On-chain proposals

Types of proposals

All the different types of proposals are listed in the specs(opens in a new tab) . Different proposals will have different permissions, data structures, as well as voting requirements.

Create a proposal

Assuming you have an account with at least 500 NAM token (in this example we are going to usemy-new-acc), lets get the corresponding address

namada

wallet

find

--alias

my-new-acc Now, we need to create a json fileproposal.json holding the content of our proposal. Copy the below text into a json file.

{ "proposal" : { "id" :

"Arbitrary" , "content" : { "title" :

"Text" , "authors" :

"Text" , "discussions-to" :

"URL" , "created" :

"YYYY--DDTHH:MM:SSZ" , "license" :

"" , "abstract" :

"Text" , "motivation" :

"Text" , "details" :

"Text" , "requires" :

"Number" } , "author" :

"author-address" , "voting_start_epoch" :

4 , "voting_end_epoch" :

6 , "grace_epoch" :

8 } , "data" :

"TODO-ADD-DATA-IF-NEEDED" } A correct Steward proposal example is shown below:

{ "proposal" : { "id" :

1 , "content" : { "title" :

"One Small Step for Namada, One Giant Leap for Memekind" , "authors" :

" " , "discussions-to" :

"forum.namada.net/t/namada-proposal/1" , "created" :

"2069-04-20T00:04:44Z" , "license" :

"MIT" , "abstract" :

"We present a proposal that will send our community to the moon. This proposal outlines all training necessary to accomplish this goal. All members are welcome to join." , "motivation" :

"When you think about it, the moon isn't actually that far away.The moon is only 384,400 km. We have not yet brought Namada to the moon, so it is only natural to use 101 as the prime number for our modular arithmetic operations. 384,400 (mod 101) = 95. 95 km is a distance that can be easily covered by a single person in a single day. Namada was produced by more than 100 people. So 95/100 = 0, rounded to the nearest integer. This means that Namada can reach the moon in no time." , "details" :

"Bringing Namada to the moon in no time is easily achievable. We just need to pass this governance proposal and set the plan in action" , "requires" :

"" } , "author" :

"tnam1qql6qme020vw7pfw4ruavrwxujf9lf8z75v90c8q" , "voting_start_epoch" :

21 , "voting_end_epoch" :

24 , "grace_epoch" :

27 } , "data" : { "add" :

"tnam1qql6qme020vw7pfw4ruavrwxujf9lf8z75v90c8q" , "remove" : ["tnam1q9jpj9u5p6ugjylarkwmj9tsxr0pjlgpn5wa8ff0"] } } In the content field, most of the fields are self-explanatory. Therequires field references a proposal id that must be passed before this proposal can be executed. Thecreated field must be in the formatYYYY-MM-DDTHH:MM:SSZ .

You should change the value of:

- Author
- field with the address ofmy-new-acc
- ;
- voting_start_epoch
- with a future epoch (must be a multiple of themin-voting-period
- found in theparameters.toml
-) for which you want the voting to begin;
- voting_end_epoch
- with an epoch greater thanvoting_start_epoch
- , a multiple ofmin-voting-period
- , and by which no further votes will be accepted;
- grace_epoch
- with an epoch greater thanvoting_end_epoch
-
    - `, in which the proposal, if passed, will come into effect.

Thedata field and its structure is dependent on the type of proposal being submitted. Below we outline the structure of the "data" field for each type of proposal. The one given in the example above is for aDefault Proposal .

Default Proposal

"data" : [1 , 2 , 3 , 4 , 5 , 6 , 7 , 8 , 9 , 10] # This is a u 8 vector The data field for default proposals is optional. This is in line with the nature of default proposals. If the proposals have code attached to them in order to change governance parameters, then this code will be represented as a wasm file and byte encoded vector will be tehe value of the data field. In order to read the path to bytes, one can use a simple python script.

with

open (file_path, "rb")

as f : byte_vec = f . read () print (byte_vec)

Steward Proposal

"data" : [{ "add" :

"tnam1qql6qme020vw7pfw4ruavrwxujf9lf8z75v90c8q" }] ð¡ The data field for steward proposals is alist of actions to be taken. The actions can be eitheradd orremove and the address is the address of the steward to be added or removed. In this way you can add or remove multiple stewards in a single proposal.

PGF Proposal

"data" : { "continuous"

: [{ "Internal" : {

"amount" :

"1337" , "target" :

"tnam1q9rxk49a7y6fp8h5qwz47d6jhml0km2w2y7emqyz"

} }] , "retro"

: [{ "Internal" : {

"amount" :

"1337" , "target" :

"tnam1q9rxk49a7y6fp8h5qwz47d6jhml0km2w2y7emqyz" } }] }, ð¡ The data field for PGF proposals contains both continuous and retroactive PGF funding actions. Within each action, the user can include multiple payments in the form of a vector. Within each payment, the target field contains the address of the recipient as well as the amount of NAM that they will receive. For continuous PGF funding, the amount specified will be sent at the end of each epoch. There is also the option to remove a recipient from the continuous PGF funding, by specifying an already existing continuous funding payment, and then setting the amount to "0". For retroactive PGF funding, the amount specified will be sent immediately.

IBC PGF proposals

For a pgf proposal that is to be executed on an IBC chain, the "Internal" data structure will be substituted with "IBC":

"IBC" : {

"amount" :

"1337" , "target" :

"tnam1q9rxk49a7y6fp8h5qwz47d6jhml0km2w2y7emqyz" , "port_id" :

"" , "channel_id" :

"" }

ETH Bridge Proposal

"data" : "" â ï¸ Note : The encoding will be submitted as a string

Submitting the proposal

As soon as yourproposal.json file is ready, you can submit the proposal with (make sure to be in the same directory as theproposal.json file):

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 all the proposals with:

namada

client

query-proposal or a single proposal with

namada

client

query-proposal

--proposal-id

0 where0 is the proposal id.

Vote on a proposal

Only delegators and delegates can vote on proposals. Assuming you fall into one of these categories, you can send a vote with the following command:

namada

client

vote-proposal \ --proposal-id

0 \ --vote

yay \ --address

< your-addres s

> The--address flag needs to be the address of the delegator or delegate that is voting. For validators, it is better to use the raw address rather than the alias. The--signing-keys can be the alias, however it is recommended to leave out the--signing-keys flag and let the client figure out the signing keys. If the key does not have a balance, use the--gas-payer flag to specify the address of the account that will pay for the gas. where--vote can be eitheryay ,nay orabstain .

Check the result

As soon as the ledger reaches the epoch defined in the json asvoting_end_epoch , no more votes will be accepted. The code defined inproposal_code json field will be executed at the beginning ofgrace_epoch epoch. You can use the following commands to check the status of a proposal:

namada

client

query-proposal

--proposal-id

0 or to just check the result:

namada

client

query-proposal-result

--proposal-id

0 It is important to note that the proposal will only be executed at the start of thegrace_epoch epoch if it has passed. If it has not passed, the proposal will be rejected and the code will not be executed.

Another important note is that the voting period differs between validators and non-validators. The validators have a shorter

voting period than the delegators. This is defined in theparameters.toml . This is to ensure that the non-validators have enough time to vote on the proposals (so that the validators cannot vote in the last block against the non-validators preference). See the specs for more information.

Submit a governance proposal with wasm code attached (advanced)

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 python script:

with

open (wasm_file_path, "rb")

as f : byte_vec =

list (f. read ()) print (str (byte_vec)) This needs to go into thedata field of the proposal json. E.g"data": [1,2,3,4,5,6,7,8,9,10]

When submitting this proposal, it is likely that the gas requirement will be large. Therefore, it is recommended to supply both the--gas-limit and--gas-price flags.

namadac

init-proposal

--data-path

proposal.json

--gas-limit

500000

--gas-price

0.01 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:

Governance and Public Goods Funding Off-chain proposals