Fadroma - SC framework

Fadroma (built byhack.bg) introduces core workflows for smart contract development and deployment on Cosmos-based platforms, specifically Secret Network! Fadroma helps developers avoid platform specific boilerplate code - automating the repetitive workflow steps by providing essential smart contract building blocks that empower developers to both maintain high development velocity and deliver a reliable product.

Fadroma is currently being developed as an in-house framework facilitating theSienna Network project. In order to bring Fadroma to a larger developer audience, development hours are being dedicated to improving test coverage, documentation, stabilizing APIs, and ensuring compatibility with a wide range of use cases and underlying platforms.

Get started with Fadroma here:https://github.com/hackbg/fadroma/blob/stable/guide/basic-project-setup.md

Fadroma Features

The following is the list of some of the important initial feature of Fadroma - creating a unique and fluid development experience.

Fadroma Prelude - reexport of platform libraries, with added types for frequently used building blocks:

- ContractLink
- , a data type that contains a contract address and code hash
- Humanize
- andCanonize
-
    - traits to convert structures that containHumanAddr
- to theirCanonicalAddr
- equivalent and vice versa
- ViewingKey
- andPermit
- viewing key and permit with custom permissions
- Uint256
-
    - 256-bit equivalents of the CosmWasm numeric types, etc.
-

Fadroma Derive : a derive macro that generates boilerplate code like message definitions and dispatch init/handle/query functions, as well as wasm32 entry points - empowering contract creation out of separate subsystems.

Fadroma Components - implementations of frequently used smart contract functionality:

- Fadroma Admin: TX sender-based admin auth
- Fadroma VK: viewing key-based auth
- Fadroma Permit: query permit-based auth
- Fadroma Killswitch: pause or permanently disable contracts
- Fadroma Token: A SNIP-20 implementation forked from thereference implementation
- and rebuilt as a trait.
- This allows individual token contract functions to be overriden without having to copy the entire codebase.
- Furthermore, it enables token functionality to be added to service contracts, obviating the need for a separate contract to represent the service token.
-

Fadroma Ensemble: fast integration testing of multiple smart contracts in Rust by mocking the CosmWasm API - useful for testing application logic when it is distributed among several contracts.and having them run instantaneously during development.

Fadroma Ops : Node.js-based operations framework for managing the smart contract lifecycle. Provides a rich vocabulary of classes for modeling your smart contract deployment workflow.

- Fadroma Build: compile smart contracts from the working tree or a past point in Git history.
- Fadroma Localnet: run a localnet for development,
- Fadroma Receipts: keep track of uploads and instantiations. It is possible to update an existing deployment by specifying its id from the receipts and adding additional instances or executing messages all from code.
- Fadroma Schema: generate TypeScript definitions from the JSON Schema exported by contracts.
- Fadroma Bundle: Execute multiple messages in a single transaction or generate an unsigned transaction for manual multi-signing.
- 100% test coverage
- Docker Compose integration for a portable development environment based on familiar tools.
-

Fadroma CLI: Command-line entrypoint for creating new projects and running default and custom management commands.

- (init) Start a project
- (add) Add contracts to it
- (compile) Compile them
-

Last updated3 months ago On this page Was this helpful?Edit on GitHub Export as PDF