NFT Contract Tutorial

Introduction

Non-Fungible Tokens (NFTs) are unique digital assets, each possessing distinct identities and attributes. Unlike fungible tokens created by the Token Factory, NFTs cannot be exchanged on a like-for-like basis.

Colony NFT Seiyan NFT This tutorial guides you through the creation and deployment of an NFT contract on Sei. By the end, you'll have deployed your own NFT contract. Select one of the tabs below to get started!

In this section, we'll be deploying a CW721 contract, a standard for NFTs in the CosmWasm ecosystem. For more about CW721, visithere(opens in a new tab) .

Requirements

Before starting, ensure you have:

- seid
- CLI
- : The Sei command-line interface tool, for interacting with the blockchain.
- Wallet with SEI tokens on testnet
- : Contains SEI tokens for transaction fees.
- Rust Programming Environment
- : Install Rust for CosmWasm contract development. Installation guidehere(opens in a new tab)
- .
- Understanding of CosmWasm
- : Familiarize yourself with CosmWasm smart contracts. Start with theCosmWasm Book(opens in a new tab)
- .
- Docker
- : Required for using the CosmWasm Rust Optimizer tool. Install fromDocker's official website(opens in a new tab)
- .

You can obtain testnet tokens from one of the faucets listedhere .

Setting Up Your Environment

To work with CosmWasm smart contracts, you'll need the Wasm rust compiler installed to build Wasm binaries. To install it, run:

rustup

target

add

wasm32-unknown-unknown Next, clone theCW721-base contract from thecw-nfts(opens in a new tab) repository:

git

clone

https://github.com/CosmWasm/cw-nfts.git cd

cw-nfts/contracts/cw721-base To test your setup, run:

cargo

test You should see that everything in the repository gets compiled and all tests pass.

Customizing the Contract

Review and modify thecw721-base contract to meet your requirements. This might include updating metadata structures or changing the minting process.

Build the Contract

To build the contract, run:

cargo

wasm This compiles a Wasm binary for uploading to Sei.

Note: The generated Wasm file will be located in the root directory of thecw-nfts repository, not in thecw721-base subdirectory. Make sure to navigate to the root directory to see your compiled.wasm file intarget/wasm32-unknown-unknown/ . Before we can upload the contract to the chain, we have to use theCosmWasm Rust Optimizer(opens in a new tab) to reduce the contract size. While not required, this is highly recommended for live contracts.

docker

run

--rm

-v

"(pwd)" :/code \ --mount

type=volume,source= "(basename "(pwd)")_cache" ,target=/target \ --mount

type=volume,source=registry_cache,target=/usr/local/cargo/registry \ cosmwasm/optimizer:0.15.0

./contracts/cw721-base/ This will generate an optimized Wasm contract in/artifacts .

⚠ If you're using a Mac M1 machine, you might need to use the Arm 64-bit optimizer. However, it's important to note that the native Arm version generates wasm artifacts that differ from those produced by the Intel version. For production environments, we strongly recommend building contracts with the Intel optimizers to ensure reliability. Learn morehere(opens in a new tab) .

Deploy the Contract

Upload your contract:

seid

tx

wasm

store

artifacts/cw721_base.wasm

--from=ACCOUNT

--chain-id=atlantic-2

--node=https://rpc.atlantic-2.seinetwork.io/

--broadcast-mode=block

--gas=5000000

--fees=500000usei ReplaceACCOUNT with your account name or address. This command stores the contract on the chain and outputs a code ID.

For detailed descriptions of these arguments, useseid help in the CLI. Instantiate your contract using the code ID:

seid

tx

wasm

instantiate CONTRACT_CODE_ID '{"name":"COLLECTION_NAME", "symbol":"SYMBOL"}'

--from=ACCOUNT

--admin=ADMIN_ADDRESS

--label=LABEL

--chain-id=atlantic-2

--node=https://rpc.atlantic-2.seinetwork.io/

--broadcast-mode=block

--gas=250000

--fees=25000usei ReplaceCONTRACT_CODE_ID ,ACCOUNT ,LABEL , andADMIN_ADDRESS appropriately. Successful instantiation will provide the NFT contract address.

- CONTRACT_CODE_ID
- : Code ID of the uploaded contract (can be found in the output of the previousstore
- command)
- COLLECTION_NAME
- : Name of the collection
- SYMBOL
- : Symbol of the collection
- ACCOUNT
- : Your account name or address
- LABEL
- : Any label for easy identification, can be used to look up the contract in future
- ADMIN_ADDRESS
- (Optional): Address that will have administrative privileges over the contract, such as the ability to upgrade it

Conclusion

Congratulations! You've successfully created and deployed an NFT contract on Sei.