How to create a vesting account with celestia-app

In this guide, we will learn how to create a vesting account using celestia-app for both a local devnet and on Mocha testnet.

note

The instructions for this tutorial are for acontinuous vesting account, if you'd like to make a delayed vesting account, just add the--delayed flag to your vesting transaction.

Local devnet

First,download and install celestia-app , selecting thenetwork and corresponding version that you would like to use.

Setting up the local devnet

Run the devnet

Next, change into theHOME/celestia-app directory and run the single-node-devnet script.

bash cd HOME /celestia-app ./scripts/build-run-single-node.sh cd HOME /celestia-app ./scripts/build-run-single-node.sh

Save the home directory path

At the top of the output, you will see a path to the "Home directory", find yours from the output (it will be unique every time):

bash ./scripts/build-run-single-node.sh Home

directory:

/var/folders/_8/ljj6hspn0kn09qf9fy8kdyh40000gn/T/celestia_app_XXXXXXXXXXXXX.XV92a3qx -->

Updating

go.mod ./scripts/build-run-single-node.sh Home

directory:

/var/folders/_8/ljj6hspn0kn09qf9fy8kdyh40000gn/T/celestia_app_XXXXXXXXXXXXX.XV92a3qx -->

Updating

go.mod And set the location as theCEIESTIA_APP_HOME variable. We will use this for the remainder of the devnet section.

bash export CEIESTIA_APP_HOME = /var/folders/_8/ljj6hspn0kn09qf9fy8kdyh40000gn/T/celestia_app_XXXXXXXXXXXXX.XV92a3qx export CEIESTIA_APP_HOME = /var/folders/_8/ljj6hspn0kn09qf9fy8kdyh40000gn/T/celestia_app_XXXXXXXXXXXXX.XV92a3qx
note

This does not replace thecelestia-appd binary that was installed withcelestia-appd , but builds and runs one in theHOME/celestia-app/build directory.

Check the version of the devnet

If you'd like to check the version of your local devnet, you can use:

bash cd HOME /celestia-app/build ./celestia-appd

version cd HOME /celestia-app/build ./celestia-appd

version

Next steps

Congratulations! You now have a private devnet running locally on your machine. The devnet is made up of one validator that is creating new blocks. This is the Celestia consensus network on your machine! The key that was created to run the validator also lives in a temporary directory for the devnet.

Now you are ready to test creating a vesting account on our devnet before going to Mocha, a live testnet.

Setting up vesting account on devnet

You already have one key setup, but you will need one more to create a vesting account.

Create a new key

First, create a vesting key:

bash cd HOME /celestia-app/build ./celestia-appd

keys

add

vesting-key

--home CEIESTIA_APP_HOME cd HOME /celestia-app/build ./celestia-appd

keys

add

vesting-key

--home CEIESTIA_APP_HOME You will see the address, mnemonic, and more details about your key in the output:

bash

address:

celestia127fpaygehlsgjdknwvlr2mux7h5uvhkxktgkc5 name:

vesting-key pubkey:

'{"@type":"/cosmos.crypto.secp256k1.PubKey","key":"A5JF/we+s5gFt6g944XbKVVYgQB9OY+U/l5dhZjLDczO"}' type : local

** Important ** write this mnemonic phrase in a safe place. It

is

the

only

way

to

recover

your

account

if

you

ever

forget

your

password.

index

enter

egg

broken

ostrich

duty

bitter

blind

all

car

hollow

coral

youth

early

verify

point

void

anger

daring

sausage

decline

net

shove

oil

address:

celestia127fpaygehlsgjdknwvlr2mux7h5uvhkxktgkc5 name:

vesting-key pubkey:

'{"@type":"/cosmos.crypto.secp256k1.PubKey","key":"A5JF/we+s5gFt6g944XbKVVYgQB9OY+U/l5dhZjLDczO"}' type : local

** Important ** write this mnemonic phrase in a safe place. It

is

the

only

way

to

recover

your

account

if

you

ever

forget

your

password.

index

enter

egg

broken

ostrich

duty

bitter

blind

all

car

hollow

coral

youth

early

verify

point

void

anger

daring

sausage

decline

net

shove

oil

List your keys

bash ./celestia-appd

keys

list

--home CEIESTIA_APP_HOME ./celestia-appd

keys

list

--home CEIESTIA_APP_HOME Output:

bash

address:

celestia1adgkqcmzuxvg7x5avx8a8rjwpmxgzex3ztef6j name:

validator pubkey:

'{"@type":"/cosmos.crypto.secp256k1.PubKey","key":"Ahzu6yr9XMPIxLquhgBhj9xL3wIaOz6PE3CvML/oPQym"}' type : local
-

address:

celestia127fpaygehlsgjdknwvlr2mux7h5uvhkxktgkc5 name:

vesting-key pubkey:

'{"@type":"/cosmos.crypto.secp256k1.PubKey","key":"A5JF/we+s5gFt6g944XbKVVYgQB9OY+U/l5dhZjLDczO"}' type : local
-

address:

celestia1adgkqcmzuxvg7x5avx8a8rjwpmxgzex3ztef6j name:

validator pubkey:

'{"@type":"/cosmos.crypto.secp256k1.PubKey","key":"Ahzu6yr9XMPIxLquhgBhj9xL3wIaOz6PE3CvML/oPQym"}' type : local
-

address:

celestia127fpaygehlsgjdknwvlr2mux7h5uvhkxktgkc5 name:

vesting-key pubkey:

'{"@type":"/cosmos.crypto.secp256k1.PubKey","key":"A5JF/we+s5gFt6g944XbKVVYgQB9OY+U/l5dhZjLDczO"}' type : local

Set variables

Set the keys as variables, using the validator address as theFROM_ADDRESS and the vesting-key as theTO_ADDRESS .

bash export FROM_ADDRESS = celestia1adgkqcmzuxvg7x5avx8a8rjwpmxgzex3ztef6j export TO_ADDRESS =
celestia127fpaygehlsgjdknwvlr2mux7h5uvhkxktgkc5 export FROM_ADDRESS =
celestia1adgkqcmzuxvg7x5avx8a8rjwpmxgzex3ztef6j export TO_ADDRESS =
celestia127fpaygehlsgjdknwvlr2mux7h5uvhkxktgkc5

Create your devnet vesting account

Create the vesting account with the following command:

note

The remainder of the instructions are for acontinuous vesting account, if you'd like to make a delayed vesting account, use the--delayed flag.

For example, the command to create a delayed vesting account would look like:

bash ./celestia-appd

tx

vesting

create-vesting-account TO_ADDRESS 100000 utia

1686748051

--from FROM_ADDRESS --gas

auto

--fees

100000 utia

--chain-id

private

--home CEIESTIA_APP_HOME --delayed ./celestia-appd

tx

vesting

create-vesting-account TO_ADDRESS 100000 utia

1686748051

--from FROM_ADDRESS --gas

auto

--fees

100000 utia

--chain-id

private

--home CEIESTIA_APP_HOME --delayed bash ./celestia-appd

tx

vesting

create-vesting-account TO_ADDRESS 100000 utia

1686748051

--from FROM_ADDRESS --gas

auto

--fees

100000 utia

--chain-id

private

--home CEIESTIA_APP_HOME ./celestia-appd

tx

vesting

create-vesting-account TO_ADDRESS 100000 utia

1686748051

--from FROM_ADDRESS --gas

auto

--fees

100000 utia

--chain-id

private

--home CEIESTIA_APP_HOME Select "Y" to choose "yes".

Optional

If you'd like to run the command with the-y flag, it will execute the transaction without needing to provide the "y" answer as above.

bash ./celestia-appd

tx

vesting

create-vesting-account TO_ADDRESS 100000 utia

1686748051

--from FROM_ADDRESS --gas

auto

--fees

100000 utia

--chain-id

private

--home CEIESTIA_APP_HOME -y ./celestia-appd

tx

vesting

create-vesting-account TO_ADDRESS 100000 utia

1686748051

--from FROM_ADDRESS --gas

auto

--fees

100000 utia

--chain-id

private

--home CEIESTIA_APP_HOME -y Output:

bash gas

estimate:

96112 auth_info: fee: amount: -

amount:

"100000" denom:

utia gas_limit:

"96112" granter:

"" payer:

"" signer_infos: [] tip:

null body: extension_options: [] memo:

"" messages: -

'@type':

/cosmos.vesting.v1beta1.MsgCreateVestingAccount amount: -

amount:

"100000" denom:

utia delayed:

false end_time:

"1686748051" from_address:

celestia1adgkqcmzuxvg7x5avx8a8rjwpmxgzex3ztef6j to_address:

celestia127fpaygehlsgjdknwvlr2mux7h5uvhkxktgkc5 non_critical_extension_options: [] timeout_height:

"0" signatures: [] confirm

transaction

before

signing

and

broadcasting

code:

0 codespace:

"" data:

"" events: [] gas_used:

"0" gas_wanted:

"0" height:

"0" info:

"" logs: [] raw_log:

'[]' timestamp:

"" tx:

null txhash:

6093 DF76DBA90F04FF63D197FC1569F04ED3DBE64081A0BBA9BAD4E69AA570D2 gas

estimate:

96112 auth_info: fee: amount: -

amount:

"100000" denom:

utia gas_limit:

"96112" granter:

"" payer:

"" signer_infos: [] tip:

null body: extension_options: [] memo:

"" messages: -

'@type':

/cosmos.vesting.v1beta1.MsgCreateVestingAccount amount: -

amount:

"100000" denom:

utia delayed:

false end_time:

"1686748051" from_address:

celestia1adgkqcmzuxvg7x5avx8a8rjwpmxgzex3ztef6j to_address:

celestia127fpaygehlsgjdknwvlr2mux7h5uvhkxktgkc5 non_critical_extension_options: [] timeout_height:

"0" signatures: [] confirm

transaction

before

signing

and

broadcasting

code:

0 codespace:

"" data:

"" events: [] gas_used:

"0" gas_wanted:

"0" height:

"0" info:

"" logs: [] raw_log:

'[]' timestamp:

"" tx:

null txhash:

6093 DF76DBA90F04FF63D197FC1569F04ED3DBE64081A0BBA9BAD4E69AA570D2 The timestamp for the previous command is in the past, so once you create the vesting account, the tokens will vest. You can check your account balances to verify this.

Query the devnet vesting account details

Check that the account has been created and works as expected by querying theTO_ADDRESS account details:

bash ./celestia-appd

query

account TO_ADDRESS --home CEIESTIA_APP_HOME ./celestia-appd

query

account TO_ADDRESS --home CEIESTIA_APP_HOME In the output, you will notice that the account type is aContinuousVestingAccount :

bash '@type' :

/cosmos.vesting.v1beta1.ContinuousVestingAccount base_vesting_account: base_account: account_number:

"7" address:

celestia127fpaygehlsgjdknwvlr2mux7h5uvhkxktgkc5 pub_key:

null sequence:

"0" delegated_free: [] delegated_vesting: [] end_time:

"1686748051" original_vesting: -

amount:

"100000" denom:

utia start_time:

"1687908352" '@type' :

/cosmos.vesting.v1beta1.ContinuousVestingAccount base_vesting_account: base_account: account_number:

"7" address:

celestia127fpaygehlsgjdknwvlr2mux7h5uvhkxktgkc5 pub_key:

null sequence:

"0" delegated_free: [] delegated_vesting: [] end_time:

"1686748051" original_vesting: -

amount:

"100000" denom:

utia start_time:

"1687908352"

Query the devnet base account details

Check theFROM_ADDRESS account details:

bash ./celestia-appd

query

account FROM_ADDRESS --home CEIESTIA_APP_HOME ./celestia-appd

query

account FROM_ADDRESS --home CEIESTIA_APP_HOME In the output, you will notice the account type isBaseAccount :

bash '@type' :

/cosmos.auth.v1beta1.BaseAccount account_number:

"0" address:

celestia1adgkqcmzuxvg7x5avx8a8rjwpmxgzex3ztef6j pub_key: '@type' :

/cosmos.crypto.secp256k1.PubKey key:

Ahzu6yr9XMPIxLquhgBhj9xL3wIaOz6PE3CvML/oPQym sequence:

"2" '@type' :

/cosmos.auth.v1beta1.BaseAccount account_number:

"0" address:

celestia1adgkqcmzuxvg7x5avx8a8rjwpmxgzex3ztef6j pub_key: '@type' :

/cosmos.crypto.secp256k1.PubKey key:

Ahzu6yr9XMPIxLquhgBhj9xL3wIaOz6PE3CvML/oPQym sequence:

"2"

Query the balances of the devnet accounts

Next, we can check the balance of the accounts:

bash ./celestia-appd

query

bank

balances TO_ADDRESS --home CEIESTIA_APP_HOME ./celestia-appd

query

bank

balances TO_ADDRESS --home CEIESTIA_APP_HOME Output will show you the balance of the vesting account:

bash balances: -

amount:

"100000" denom:

utia pagination: next_key:

null total:

"0" balances: -

amount:

"100000" denom:

utia pagination: next_key:

null total:

"0" bash ./celestia-appd

query

bank

balances FROM_ADDRESS --home CEIESTIA_APP_HOME ./celestia-appd

query

bank

balances FROM_ADDRESS --home CEIESTIA_APP_HOME The output will show the remaining balance of the validator:

bash balances: -

amount:

"999994999800000" denom:

utia pagination: next_key:

null total:

"0" balances: -

amount:

"999994999800000" denom:

utia pagination: next_key:

null total:

"0" Congratulations! You've now made your own vesting account on a local devnet. Next, you can learn how to create a vesting account on Mocha testnet.

Mocha

In the previous section of this tutorial, we learned how to create a vesting account on a local devnet. In this portion of the tutorial, we'll cover how to set up a full consensus node and set up a vesting account onMocha testnet .

First, be sure that you haveinstalled celestia-app for the latest version for Mocha testnet .

Create a wallet

Set the keyring backend, so you don't need to use the flag for every command:

bash celestia-appd

config

keyring-backend

test celestia-appd

config

keyring-backend

test Add a new key for a full node and one for a vesting account:

bash celestia-appd

keys

add

origin && celestia-appd

keys

add

vesting celestia-appd

keys

add

origin && celestia-appd

keys

add

vesting List the keys:

bash celestia-appd

keys

list celestia-appd

keys

list Set your keys as variables:

bash export FROM_ADDRESS = address_of_origin_account export TO_ADDRESS = address_of_vesting_account export FROM_ADDRESS = address_of_origin_account export TO_ADDRESS = address_of_vesting_account

Fund your account

Head tothe faucet , and fund yourorigin address.

Create a vesting account on Mocha

To create a vesting account on Mocha, you will need an RPC URL to send the transaction to. You can find the RPC endpoints on the Mocha testnet page .

Set your RPC URL:

bash export RPC_URL = https://rpc-mocha.pops.one:443 export RPC_URL = https://rpc-mocha.pops.one:443 We will use a few flags in our vesting command that are different than the devnet version. Since we aren't using our own validator or full node, we will use an RPC URL.

We also need to declare the chain ID asmocha .

View the help menu for vesting to understand these flags more:

bash celestia-appd

tx

vesting

--help celestia-appd

tx

vesting

--help Here's an example command to set up the vesting account:

bash celestia-appd

tx

vesting

create-vesting-account TO_ADDRESS 100000 utia

1686748051

--from FROM_ADDRESS --gas

100000

--fees

100000 utia

--node RPC_URL --chain-id

mocha

--delayed celestia-appd

tx

vesting

create-vesting-account TO_ADDRESS 100000 utia

1686748051

--from FROM_ADDRESS --gas

100000

--fees

100000 utia

--node RPC_URL --chain-id

mocha

--delayed

Optional: Set up a full consensus node or validator

Running a full consensus node or validator will prevent you from needing to use an RPC.

You canset up a validator or full consensus nodefor the previous portion of the tutorial.

Note: this may take some time depending on how you choose to sync the state of the chain.

Optional: Change your client.toml

If you edit your client configuration inclient.toml , you can set both the chain ID and the node RPC URL. This will prevent you from needing to run each flag for every command line that you use.

toml

This is a TOML config file.

For more information, see https://github.com/toml-lang/toml

Client Configuration

The network chain ID

chain-id = "mocha"

The keyring's backend, where the keys are stored (os|file|kwallet|pass|test|memory)

keyring-backend = "test"

CLI output format (text|json)

output = "text"

: to Tendermint RPC interface for this chain

node = "tcp://rpc-mocha.pops.one:443"

Transaction broadcasting mode (sync|async|block)

broadcast-mode = "sync"

This is a TOML config file.

For more information, see https://github.com/toml-lang/toml

Client Configuration

The network chain ID

chain-id = "mocha"

The keyring's backend, where the keys are stored

(os|file|kwallet|pass|test|memory)

keyring-backend = "test"

CLI output format (text|json)

output = "text"

: to Tendermint RPC interface for this chain

node = "tcp://rpc-mocha.pops.one:443"

Transaction broadcasting mode (sync|async|block)

broadcast-mode = "sync"

Notes

Not all vesting accounts can be created with a message, some need to be set at genesis. You can learn more in the Cosmos Network documentation .

Conclusion

Congratulations! You've learned how to create a local devnet, create a vesting account on it, and how to make a vesting account on the Mocha testnet! [][Edit this page on GitHub] Last updated: Previous page Multisig Next page SystemD []