourites from £15

Start collecting tokens today Pullout inside

Chancellor on brink of second bailout for banks

Billions may be needed as lending squeeze tightens

Francis Elliott Deputy Political Editor
Gary Duncan Economics Editor

Alistair Darling has been forced to consider a second bailout for banks as the lending drought worsens.

The Chancellor will decide within weeks whether to pump billions more into the economy as evidence mounts that the £37 billion part-nationalisation last year has failed to keep credit flowing. Options include cash injections, offering banks cheaper state guarantees to raise money privately or buying up "toxic assets", The Times has learnt.

The Bank of England revealed yesterday that, despite intense pressure, the banks curbed lending in the final quarter of last year and plan even tighter restrictions in the coming months. Its findings will alarm the Treasury.

The Bank is expected to take yet more aggressive action this week by cutting the base rate from its current level of 2 per cent. Doing so would reduce the cost of borrowing but have little effect on the availability of loans.

Whitehall sources said that ministers planned to "keep the banks on the hoof" but accepted that they need more help to restore lending levels. Formally, the Treasury plans to focus on state-backed guarantees to encourage private finance, but a number of interventions are on the table, including further injections of taxpayers' cash.

Under one option, a "bad bank" would be created to dispose of bad debts. The Treasury would take bad loans off the hands of troubled banks, perhaps swapping them for government bonds. The toxic assets, blamed for poisoning the financial system, would be parked in a state vehicle or "bad bank" that would manage them and attempt to dispose of them while "detoxifying" the mainstream banking system.

The idea would mirror the initial proposal by Henry Paulson, the US Treasury Secretary, to underpin the American banking system by buying

Continued on page 6, col 1
Leading article, page 2

99p

Pub chain cuts the price of a pint from £1.69 to 1968 levels

Business, page 47

Salman Rushdie I won't marry again

Pages 22, 23

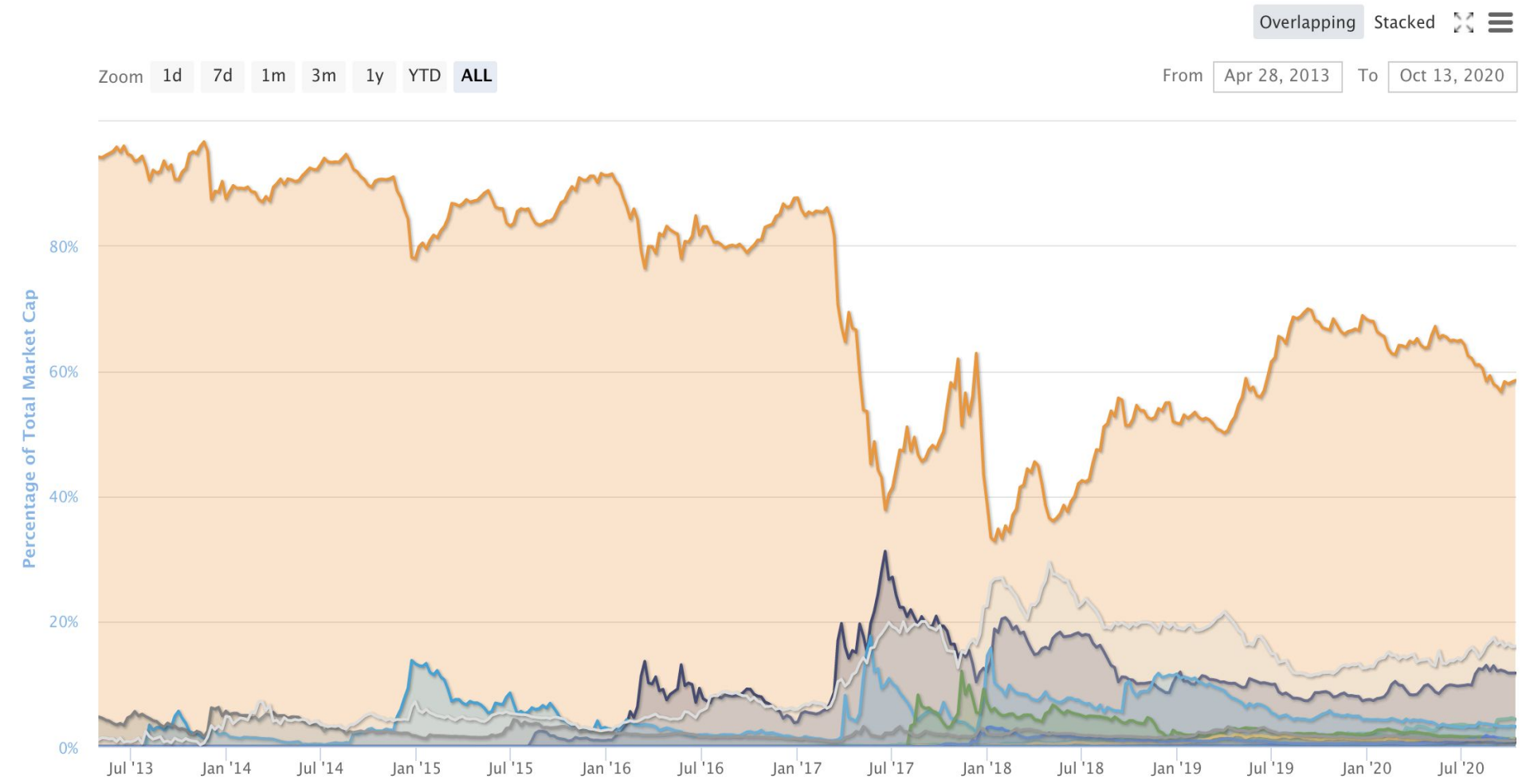
Giant killing? Guide to the FA Cup third round

Sport

Bitcoin

- Launched 03 January 2009 by “Satoshi Nakamoto”
- First widespread implementation of decentralized cryptocurrency
- Current Market Cap > $200 Billion USD
-

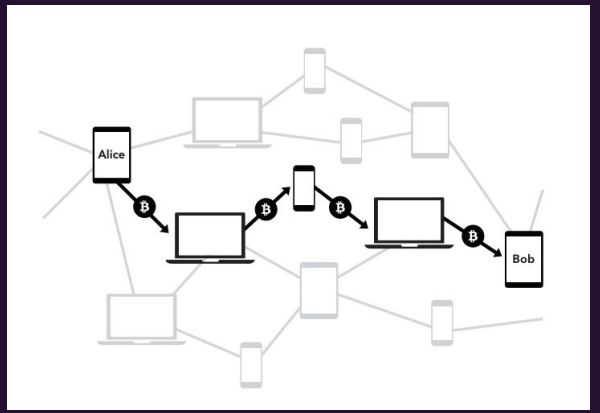
Percentage of Total Market Capitalization (Dominance)



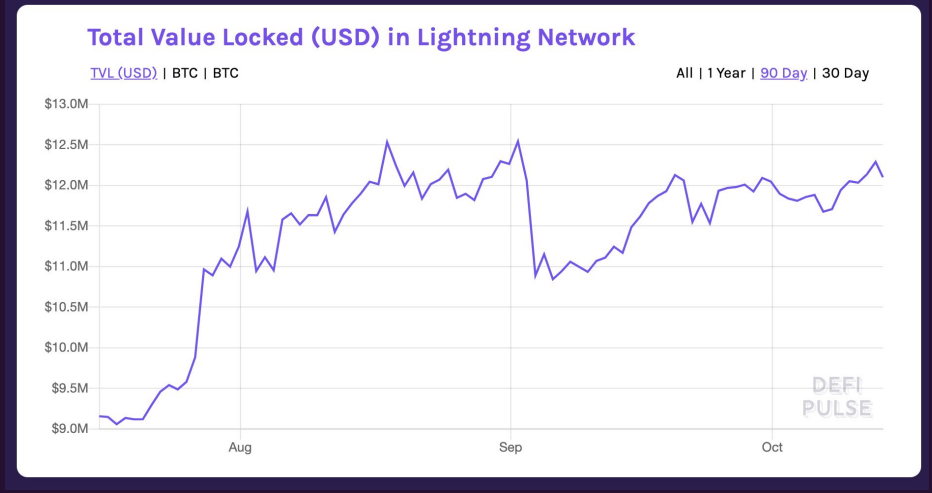
Lightning Network

Lightning Network

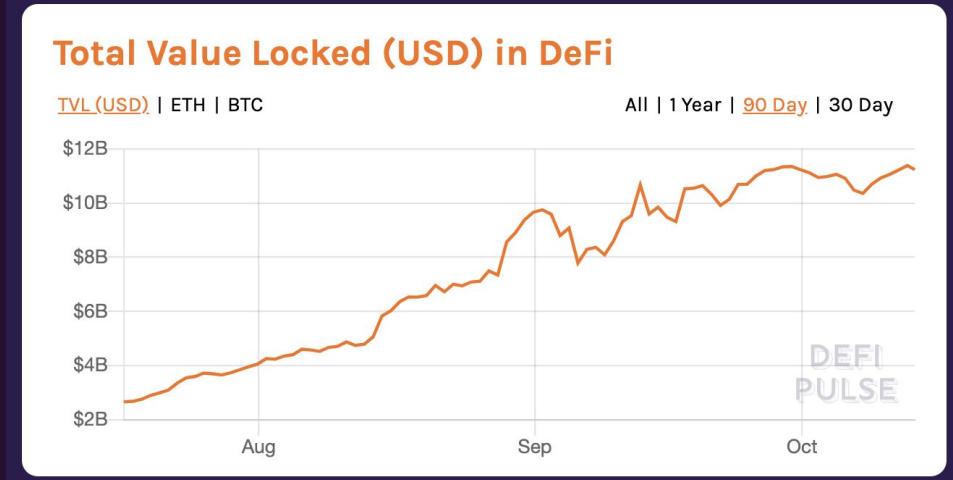
- https://lightning.network/
- Scalability solution.
- Whitepaper written by Joseph Poon and Thaddeus Dryja in 2015.
- Transactions are sent over a network of micropayment channels.
- Moves small and frequent transactions off-chain, allowing for fast peer-to-peer transactions and low fees.
- Uses native smart-contract scripting language.
- Creates a two-party ledger entry.
- Bi-directional.
- Makes cross-chain atomic swaps possible so long as the chains can support the same cryptographic hash function.
- Channel closes upon completion.



Lightning Network vs. DeFi



12.1 Million USD



11.23 Billion USD

At **Blockstack**, we're building the tools needed for a **user-owned internet**.

BLOCKSTACK

Software for a user owned internet

Blockstack is an **open-source** effort to develop software that provides an alternative to traditional (**centralized**) web applications.

We've developed a full-stack **decentralized computing network** that enables a new generation of applications where developers and users can interact **fairly** and **securely**.

Blockstack uses **blockchain** technology to build **protocols** and developer **tools** designed to enable a fair and open Internet that returns **digital rights** to **developers** and **consumers**.

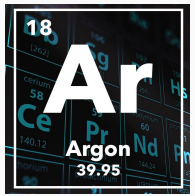
Stacks 2.0

The Stacks 2.0
Testnet is Live

At Blockstack, we believe Web 3 will introduce true internet ownership anchored to the most secure blockchain: **Bitcoin**.

Stacks 2.0 represents the design by which Web 3 can emerge and scale.

Designed with security, scalability, and speed in mind.



Proof of Transfer (PoX): A novel mining mechanism that leverages Bitcoin to secure a new network. PoX enables benefits not possible with just proof-of-work or proof-of-stake.

Stacking: A novel value transfer mechanism that we have proposed that allows Stacks holders to earn Bitcoin for actively participating in the consensus algorithm.

Clarity: A smart contract language that optimizes for predictability and security. Enables developers to write expressive smart contracts and experiment with new business models.

Smart Contracts

- Smart contracts **encode** and **enforce** rules for modifying a particular set of data that is shared among people and entities who don't necessarily trust each other.
- Exist in a **blockchain**, anyone can query them, and anyone can submit transactions to execute them.
- A smart contract execution can result in new **transactions** being written to the blockchain.
- Apps can take advantage of smart contracts to manage a **global state** that is visible to the public.
- Anyone can **audit** the blockchain in order to independently verify that an app's global shared state has been managed correctly according to the smart contracts' rules.

- **Use Cases**
  - Access control (e.g. pay to access)
  - Non-fungible (e.g. collectibles) and fungible tokens (e.g. stablecoins)
  - Business model templates (e.g. subscriptions)
  - App-specific blockchains
  - Decentralized Autonomous Organizations

Clarity Programming Language

- Clarity is a programming language for writing smart contracts on the Stacks 2.0 blockchain.
- Differs from other SC languages:
  - **Interpreted:** Human-readable and auditable
  - **Decidable:** Determine precisely what code is being executed, for any function.
- Ability to write fully expressive smart contracts that anchor to Bitcoin.
- The Clarity language uses a strong static type system. LISP-based.

- A Clarity smart contract is composed of two parts — a data space and a set of functions.
  - Only the associated smart contract may modify its corresponding data space on the blockchain.
- Users call smart contracts' public functions by broadcasting a transaction on the blockchain which invokes the public function.
- Function arguments and database schemas require specified types, and use of types is checked during contract launch.

Clarity Programming Language

These are impossible in Clarity

- Breaking news: 0x down. # assembly # compilation
- DAO Hack. $50M # re-entrancy # dynamic-dispatch
- Multi-Sig hack(s). $200M # re-entrancy # dynamic-dispatch
- Hundreds of txs aborting: ___$ / day. # compilation # out-of-gas
- Audits are hard: [$__k: $___k] / contract. # undecidibility

Basic Example - Counter

counter.clar

contracts > counter.clar

```
(define-data-var counter int 0)

(define-public (get-counter)
  (ok (var-get counter)))

(define-public (increment)
  (begin
    (var-set counter (+ (var-get counter) 1))
    (ok (var-get counter))))

(define-public (decrement)
  (begin
    (var-set counter (- (var-get counter) 1))
    (ok (var-get counter))))
```

1. `define-data-var` : initializes a new integer variable `counter` with the value set to 0.
   - The counter variable is stored in the data space associated with this particular smart contract.
2. `define-public` : provides access to the `counter` variable from outside of the current smart contract.
   - The `var-get` statement looks for a variable in the contract's data space and returns it.
3. `begin` statement evaluates the multi-line expressions and returns the value of the last expression. In this case, it is used to set a new value and return the new value.

Testing - Counter

```
counter contract test suite
    ✓ should have a valid syntax (39ms)
    deploying an instance of the contract
    ✓ should start at zero
    ✓ should increment (133ms)
    ✓ should decrement (177ms)


4 passing (586ms)
```

1. Should be successfully deployed with valid syntax
2. Start at 0
3. Able to run increment method and add 1 to variable
4. Able to run decrement method and subtract 1 from variable

Testing - Counter

```
describe("counter contract test suite", () => {
  let counterClient: Client;
  let provider: Provider;
  before(async () => {
    provider = await ProviderRegistry.createProvider();
    counterClient = new Client("SP3GWX3NE58KXHESRYE4DYQ1S31PQJTCRXB3PE9SB.counter", "counter", provider);
  });
  it("should have a valid syntax", async () => {
    await counterClient.checkContract();
  });
  describe("deploying an instance of the contract", () => {
    const getCounter = async () => {
      const query = counterClient.createQuery({
        method: { name: "get-counter", args: [] }
      });
      const receipt = await counterClient.submitQuery(query);
      const result = Result.unwrapInt(receipt);
      return result;
    }
    const execMethod = async (method: string) => {
      const tx = counterClient.createTransaction({
        method: {
          name: method,
          args: [],
        },
      });
      await tx.sign("SP2J6ZY48GV1EZ5V2V5RB9MP66SW86PYKKNRV9EJ7");
      const receipt = await counterClient.submitTransaction(tx);
      return receipt;
    }
    before(async () => {
      await counterClient.deployContract();
    });
```

Notice how the instance of the smart contract is created on line 8

- Where to find `counter.clar`

Creates a transaction query that tests the valid syntax.

Passes first test!

Testing - Counter

```
it("should start at zero", async () => {
  const counter = await getCounter();
  assert.equal(counter, 0);
})
it("should increment", async () => {
  await execMethod("increment");
  assert.equal(await getCounter(), 1);
  await execMethod("increment");
  assert.equal(await getCounter(), 2);
})
it("should decrement", async () => {
  await execMethod("decrement");
  assert.equal(await getCounter(), 1);
  await execMethod("decrement");
  assert.equal(await getCounter(), 0);
})
```

1. Gets the counter variable, checks it is equal to zero.
2. Checks if variable can be incremented by calling `increment` then `getCounter` twice.
3. Checks if variable can be decremented by calling `decrement` then `getCounter` twice.

Tip Calculator

```
1
2   (define-data-var meal-cost int 0)
3   (define-data-var tip int 0)
4   (define-data-var rating int 0)
5
6   (define-public (reserve-meal-cost (cost int))
7     (ok
8       (begin
9         (var-set meal-cost cost)
10        (calculate-tip)
11        (var-get meal-cost))))
12
13  (define-public (get-meal-cost)
14    (ok (var-get meal-cost)))
15
16  (define-public (get-tip-value)
17    (ok (var-get tip)))
18  (define-public (get-rating)
19    (ok (var-get rating)))
20
```

```
21  (define-public (finish-meal (mealRating int))
22    (ok
23      (begin
24        (var-set rating mealRating)
25        (calculate-tip)
26        (var-get rating))))
27
28  ;; if rating is greater than 3 then the user is very satisfied
29  ;; if rating is less than or equals 3 then the user is dissatisfied
30  ;; tip would be the minimum 15%
31  ;; tip would be 20%
32  ;; Support these workers they are literally putting their lives on the line
33  ;; if you'd like to support organizations helping in the corona effort
34  ;; help people get food through https://www.cityharvest.org/
35  ;; I hope this doesn't disqualify me xD
36  (define-private (calculate-tip)
37    (begin
38      (if
39        (> (var-get rating) 3)
40        (var-set tip (/ (* (var-get meal-cost) 20) 100))
41        (var-set tip (/ (* (var-get meal-cost) 15) 100))
42        )))
```

Types of Smart Contracts

- Simple exchange
- DAOs
- Dapp
- Counter
- Supply chain
- Deeds
- Access Restriction
- Withdrawals
- State Machines
- Balance Address Checker

- Contracts call another Contract
- Fundraising
- Simple Marketplace
- Basic Provenance
- Asset transfer
- Lottery
- Delegated Voting
- Smart contract IOT - devices
- Legal Agreements
- Payment Splitting

Stacks 2.0 Testnet Rollout

Testnet launch broken into four phases:

- Neon, Argon, Krypton, Xenon → Mainnet!

Benefits of testnet:

- Testnet $BTC & $STX.
  - No monetary value or attack incentive.
- Parallel network built for testing purposes.
  - Not losing value with txn fees.
- Let community tinker before final product.

	Phase 1	Phase 2	Phase 3	Phase 4
Simple Proof of Transfer mining	•	•		
Send and receive STX	•	•	•	•
Deploy Clarity contracts	•	•	•	•
New Stacks Explorer		•	•	•
stacks-transactions-js		•	•	•
Transaction signing		•	•	•
Proof of Transfer mining			•	•
Stacking			•	•
New Stacks Wallet			•	•
Microblocks				•
Testing upgrade to Stacks 2.0				•
Integration with Bitcoin testnet				•

Testnet Activities

- Download the stacks-blockchain software and spin up a **node**.
- Turn your node into a **miner** and participate in PoX consensus.
- Create a **wallet** with the JavaScript CLI.
- Try out earning $BTC rewards with **stacking.**

- Build a **Clarity smart contract** and deploy it to testnet.
- Send **testnet transactions** between accounts.
- Play with the new testnet **explorer** to analyze transactions.
  - **Sandbox**: Deploy & Call Contracts, get testnet $STX, transfer $STX, and more.
- Submit **bug bounties** to earn $BTC.

Tutorials

Running a Testnet **Node**

Running a testnet node

Learn how to set up and run a Stacks 2.0 testnet node.

Beginners 30 minutes

Introduction

The Stacks 2.0 testnet is currently in development. As part of the testnet, you can run a node and connect it to a public network. This guide will walk you through downloading and running your own node in the testnet network.

Prerequisites

Note: If you use Linux, you may need to manually install `libssl-dev` and other packages. In your command line, run the following to get all packages:

```
sudo apt-get install build-essential cmake libssl-dev pkg-config
```

https://docs.blockstack.org/stacks-blockchain/testnet-node

Starting a **Miner**

Mine Stacks tokens

Set up and run a miner on the Stacks 2.0 testnet

Beginners 10 minutes

Introduction

Make sure you've followed our guide for getting a Stacks 2.0 Testnet node up and running, once completed it's only a few more steps to run a proof-of-burn miner on the testnet.

Running a testnet node Tutorial
Learn how to set up and run a Stacks 2.0 testnet node.

Running a miner

First, we need to generate a keychain. With this keychain, we'll get some testnet BTC from a faucet, and then use that BTC to start mining.

https://docs.blockstack.org/mining

Prerequisites

- Command Line Interface (CLI)
  - Used to interact with the blockchain and enter commands.

- RUST
  - Programming language built for performance, reliability, and productivity. Needed for your machine to run node software successfully.

- Linux Users: Libssl-dev
  - API needed for Linux OS to process stacks-blockchain software.

```
sudo apt-get install build-essential cmake libssl-dev pkg-config
```

- Stacks-blockchain Github Repository
  - Source code for running a node on the Stacks 2.0 testnet.

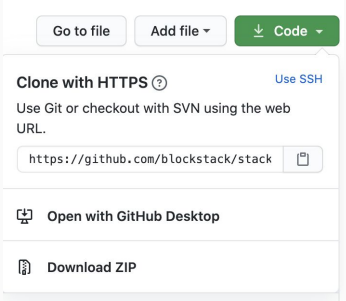
Downloading RUST & stacks-blockchain

- Installing RUST

```
curl --proto '=https' --tlsv1.2 -sSf https://sh.rustup.rs | sh
```

```
1) Proceed with installation (default)
2) Customize installation
3) Cancel installation
>
```

- Installing ‘stacks-blockchain’
  - https://github.com/blockstack/stacks-blockchain
- From Github Website



- From Command Line

```
git clone https://github.com/blockstack/stacks-blockchain.git
```

Installing stacks-blockchain

- Install the ‘stacks-blockchain’ software

```
cargo install --path ./testnet/stacks-node
```

- Important! Change your active directory to the stacks-blockchain folder.

```
cd stacks-blockchain
```

- Start your node!

```
stacks-node argon
```

- Evaluate Log Output

```
INFO [1588108047.585] [src/chainstate/stacks/index/marf.rs:732]
```

```
First-ever block 0f9188f13cb7b2c71f2a335e3a4fc328bf5beb436012afca590b1a11466e2206
```

Stacks-blockchain Output

First Block Found

Addresses Credited STX

```
[➜  ~ cd Documents
[➜  Documents cd Atom\ Projects
[➜  Atom Projects cd stacks-blockchain-master
[➜  stacks-blockchain-master stacks-node argon
INFO [1598224092.473] [testnet/stacks-node/src/run_loop/neon.rs:98] [ThreadId(1)] Follower node: starting up
ERROR [1598224092.484] [src/chainstate/stacks/index/storage.rs:1190] [ThreadId(1)] Not found (no file is open)
INFO [1598224092.485] [src/chainstate/stacks/index/marf.rs:751] [ThreadId(1)] First-ever block 0f9188f13cb7b2c71f2a335e3a4fc328bf5beb436
012afca590b1a11466e2206 in /tmp/stacks-testnet-3dea2b0b76ef1c0d/burnchain/db/bitcoin/regtest/sortition.db/marf
INFO [1598224165.559] [src/burnchains/bitcoin/spv.rs:682] [ThreadId(1)] Truncate received headers from block range 2000-3433 to range 20
00-1
INFO [1598224166.202] [src/burnchains/burnchain.rs:704] [ThreadId(1)] Node will fetch burnchain blocks 0-3434...
INFO [1598224310.420] [src/chainstate/stacks/index/marf.rs:751] [ThreadId(1)] First-ever block 8aeecfa0b9f2ac7818863b1362241e4f32d06b100
ae9d1c0fbcc4ed61b91b17a in /tmp/stacks-testnet-3dea2b0b76ef1c0d/chainstate/chain-00000080-testnet/vm/clarity/marf
INFO [1598224310.462] [src/chainstate/stacks/db/accounts.rs:183] [ThreadId(1)] STB44HYPYAT2BB2QE513NSP81HTMYWBJP02HPGK6 credited: 100000
00000000000 uSTX
INFO [1598224310.463] [src/chainstate/stacks/db/accounts.rs:183] [ThreadId(1)] ST11NJTTKGVT6D1HY4NJRVQWMQM7TVAR091EJ8P2Y credited: 10000
000000000000 uSTX
INFO [1598224310.463] [src/chainstate/stacks/db/accounts.rs:183] [ThreadId(1)] ST1HB1T8WRNBYB0Y3T7WXZS38NKKPTBR3EG9EPJKR credited: 10000
000000000000 uSTX
INFO [1598224310.464] [src/chainstate/stacks/db/accounts.rs:183] [ThreadId(1)] STRYYQQ9M8KAF4NS7WNZQYY59X93XEKR31JP64CP credited: 100000
00000000000 uSTX
INFO [1598224310.468] [src/chainstate/stacks/index/marf.rs:751] [ThreadId(1)] First-ever block 8aeecfa0b9f2ac7818863b1362241e4f32d06b100
ae9d1c0fbcc4ed61b91b17a in /tmp/stacks-testnet-3dea2b0b76ef1c0d/chainstate/chain-00000080-testnet/vm/index
BOOTSTRAP WITH [Neighbor { addr: facade01+80000000://V4(35.236.218.197:20444), public_key: Secp256k1PublicKey { key: PublicKey(aa164d5d7
a91c179f60534e75bfd54583715095f00ee3d535358710161f2d48d275f6fe25afbbd24eb89d77401918cad7df08712fec5b8362a139c7a0d4c3caf), compressed: fa
lse }, expire_block: 99999, last_contact_time: 0, allowed: 0, denied: 0, asn: 0, org: 0, in_degree: 0, out_degree: 0 }]
INFO [1598224310.959] [src/net/db.rs:141] [ThreadId(1)] Peer's public key: 04f276fce9f3def7124e0678ebb670d71b6b180d2a4d1ab2031d065ac094e
45f9aefef353678cd7e26f9ba821d5d1a2851099f5ae941fff96cb9f7b6dd196f01a3
INFO [1598224310.994] [testnet/stacks-node/src/neon_node.rs:647] [ThreadId(1)] Bound HTTP server on: 0.0.0.0:20443
INFO [1598224310.994] [testnet/stacks-node/src/neon_node.rs:648] [ThreadId(1)] Bound P2P server on: 0.0.0.0:20444
INFO [1598224311.005] [testnet/stacks-node/src/run_loop/neon.rs:118] [ThreadId(1)] Begin run loop
INFO [1598224311.385] [src/burnchains/burnchain.rs:704] [ThreadId(1)] Node will fetch burnchain blocks 3434-3435...
INFO [1598224311.570] [testnet/stacks-node/src/neon_node.rs:882] [ThreadId(1)] Received burnchain block #3435 including block_commit_op
(winning) - mo8CosqCssC3jgdQZEiHqbE5m1kHTGoGCX
```

Stacks-blockchain Output

Processing Blocks

Contract Call

```
l to ST3A28CNCDFR1RTEEHVTVV9JF3SEQ3DE7XX1A6YCC.hello_world.ClarityName("set-value") args [Buffe
r(666f6f), Buffer(626172)] returned Response(ResponseData { committed: true, data: UInt(1) })
INFO [1598225435.071] [src/chainstate/stacks/db/transactions.rs:564] [ThreadId(5)] Contract-cal
l to ST3A28CNCDFR1RTEEHVTVV9JF3SEQ3DE7XX1A6YCC.hello_world.ClarityName("set-value") args [Buffe
r(666f6f), Buffer(626172)] cost ExecutionCost { write_length: 107, write_count: 1, read_length:
 1573, read_count: 2, runtime: 1780 }
INFO [1598225435.320] [src/net/relay.rs:800] [ThreadId(5)] Processing newly received blocks: 6
INFO [1598225435.741] [src/net/relay.rs:800] [ThreadId(5)] Processing newly received blocks: 6
INFO [1598225436.480] [src/net/relay.rs:800] [ThreadId(5)] Processing newly received blocks: 6
INFO [1598225436.856] [src/net/relay.rs:800] [ThreadId(5)] Processing newly received blocks: 6
INFO [1598225437.225] [src/net/relay.rs:800] [ThreadId(5)] Processing newly received blocks: 6
INFO [1598225437.694] [src/net/relay.rs:800] [ThreadId(5)] Processing newly received blocks: 6
INFO [1598225438.142] [src/net/relay.rs:800] [ThreadId(5)] Processing newly received blocks: 6
INFO [1598225438.545] [src/net/relay.rs:800] [ThreadId(5)] Processing newly received blocks: 6
INFO [1598225439.071] [src/net/relay.rs:800] [ThreadId(5)] Processing newly received blocks: 6
0x4576656e74212048656c6c6f20776f726c64
INFO [1598225439.157] [src/chainstate/stacks/db/transactions.rs:563] [ThreadId(5)] Contract-cal
l to ST8QVG6WC82C54AYPCVNN5VVQZVDCBD9S709VZ4Y.hello_world.ClarityName("set-value") args [Buffer
(666f6f), Buffer(626172)] returned Response(ResponseData { committed: true, data: UInt(1) })
INFO [1598225439.157] [src/chainstate/stacks/db/transactions.rs:564] [ThreadId(5)] Contract-cal
l to ST8QVG6WC82C54AYPCVNN5VVQZVDCBD9S709VZ4Y.hello_world.ClarityName("set-value") args [Buffer
(666f6f), Buffer(626172)] cost ExecutionCost { write_length: 107, write_count: 1, read_length:
1573, read_count: 2, runtime: 1780 }
INFO [1598225439.184] [src/chainstate/stacks/db/transactions.rs:563] [ThreadId(5)] Contract-cal
l to ST8QVG6WC82C54AYPCVNN5VVQZVDCBD9S709VZ4Y.hello_world.ClarityName("set-value") args [Buffer
(666f6f), Buffer(626172)] returned Response(ResponseData { committed: true, data: UInt(1) })
INFO [1598225439.184] [src/chainstate/stacks/db/transactions.rs:564] [ThreadId(5)] Contract-cal
l to ST8QVG6WC82C54AYPCVNN5VVQZVDCBD9S709VZ4Y.hello_world.ClarityName("set-value") args [Buffer
```

Creating Keychain

- Create a keychain
  - With this keychain, we'll get some testnet BTC from a faucet, and then use that BTC to start mining.
- Simplest way is using ‘blockstack-cli’

```
npx blockstack-cli@1.1.0-beta.1 make_keychain -t
```

- After this runs, you'll probably see some installation logs, and at the end you should see some JSON that looks like this:

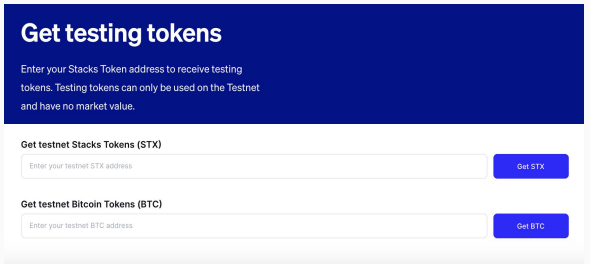
```
```json
{
  "mnemonic": "exhaust spin topic distance hole december impulse gate century absent breeze ostrich armed clerk oak peace want scrap auction sniff cradle siren blur blur",
  "keyInfo": {
    "privateKey": "2033269b55026ff2eddaf06d2e56938f7fd8e9d697af8fe0f857bb5962894d5801",
    "address": "STTX57EGWW058FZ6WG3WS2YRBQ8HDFGBKEFBNXTF",
    "btcAddress": "mkRYR7KkPB1wjxNjVz3HByqAvVz8c4B6ND",
    "index": 0
  }
}
```
```

Funding the miner

- Get your BTC address
  - The ‘btcAddress’ field from the JSON snippet

```json
{
  "mnemonic": "exhaust spin topic distance hole december impulse gate century absent breeze ostrich armed clerk oak peace want scrap auction sniff cradle siren blur blur",
  "keyInfo": {
    "privateKey": "2033269b55026ff2eddaf06d2e56938f7fd8e9d697af8fe0f857bb5962894d5801",
    "address": "STTX57EGWW058FZ6WG3WS2YRBQ8HDFGBKEFBNXTE",
    "btcAddress": "mkRYR7KkPB1wjxNjVz3HByqAvVz8c4B6ND",
    "index": 0
  }
}
```

- Request testnet BTC from the Stacks 2.0 Testnet Faucet
  - “Get testnet Bitcoin Tokens (BTC)” Field
  - https://www.blockstack.org/testnet/faucet

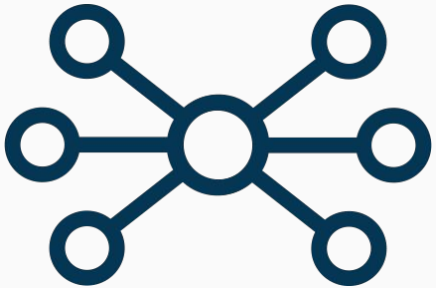


  - You'll be sent 0.5 testnet BTC to that address. Don't lose this information - we'll need to use the ‘privateKey’ field later on.

Configure Node

- We have to configure the node so it knows to use the provided BTC keychain
- In the ‘stacks-blockchain’ folder, create a new file called:

```
testnet/stacks-node/conf/testnet-miner-conf.toml
```



- In that new file, replace the ‘seed’ value with the ‘privateKey’ value from the JSON snippet.

```
01  [node]
02  rpc_bind = "0.0.0.0:20443"
03  p2p_bind = "0.0.0.0:20444"
04  bootstrap_node = "048dd4f26101715853533
05  # Enter your private key here!
06  seed = "replace-with-your-private-key"
07  miner = true
```

Configuration File Explained

- Configuring Burnchain

```
[burnchain]
chain = "bitcoin"
mode = "argon"
peer_host = "argon.blockstack.org"
rpc_port = 18443
peer_port = 18444
```

- Configuring Balances

```
[[mstx_balance]]
address = "STB44HYPYAT2BB2QE513NSP81HTMYWBJPO2HPGK6"
amount = 10000000000000000
[[mstx_balance]]
address = "ST11NJTTKGVT6D1HY4NJRVQWMQM7TVARO91EJ8P2Y"
amount = 10000000000000000
[[mstx_balance]]
address = "ST1HB1T8WRNBYBOY3T7WXZS38NKKPTBR3EG9EPJKR"
amount = 10000000000000000
[[mstx_balance]]
address = "STRYYQQ9M8KAF4NS7WNZQYY59×93XEKR31JP64CP"
amount = 10000000000000000
```

Start Running Your Miner!

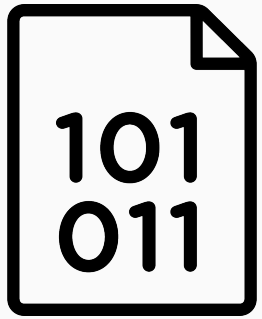
Enter this command to begin mining:

```
stacks-node start --config=./testnet/stacks-node/conf/testnet-miner-conf.toml
```

Creating an Optimized Binary

- If you want to host a node on a server somewhere, you might want to generate an optimized binary.
  - Run this command to create the binary

```
cd testnet/stacks-node
cargo build --release --bin stacks-node
```



- Now that your optimized binary is compiled, go ahead and run it.

```
cd ../..
./target/release/stacks-node start --config=./testnet/conf/argon-follower-conf.toml
```

Enable Debug Logging

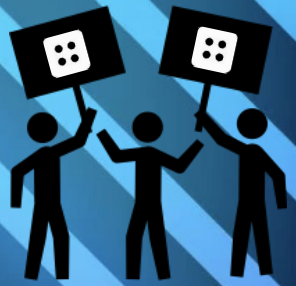
- In case you are running into issues, or would like to see more detailed logging, you can run your node with debug logging enabled.
- In the command line, run:

```
BLOCKSTACK_DEBUG=1 stacks-node argon
```

Why Running a Node is Important

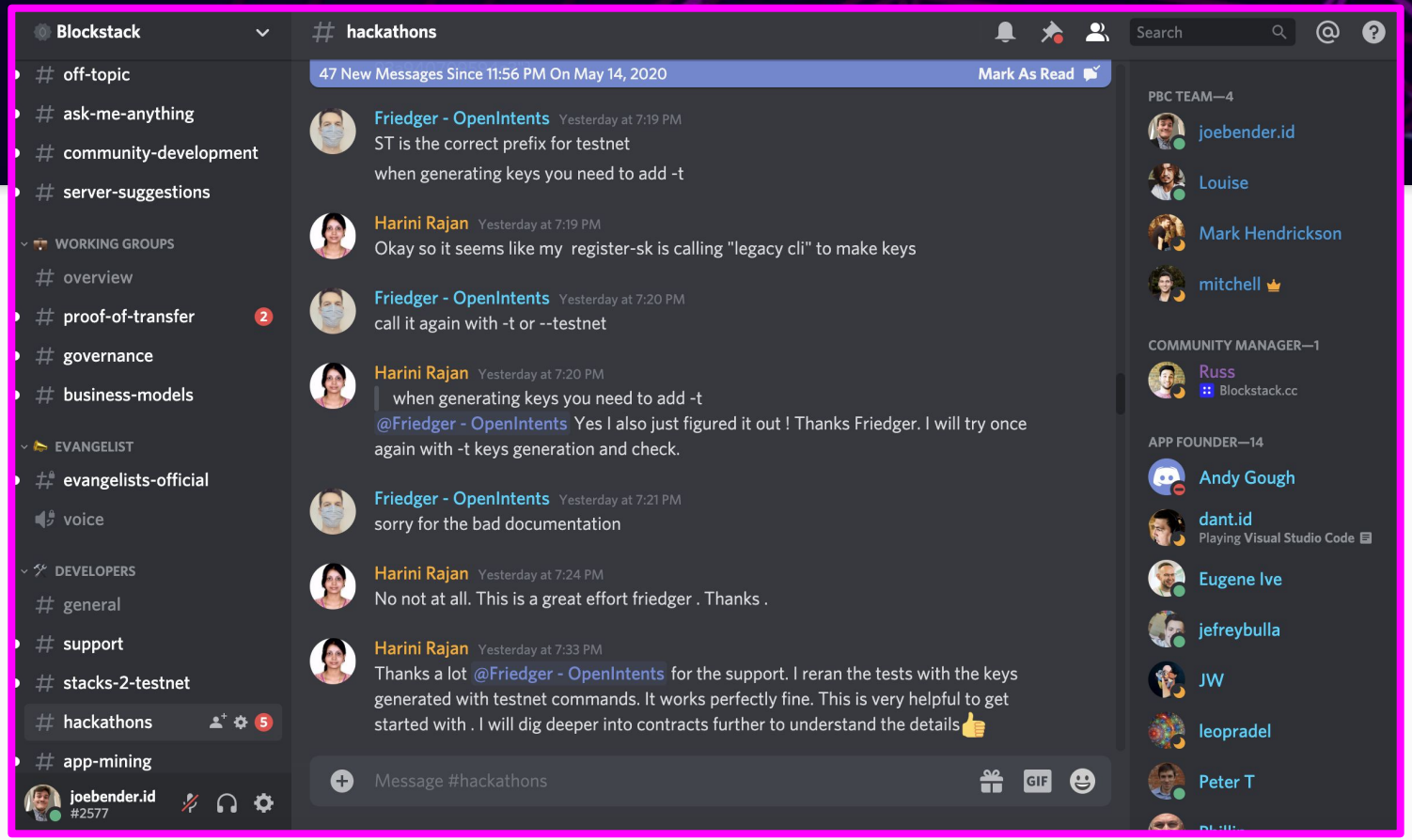
- **Trust**
  - To have a copy of the blockchain that you have validated yourself, rather than having to trust a third party to be honest about the state of the chain.
- **Control**
  - You do not depend on third parties for broadcasting your transactions to the network.
- **Scalability**
  - More nodes & miners = faster network!
- **It's inexpensive!**
  - Unlike Proof-of-Work, Stacks 2.0 miners don't need overpriced hardware to begin participating.
- **Increase Network Security**
  - By adding a node that can validate the state of the chain, you are actively making the blockchain safer.
- **Participate in Mining**
  - Running a node is the first step to mining.
- **Autonomy**
  - When a blockchain undergoes a hard fork, lightweight nodes will automatically follow the chain with the biggest accumulated difficulty.
- **It's easy!**
  - All you need is a computer, and a few commands in the terminal.

How to get involved!

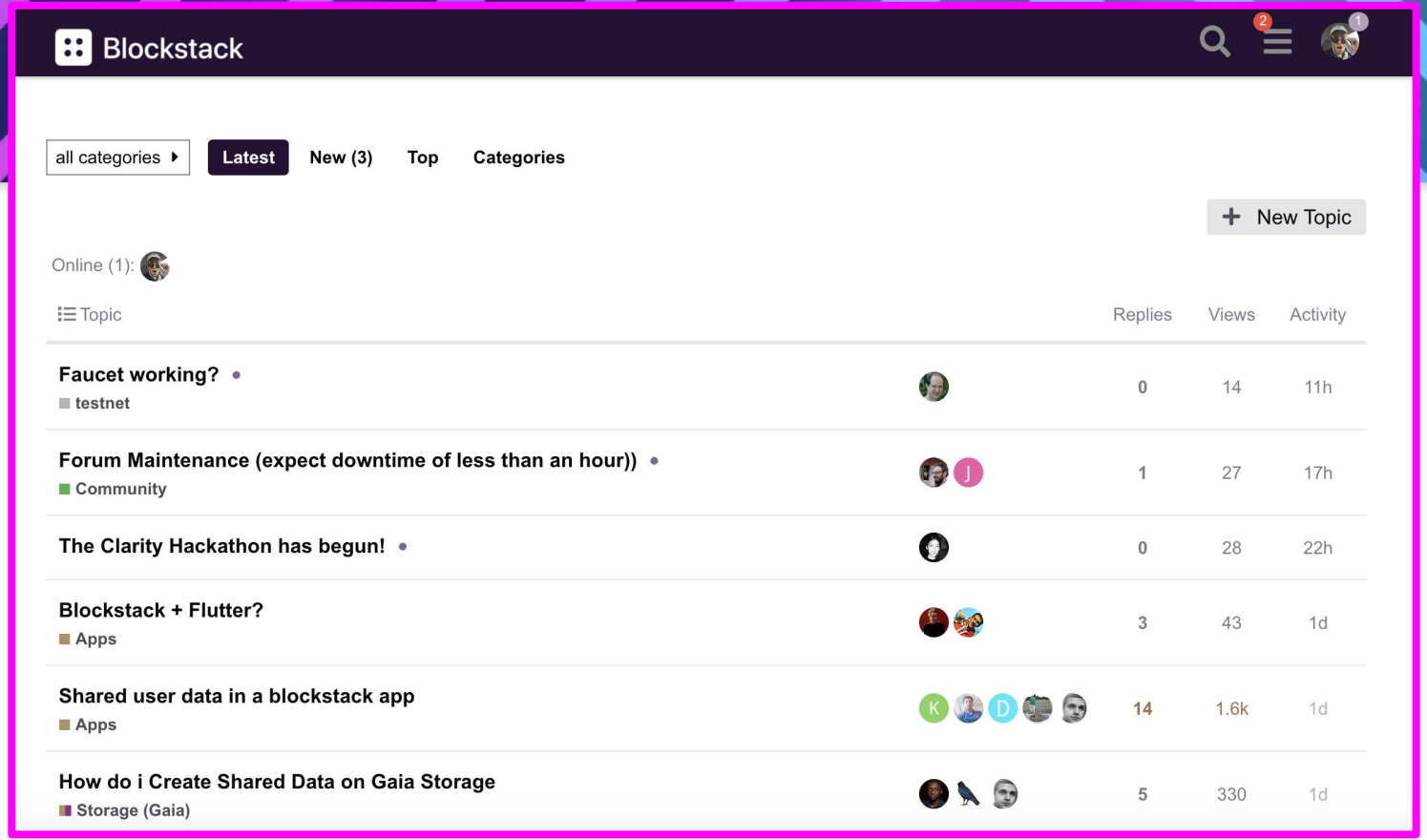


- Checkout our updated Documentation
  - https://docs.blockstack.org/
- Join the Forum
  - https://forum.blockstack.org/
- Join our Discord
  - http://discord.gg/unFGwwu
- Follow Blockstack on Twitter
  - https://twitter.com/blockstack
- Download the Stacks Wallet
  - https://wallet.blockstack.org/
- Read our Whitepapers
  - https://www.blockstack.org/papers
- Contribute Code
  - github.com/blockstack
- Host meetups, give talks, and spread the word!
  - https://community.blockstack.org/
- Complete Testnet Bounties
  - https://www.blockstack.org/testnet/bounties
- Participate in the Stacks 2.0 Hackathon Series
  - https://community.blockstack.org/stacks-series
- Sign-up for the Blockstack Newsletter
  - http://newsletters.blockstack.org/
- Experiment with the Stacks 2.0 Testnet
  - https://www.blockstack.org/testnet

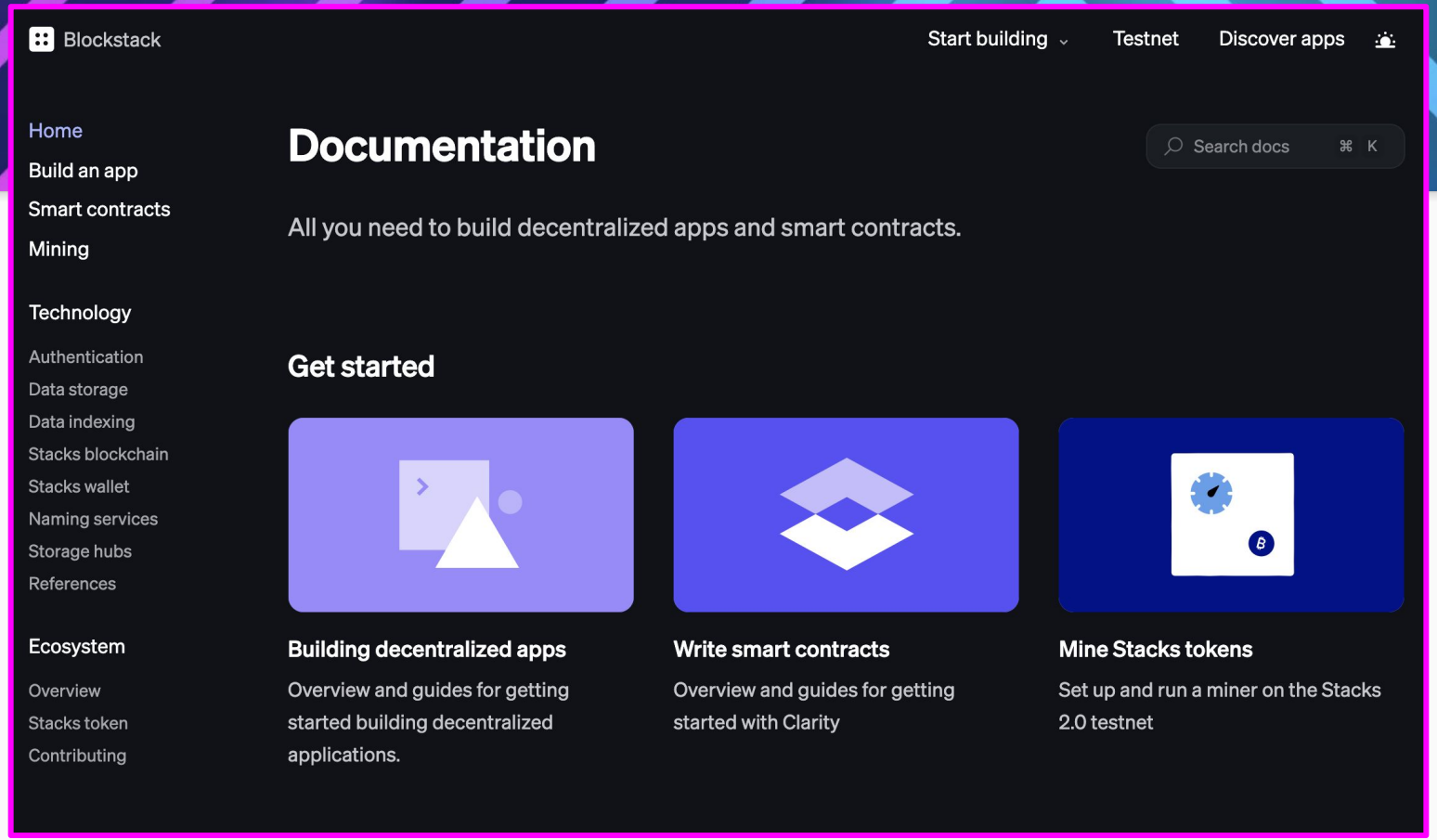
Blockstack Discord



Blockstack Forum



Blockstack Docs



02

#HackStacks

SEPTEMBER 30 - NOVEMBER 11

#HackStacks is a virtual hackathon focused on leveraging the power of PoX, a novel, Bitcoin-secured mining mechanism enabling endless potential for new business models. PoX will enable mining and stacking, unique functionality that offer all stakeholders the opportunity to share in value creation on the Stacks 2.0 network. #HackStacks supports and rewards developers for building tools and products that make mining and stacking more accessible to the Stacks community.

03

#HackDeFi

OCTOBER 20 - NOVEMBER 20

Taking Defi to the next level with Bitcoin. Build safe, smart, secure DeFi projects anchored to Bitcoin by leveraging Proof of Transfer, Stacking, and Clarity smart contracts on the Stacks 2.0 testnet and reap the rewards.

Discussion!

- What do you think the biggest hurdle would be for a non-crypto enthusiast?
- What balance of blockchain education vs. obfuscation do you think is best?
- What will be the catalyst that drives the mainstream public to try crypto?
- Do you believe in the ‘Crypto already has a 1%’ assertion?
- Will digital assets eventually be valued in the same way as physical assets?
- Does blockchain need more developers, or better marketing?
- What decentralized application would you most desire implemented into your life?

Thanks!

joe@blockstack.com
@josephbender
www.blockstack.org