What is a Smart Contract?

Smart contracts are pieces ofexecutable code that live in a NEAR account. They canstore data ,perform transactions in the account's name, andexpose methods so other accounts can interact with them.

Do not worry if you don't know how smart-contract blockchains work. As a developer, it is sufficient to understand that NEAR smart-contracts:

1. Havelimited
2. computational resources.
3. Interact with other contracts in anasynchronous
4. way.
5. Deal withreal money
6. , for which security must be a top concern.

HTTP Requests and Smart Contracts Smart contractscannot perform HTTP requests , meaning they can't fetch data from outside the NEAR network.

Where do contracts live?

Smart Contracts are deployed intoNEAR accounts . Any NEAR account can hold a contract, needing to pay for thecontract's code and thedata it stores .

Once in an account, anyone can interact with the contract. Thanks to the underlying network structure, executing code from a contract is bothfast (avg. 1.4s finality) andcheap . Moreover,read-only operations arefree for everyone .

tip Storing 100kb costs 1Ⓝ, so deploying a contract generally costs only a few NEARs.

Development flow

Just like any piece of software, smart contracts have a development flow - starting with its creation and ending with monitoring it, all of which we cover in our documentation.

The development flow can be summarized as follows:

- Scaffold
- : The simplest way to create a project is by starting from a template.
- Build
- : To write a contract developers can choose between Javascript and Rust.
- Test
- : Our Sandbox enables to simulate interactions with one or multiple contracts in a realistic environment.
- Deploy
- : After making sure the contract is secure, developers can deploy the contract into their accounts.
- Use
- : Any user can interact with the contract through their NEAR Wallet.
- Monitor
- : The contract's activity can be monitored through simple APIs.

Supported Languages

During the whole cycle, developers can choose betweenJavaScript andRust , allowing them to use their favorite language at each step of their journey.

Details Other languages Theoretically, you can use any language that compiles to Wasm for developing NEAR smart contract. However, in order to have a user-friendly experience we would need to provide a library that wraps around low-level runtime APIs, while also offering other high-level functionalities.

We envision that in the future, more languages will be supported and the support will be done through the effort from the wider community, not just NEAR alone.

Contract primitives

Contract primitives such as FT, NFT, and DAOs are fundamental building blocks that can be combined to create awesome user experiences such as reward tokens, decision-making tools, and marketplaces.

tip Check our section onprimitives to learn more about them Edit this page Last updatedonMar 25, 2024 bygagdiez Was this page helpful? Yes No

Previous Relayer + Gas Station Next Quickstart ✨