Governance actions

Submitting the proposal

While in the same directory as your finalproposal.json file, you can submit the proposal with:

For a default proposal:

namada

client

init-proposal

--data-path

proposal.json For non-default proposals:

One of the flags--pgf-stewards ,--pgf-funding ,--eth must be specified. For example, for a PGF steward proposal:

namada

client

init-proposal

--pgf-stewards

--data-path

proposal.json

Query the proposal

If the submitted transaction was accepted, the user can query the proposal with its proposal ID:

namada

client

query-proposal

--proposal-id ID Additionally, the user can query some of the most recent proposals with:

namada

client

query-proposal

Vote on a proposal

Only delegators and validators can vote on proposals. A delegator or validator can send a vote with the following command:

namada

client

vote-proposal \ --proposal-id ID \ --vote

yay \ --address YOUR_ADDRESS where--vote can be eitheryay ,nay orabstain .

The--address flag needs to be the address of the delegator or validator that is voting. You may also use the--gas-payer flag to specify the address of the account that will pay for the gas.

Check the result

As soon as the ledger reaches the epoch defined in the json asvoting_end_epoch , no more votes will be accepted. At the beginning of theactivation_epoch , the votes are tallied. If the proposal passes, the code defined inproposal_code json field

will be executed at this time.

You can use the following commands to check the status of a proposal:

namada

client

query-proposal

--proposal-id ID or to just check the result:

namada

client

query-proposal-result

--proposal-id ID If the proposal has not passed, it will be rejected, and the code will not be executed.

Another important note is that the voting period differs between validators and non-validators. Validators have a shorter voting period than delegators. This ensures that, if a delegator would like to vote differently than the validator to which they are delegated, there is a period during which they can change their vote to their own, rather than let the validator vote using their voting power. This prevents validators from voting in the last block of the voting period against the true preferences of their delegators. See the specs for more information.

The output ofnamadac query-proposal-result will detail the latest epoch in which validators can vote on a given proposal.

Tally types

The governance mechanism defines different criteria depending on proposal type in order for the proposal to pass:

Default proposals require at least 40% of the total active voting power to have voted AND theyay voting power must be at least 2/3 of the combinedyay + nay voting power.

PGF steward proposals require at least 1/3 of the total active voting power to have voted AND theyay voting power must be larger than thenay voting power.

Funding proposals are tallied differently depending on if the proposal author is a PGF steward:

- Non-steward: the funding proposal is tallied in the same way as the PGF steward proposal (above).
- Steward: the proposal can pass without any votes at all. However, the network can veto the proposal if 1/3 of the total active voting power votes, and thenay
- votes are larger than theyay
- votes.
- In other words, the proposal passes if less than 1/3 of the total voting power votes OR there are moreyay
- votes thannay
- votes.

Submit a governance proposal with wasm code attached

First you will need a valid.wasm file. You then need to read this file into a vector of bytes. This can be done with the following small python script:

with

open (wasm_file_path, "rb")

as f : byte_vec =

list (f. read ()) print (str (byte_vec)) The output can then be copied into thedata field of the proposal json. E.g"data": [1,255,3,4,5,182,7,81,90,10] .

Additionally, there is a script in the namada repo calledadd_proposal_wasm_code.py (opens in a new tab) that can be used to add the wasm data to a proposal template liketemplate_proposal.json (LINK TO THIS TOO!):

python3

add_proposal_wasm_code.py

--proposal-path JSON_FILE --wasm-path WASM_FILE This command will add the wasm data to theJSON_FILE .

When submitting this proposal, it is likely that the gas requirement will be large. Therefore, it is recommended to supply at least the--gas-limit flag.

namadac

init-proposal

--data-path

proposal.json

--gas-limit

500000 Hint: use the--dry-run feature to figure out how much gas will be needed and usenamadac query-protocol-paramters to see the current minimum gas price.

A video tutorial

Skip all the boring text and watch a video tutorial on how to submit a proposal:

Types of proposals Public Goods Funding (PGF)