

# JS SDK CLI

The SDK [Command Line Interface](#) (CLI) is a tool that enables to act on different parts of the build process as well as generate validations and an [ABI](#) . 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 flags [near-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 of npm and NodeJS installed.

#### Mac and Linux

1. Install npm
2. and node
3. using a [package manager](#)
4. like nvm
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. Install near-cli
8. globally by running:

```
npm install -g near-cli
```

#### Windows

For Windows users, we recommend using Windows Subsystem for Linux (WSL ). 1. Install WSL 2. [click here](#) 3. Install npm 4. [click here](#) 5. Install Node.js 6. [click here](#) 7. Change npm 8. default directory [click here](#) 9. \* This is to avoid any permission issues with WSL 10. Open WSL 11. and install near-cli 12. globally by running:

```
npm install -g near-cli
```

 heads up Copy/pasting can be a bit odd using WSL .

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

## Commands

### near-sdk-js build

Build a NEAR JS Smart-contract, specifying the source, target, package.json , and tsconfig.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 to false .

- 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:
```

```
npm run 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 ignores tsconfig.json . \* arguments: [source] \* options: --verbose

Example:

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

```
npm run 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:
```

```
npm run 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:
```

```
npm run near-sdk-js transpileJsAndBuildWasm [transpileJsAndBuildWasm] › ✓ success Generated build/contract.wasm contract successfully! Edit this page Last updated on Jan 30, 2023 by Dennis Was this page helpful? Yes No
```

