State

Each account has an associated state where it stores itsmetadata and all the contract-related data(contract's code + storage) .

Accounts' states can beread by anyone in the network, but only the account itself can change it.

Each accountpays for their own storage by locking a part of their balance proportional to the space used.

Account's Metadata

The state keeps track of relevant metadata from the contract. Particularly, the state stores the following fields:

1. amount
2. : The accounts balance expressed in yoctoNEAR (1 Ⓝ = 1024
3. yⓃ).
4. code_hash
5. : A hash of the contract's Wasm file, filled with1s
6. if no contract is present.
7. storage_usage
8. : Amount of bytes used for storage by the account (code + metadata + data storage).

info You can check an accounts metadata by running the followingnear cli command:

near state hello-nearverse.testnet

Contract's State

The state is also the place where both thecontract's code and thecontract's storage are stored.

The contract's storage is organized askey-value pairs encoded using base64 and JSON serialization (oBorsh in Rust).

info You can check an accounts contract state by running the followingnear cli command:

near view-state hello-nearverse.testnet --finality final --utf8 true tip When developing contracts our SDK will handle serializing the storage, so you can focus on what matters.

Paying for Storage (1 Ⓝ ~ 100kb)

In order to pay for storage, accounts needs to lock a portion of their balance proportional to the amount of data being stored. This means that:

- If more data is added and thestate increases ↑
- , the account'sbalance decreases ↓
- .
- If data is deleted and thestate decreases ↓
- , the account'sbalance increases ↑
- .

Currently, it cost approximately1 Ⓝ to store100kb of data.Edit this page Last updatedonDec 2, 2022 byAnton Puhach Was this page helpful? Yes No

Previous Smart Contract Next Creating Accounts