NEAR CLI - Basics

After your contract is ready you can deploy it in the NEAR network for everyone to use it.

Let us guide you on how to use theNEAR CLI to deploy your contract and call its methods.

info On this page, we will only cover the basics of NEAR CLI. For more information visit theNEAR CLI documentation page .

Deploying the Contract

Thanks to theNEAR CLI deploying a contract is as simple as:

1. Compiling the contract to wasm (done automatically throughyarn build
2. in our templates).
3. Create an account
4. anddeploy the contract
5. into it usingNEAR CLI
6. .

Create an Account and Deploy

- near-cli
- near-cli-rs

Create a new account pre-funded by a faucet & deploy

near create-account --useFaucet near deploy

Get the account name

cat ./neardev/dev-account

Automatically deploy the wasm in a new account

near account create-account sponsor-by-faucet-service .testnet autogenerate-new-keypair save-to-keychain network-config testnet create

near contract deploy .testnet use-file without-init-call network-config testnet sign-with-keychain

Deploy in an Existing Account

- near-cli
- near-cli-rs

login into your account

near login

deploy the contract

near deploy

login into your account

near account import-account using-web-wallet network-config testnet

deploy the contract

near contract deploy use-file without-init-call network-config testnet sign-with-keychain send tip You can overwrite a contract

by deploying another on top of it. In this case, the account's logic will change, but the state will persist info By defaultnear-cli uses thetestnet network. DefineNEAR_ENV=mainnet to deploy intomainnet . Naming Convention for Public-Facing Methods Once the contract is deployed to the network, anyone and any other contract (i.e., any other account on NEAR) can interact with it by calling its methods. Furthermore, any transactions involving the contract will also be included in the network's data stream, which means its activity can also be visible to any who listens to particular events.

Considering this, we advise to name methods usingsnake_case in all SDKs as this is compatible with the remainder of the NEAR ecosystem which is predominantly comprised of Rust contracts.

Initializing the Contract

If your contract has aninitialization method you can call it to initialize the state. This is not necessary if your contract implementsdefault values for the state.

- near-cli
- near-cli-rs

Call the initialization method (init in our examples)

near call [] --accountId

Call the initialization method (init in our examples)

near contract call-function as-transaction json-args [] prepaid-gas '30 TeraGas' attached-deposit '0 NEAR' sign-as network-config testnet sign-with-keychain send info You can initialize your contractduring deployment using the--initFunction &--initArgs arguments.

Calling the Contract

Once your contract is deployed you can interact with it right away usingNEAR CLI .

View methods

View methods are those that performread-only operations. Calling these methods is free, and do not require to specify which account is being used to make the call:

- near-cli
- near-cli-rs

near view near contract call-function as-read-only text-args '' network-config testnet now tip View methods have by default 200 TGAS for execution

Change methods

Change methods are those that perform both read and write operations. For these methods we do need to specify the account being used to make the call, since that account will expend GAS in the call.

- near-cli
- near-cli-rs

near call --accountId [--deposit] [--gas] near contract call-function as-transaction json-args prepaid-gas attached-deposit sign-as network-config testnet sign-with-keychain send Edit this page Last updatedonFeb 9, 2024 bygagdiez Was this page helpful? Yes No

Previous Local Development Next Updating Contracts