JS SDK CLI

The SDKCommand Line Interface (CLI) is a tool that enables to act on different parts of the build process as well as generate validations and anABI . Among other things, the SDK CLI enables you to:

- Control the different parts of the build process
- Validate your contract and TypeScript code
- Create an ABI JSON file

Overview

Click on a command for more information and examples.

Commands

Command Description near-sdk-js build Build a NEAR JS Smart-contract near-sdk-js validateContract Validate a NEAR JS Smart-contract near-sdk-js checkTypescript Run TSC with some CLI flagsnear-sdk-js createJsFileWithRollup Create an intermediate JavaScript file for later processing with QJSC near-sdk-js transpileJsAndBuildWasm Transpiles the target javascript file into .c and .h using QJSC then compiles that into wasm using clang

Setup

Installation

Make sure you have a current version ofnpm andNodeJS installed.

Mac and Linux

1. Installnpm
2. andnode
3. using apackage manager
4. likenvm
5. as sometimes there are issues using Ledger due to how macOS handles node packages related to USB devices.
6. Ensure you have installed Node version 12 or above.
7. Installnear-cli
8. globally by running:

npm install -g near-cli

Windows

For Windows users, we recommend using Windows Subsystem for Linux (WSL). 1. InstallWSL 2[click here] 3. Installnpm 4. [click here] 5. InstallNode.js 6.[click here]7. Changenpm 8. default directory[click here]9. * This is to avoid any permission issues withWSL 10. OpenWSL 11. and installnear-cli 12. globally by running:

npm install -g near-cli heads up Copy/pasting can be a bit odd usingWSL .

- "Quick Edit Mode" will allow right-click pasting.
- Depending on your version there may be another checkbox allowingCtrl
- +V
- pasting as well.

Commands

near-sdk-js build

Build a NEAR JS Smart-contract, specifying the source, target,package.json , andtsconfig.json files. If none are specified, the default values are used. The argument default values are:

- source:src/index.js
- target:build/contract.wasm
- packageJson:package.json
- tsConfig:tsconfig.json

Options default values are set tofalse .

- arguments (optional):[source] [target] [packageJson] [tsConfig]
- options:--verbose --generateABI

Example:

near-sdk-js build src/main.ts out/main.wasm package.json tsconfig.json --verbose true --generateABI true

near-sdk-js validateContract

Validate a NEAR JS Smart-contract. Validates the contract by checking that all parameters are initialized in the constructor. Works only for TypeScript.

- arguments:[source]
- options:--verbose

Example:

near-sdk-js validateContract src/main.ts --verbose true Example Response:

npx near-sdk-js validateContract src/index.ts [validate] › … awaiting Validating src/index.ts contract...

near-sdk-js checkTypescript

Run TSC with some CLI flags.

warning This command ignorestsconfig.json . * arguments:[source] * options:--verbose

Example:

near-sdk-js checkTypescript src/main.ts --verbose true Example Response:

npx near-sdk-js checkTypescript src/index.ts [checkTypescript] › … awaiting Typechecking src/index.ts with tsc...

near-sdk-js createJsFileWithRollup

Create an intermediate JavaScript file for later processing with QJSC.

- arguments:[source]
- [target]
- options:--verbose

Example:

near-sdk-js createJsFileWithRollup src/main.ts out/main.js --verbose true Example Response:

npx near-sdk-js createJsFileWithRollup src/index.ts [createJsFileWithRollup] › … awaiting Creating src/index.ts file with Rollup...

near-sdk-js transpileJsAndBuildWasm

Create an intermediate JavaScript file for later processing with QJSC.

- arguments:[source]
- [target]
- options:--verbose

Example:

near-sdk-js transpileJsAndBuildWasm src/main.js out/main.wasm --verbose true Example Response:

npx near-sdk-js transpileJsAndBuildWasm [transpileJsAndBuildWasm] › ✔ success Generated build/contract.wasm contract successfully! Edit this page Last updatedonJan 30, 2023 byDennis Was this page helpful? Yes No