Testing Blockchain Transactions

FinTech

Lesson 19.3

Class Objectives

By the end of this lesson, you will be able to:

- Use a blockchain explorer to visualize each part of a transaction.
- Develop code by using Python and Web3.py to connect to a local Ganache blockchain.
- Use Web3.py in conjunction with Ganache to test your transaction
- Formulate code to sign and send a transaction by using Web3.py.
- Use Streamlit and Web3.py together to build an application that communicates with the blockchain.
- Test a blockchain web application by using Ganache.

WELCOME

Let’s begin by recapping the previous unit and lessons.

What is the difference between a hot wallet and a cold wallet?

Hot Wallet

- Software or hardware device
- Often or always connected to the internet
- Easier to access account information
- Less secure

Cold Wallet

- Hardware device
- Rarely connected to the internet
- More difficult to access account information
- Extremely secure

What are public and private keys?

Public and Private Keys

Public key

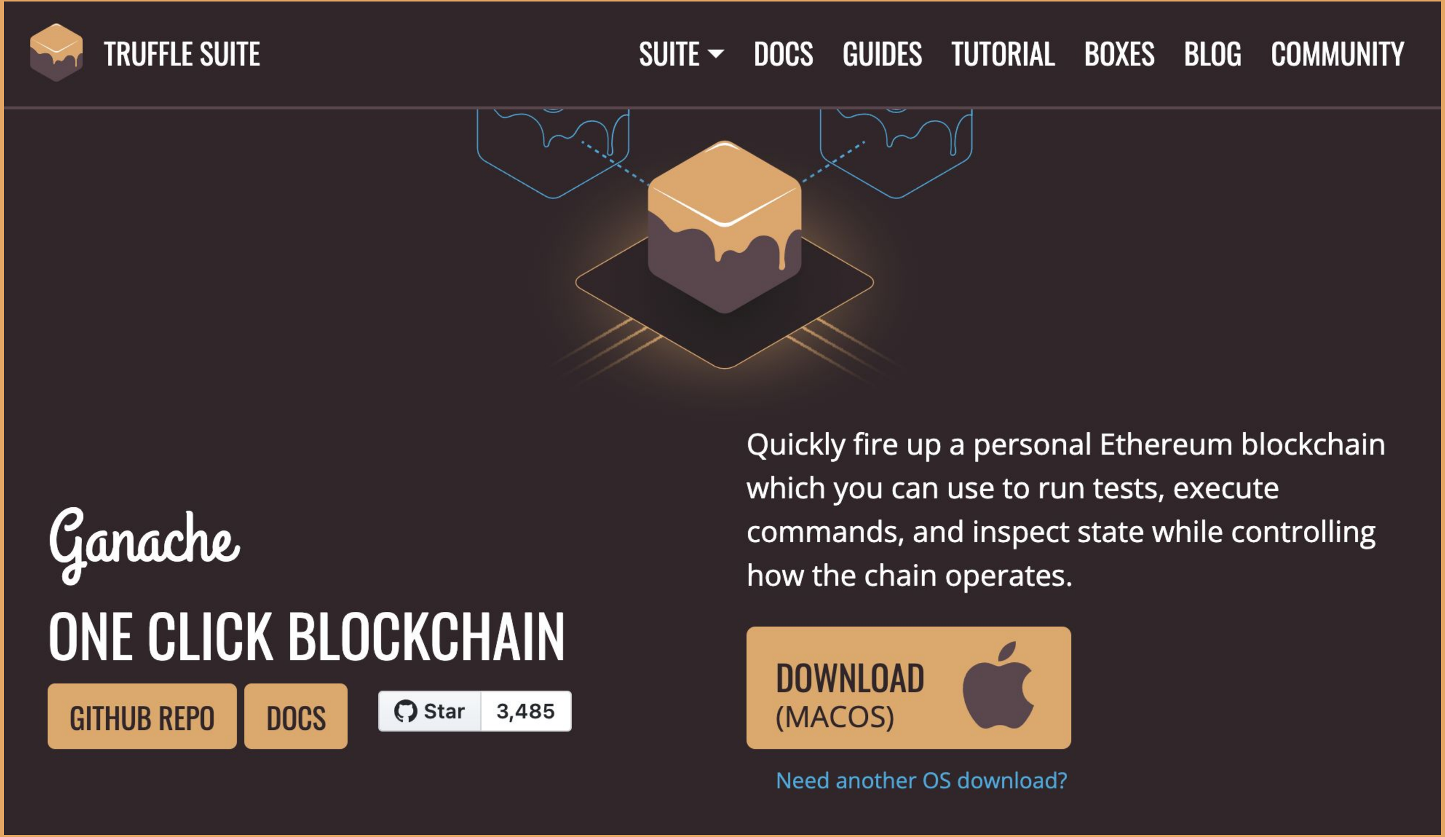
- Is used to encrypt plaintext to convert it to cipher text.
- Known by all users of a blockchain.
- Can be used by all members of the blockchain community to send transactions to a particular user.

Private key

- Is used by the receiver to decrypt the cipher text in order to read messages or send ether.
- Known only by the account owner.
- Is needed to send ether to other accounts.

Introduction to Ganache

Confirm that you have downloaded and installed Ganache.



ONE CLICK BLOCKCHAIN

Quickly fire up a personal Ethereum blockchain which you can use to run tests, execute commands, and inspect state while controlling how the chain operates.

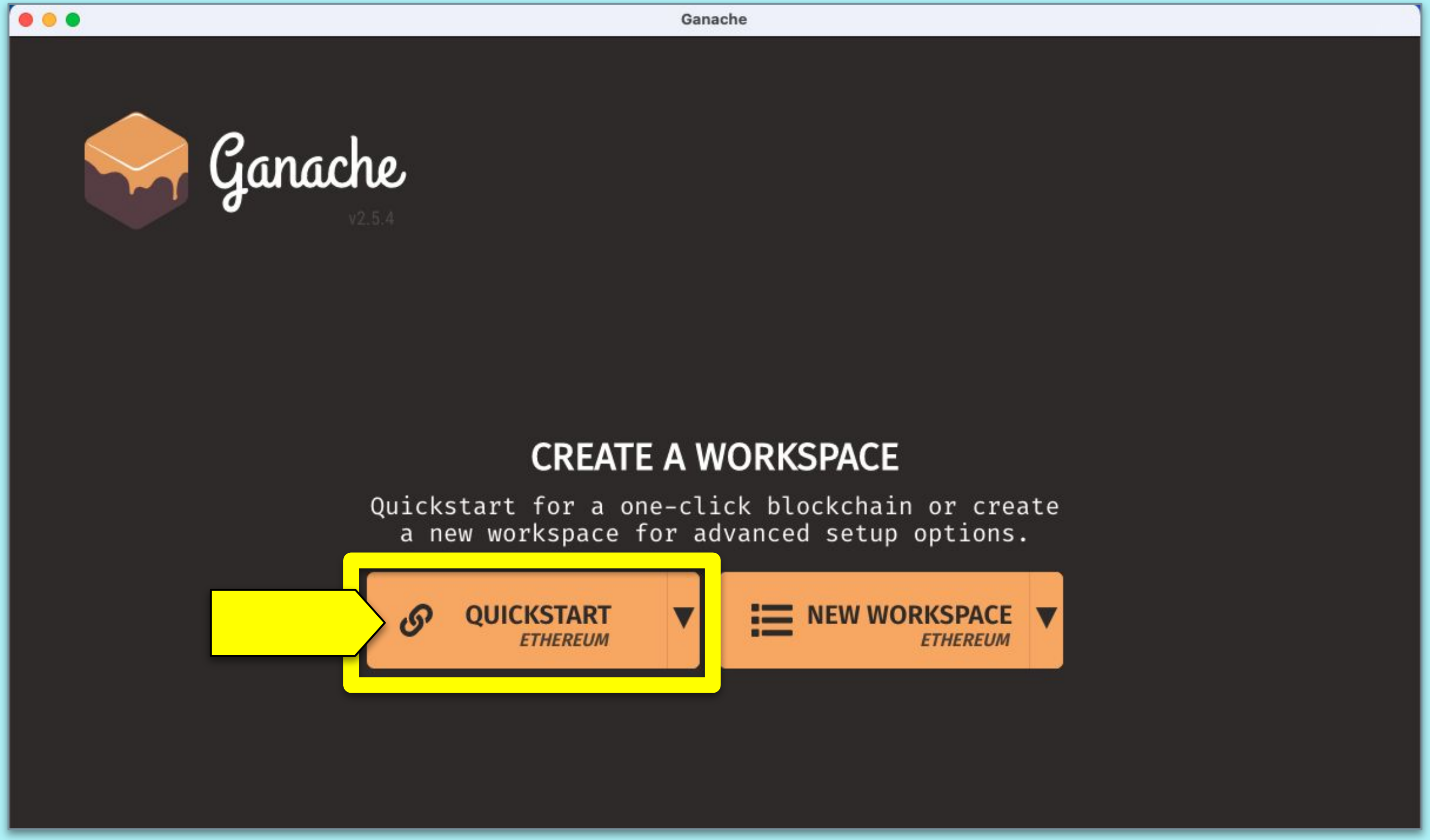
Ganache

Ganache is a personal local blockchain with accounts that are preloaded with ether.

The ether in these accounts has no real value on the Ethereum blockchain, as this blockchain is local only.

Ganache is a useful tool for testing transactions and smart contracts.

Launch Ganache and select **Quickstart Ethereum**.

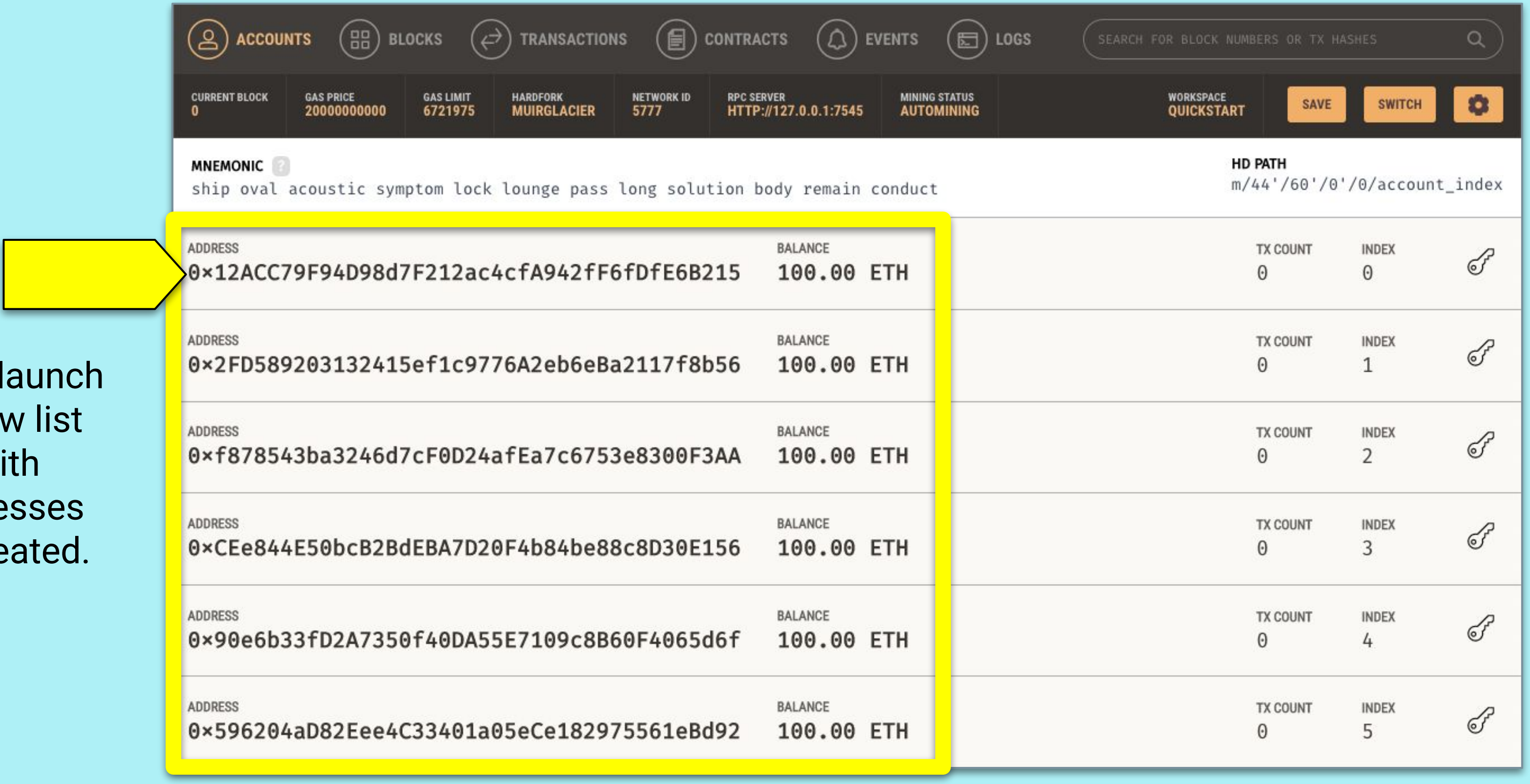


CREATE A WORKSPACE

Quickstart for a one-click blockchain or create a new workspace for advanced setup options.

The result is a list of 10 different account addresses with a balance of 100 ETH in each.

Each time we launch Ganache, a new list of accounts with different addresses and keys is created.



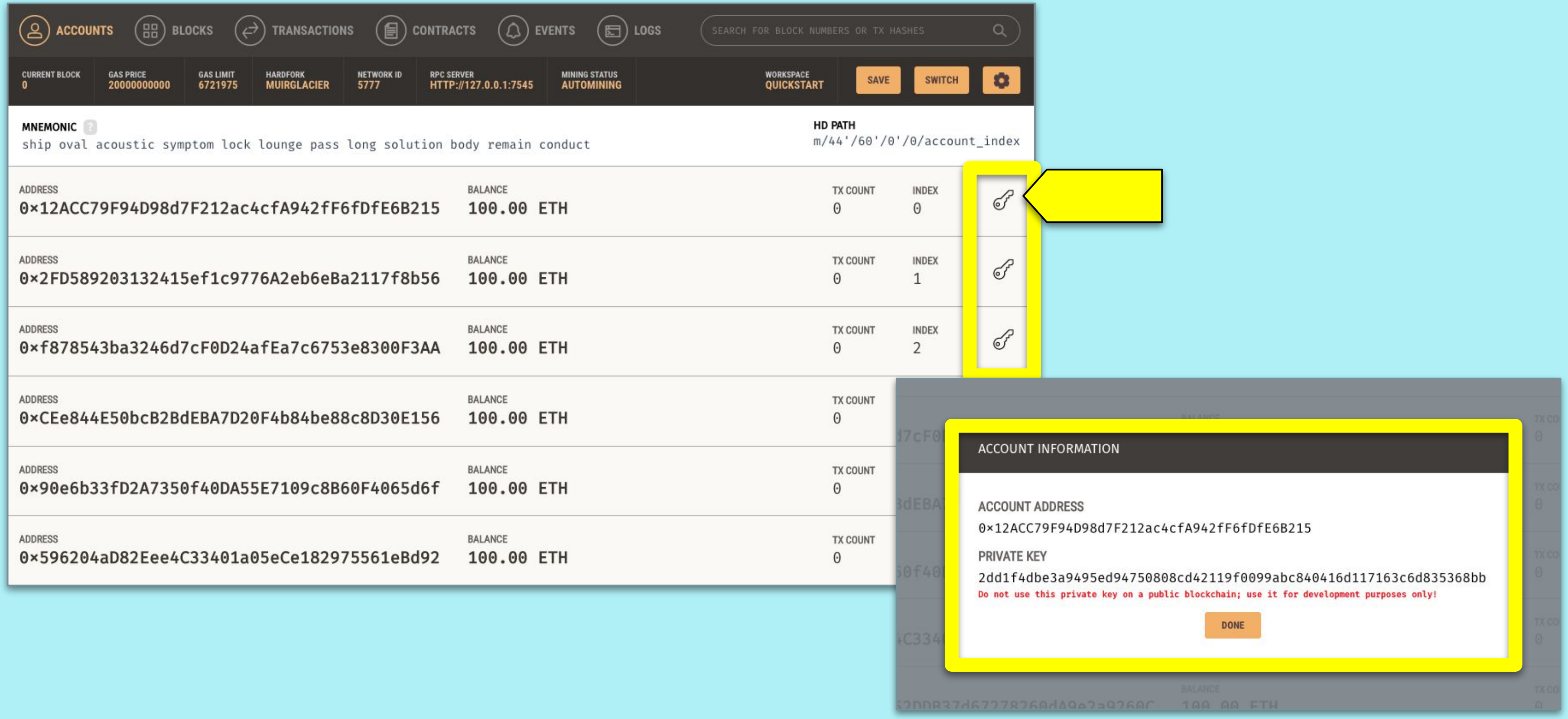
CURRENT BLOCK 0 | GAS PRICE 20000000000 | GAS LIMIT 6721975 | HARDFORK MUIRGLACIER | NETWORK ID 5777 | RPC SERVER HTTP://127.0.0.1:7545 | MINING STATUS AUTOMINING | WORKSPACE QUICKSTART

MNEMONIC
ship oval acoustic symptom lock lounge pass long solution body remain conduct

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

ADDRESS	BALANCE	TX COUNT	INDEX
0×12ACC79F94D98d7F212ac4cfA942fF6fDfE6B215	100.00 ETH	0	0
0×2FD589203132415ef1c9776A2eb6eBa2117f8b56	100.00 ETH	0	1
0×f878543ba3246d7cF0D24afEa7c6753e8300F3AA	100.00 ETH	0	2
0×CEe844E50bcB2BdEBA7D20F4b84be88c8D30E156	100.00 ETH	0	3
0×90e6b33fD2A7350f40DA55E7109c8B60F4065d6f	100.00 ETH	0	4
0×596204aD82Eee4C33401a05eCe182975561eBd92	100.00 ETH	0	5

Clicking an account's key symbol opens a window that reveals the test account's private key as well as the address.



CURRENT BLOCK 0 | GAS PRICE 20000000000 | GAS LIMIT 6721975 | HARDFORK MUIRGLACIER | NETWORK ID 5777 | RPC SERVER HTTP://127.0.0.1:7545 | MINING STATUS AUTOMINING | WORKSPACE QUICKSTART

MNEMONIC
ship oval acoustic symptom lock lounge pass long solution body remain conduct

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

ADDRESS	BALANCE	TX COUNT	INDEX
0×12ACC79F94D98d7F212ac4cfA942fF6fDfE6B215	100.00 ETH	0	0
0×2FD589203132415ef1c9776A2eb6eBa2117f8b56	100.00 ETH	0	1
0×f878543ba3246d7cF0D24afEa7c6753e8300F3AA	100.00 ETH	0	2
0×CEe844E50bcB2BdEBA7D20F4b84be88c8D30E156	100.00 ETH	0	
0×90e6b33fD2A7350f40DA55E7109c8B60F4065d6f	100.00 ETH	0	
0×596204aD82Eee4C33401a05eCe182975561eBd92	100.00 ETH	0	

ACCOUNT INFORMATION

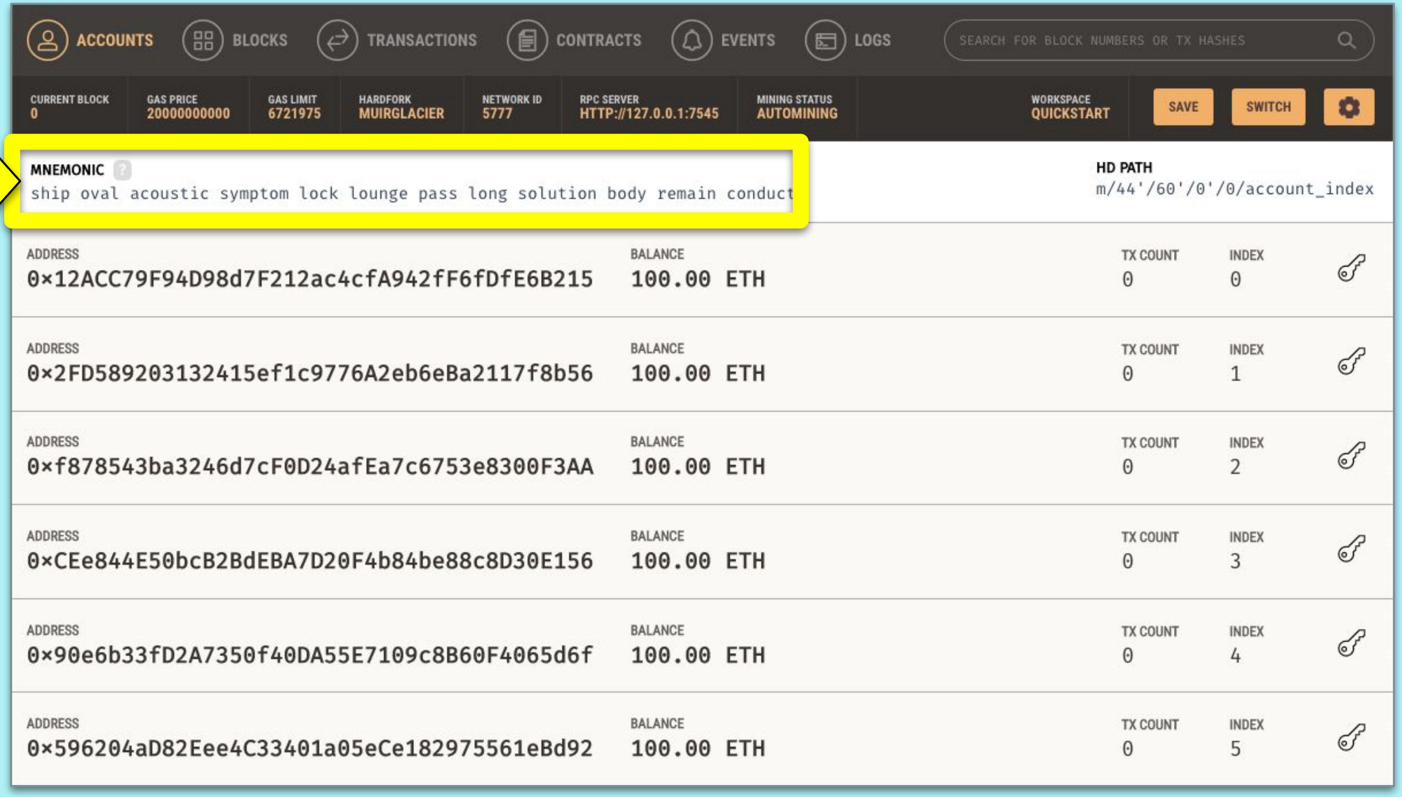
ACCOUNT ADDRESS
0×12ACC79F94D98d7F212ac4cfA942fF6fDfE6B215

PRIVATE KEY
2dd1f4dbe3a9495ed94750808cd42119f0099abc840416d117163c6d835368bb
Do not use this private key on a public blockchain; use it for development purposes only!

DONE

The mnemonic seed phrase is in the top left part of the interface.

This phrase can be added to a `.env` file, which we'll do later in the lesson, to enable Ganache to run as the provider for transactions.



CURRENT BLOCK	GAS PRICE	GAS LIMIT	HARDFORK	NETWORK ID	RPC SERVER	MINING STATUS	WORKSPACE
0	20000000000	6721975	MUIRGLACIER	5777	HTTP://127.0.0.1:7545	AUTOMINING	QUICKSTART

MNEMONIC
ship oval acoustic symptom lock lounge pass long solution body remain conduct

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

ADDRESS	BALANCE	TX COUNT	INDEX
0x12ACC79F94D98d7F212ac4cfA942fF6fDfE6B215	100.00 ETH	0	0
0x2FD589203132415ef1c9776A2eb6eBa2117f8b56	100.00 ETH	0	1
0xf878543ba3246d7cF0D24afEa7c6753e8300F3AA	100.00 ETH	0	2
0xCEe844E50bcB2BdEBA7D20F4b84be88c8D30E156	100.00 ETH	0	3
0x90e6b33fD2A7350f40DA55E7109c8B60F4065d6f	100.00 ETH	0	4
0x596204aD82Eee4C33401a05eCe182975561eBd92	100.00 ETH	0	5

Time to Code

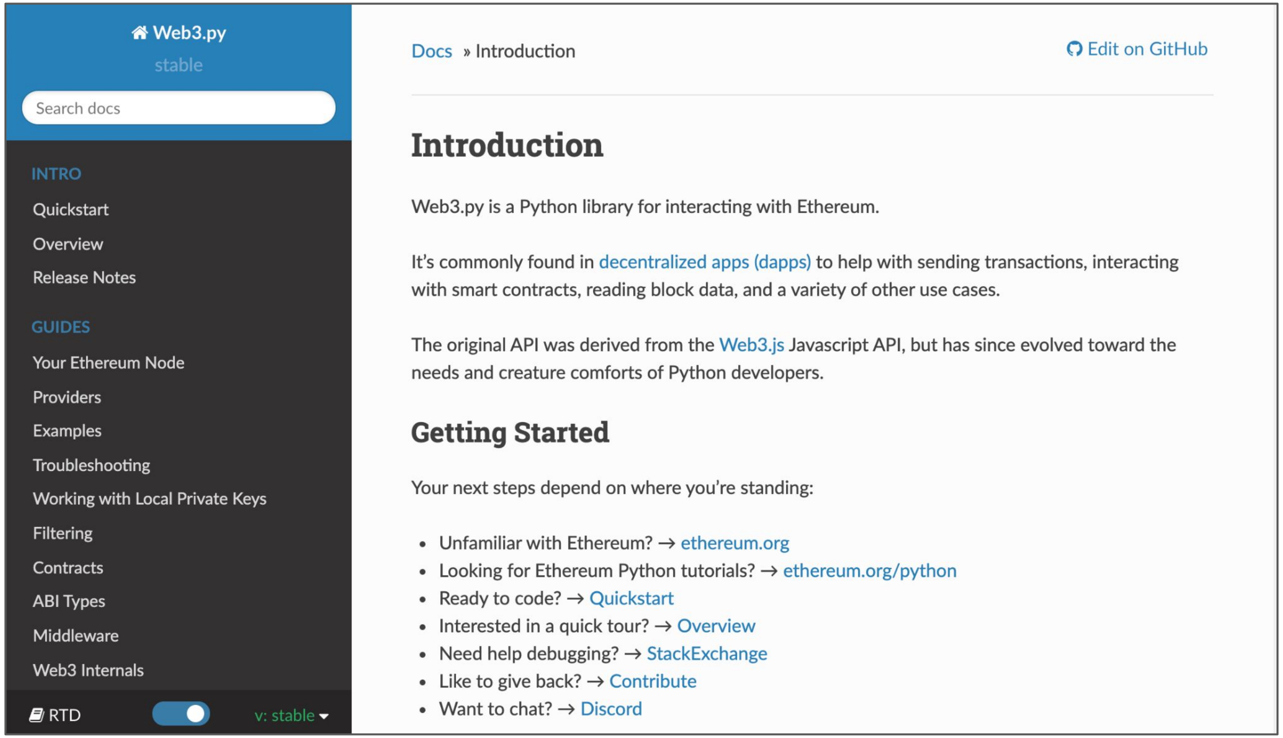
Using Streamlit with Web3.py

Suggested Time:

5 minutes

Using Streamlit with Web3.py

Web3.py will handle the transactions in the application you are building today.



Web3.py
stable
Search docs

INTRO
Quickstart
Overview
Release Notes

GUIDES
Your Ethereum Node
Providers
Examples
Troubleshooting
Working with Local Private Keys
Filtering
Contracts
ABI Types
Middleware
Web3 Internals

RTD v: stable

Docs » Introduction   Edit on GitHub

Introduction

Web3.py is a Python library for interacting with Ethereum.

It's commonly found in decentralized apps (dapps) to help with sending transactions, interacting with smart contracts, reading block data, and a variety of other use cases.

The original API was derived from the Web3.js Javascript API, but has since evolved toward the needs and creature comforts of Python developers.

Getting Started

Your next steps depend on where you're standing:

- Unfamiliar with Ethereum? → ethereum.org
- Looking for Ethereum Python tutorials? → ethereum.org/python
- Ready to code? → Quickstart
- Interested in a quick tour? → Overview
- Need help debugging? → StackExchange
- Like to give back? → Contribute
- Want to chat? → Discord

Using Streamlit with Web3.py

In most decentralized applications that are built on the blockchain, Web3 is used to interact with smart contracts and read the block data. You will use it to send transactions.

The `web3. eth. Contract` objects can help developers interact with smart contracts on the Ethereum blockchain.

When you create a new contract object, you give it the JSON interface of the respective smart contract, and web3 will auto-convert all calls into low-level ABI calls over the remote procedure call (RPC).

Today we will connect web3 to Streamlit to enable our applications to interact with local blockchains.

Questions?

Time to Code

Ethereum and Streamlit

Suggested Time:

30 minutes

Ethereum and Streamlit

The focus of this activity is twofold:

01 To automate the Ethereum account functionality that we have learned up to now with the use of Python functions.

02 To integrate those Python functions with a Streamlit web application

Because Streamlit is involved, the activity will involve Python files rather than Jupyter notebooks.

Visual Studio Code will be the IDE used for this and the following activities.

Ethereum and Streamlit

In the `ethereum.py` file, you will create a Python function that does the following:

01 Accesses the `MNEMONIC` variable from the `.env` file.

02 Uses the `mnemonic` variable to create an HD `wallet`.

03 Uses the `wallet` to generate a public/private key pair.

04 Uses the `private` key to create an Ethereum `account`.

05 Returns the `account` from the function.

Activity: Automating Ethereum

In this activity, you will create a Streamlit web application that’s similar to the one that you’ll be asked to create in the homework assignment.

Suggested Time:

35 minutes

Time’s Up! Let’s Review.

Questions?

Break

Activity: Cats Mini-Project

In this activity, you will add functions to automate the process of accessing the balance from the Ganache blockchain, as well as sending a signed transaction.

You will then incorporate these functions into the Streamlit web application.

Suggested Time:

45 minutes

Time’s Up! Let’s Review.

Questions?

Structured Review

Structured Review

01

Are any activities that you want to review?

02

Let’s revisit key activities that are relevant for the homework assignment.

03

You can start the homework assignment, which is a combination of sections 1 and 3.

The End