NFT Parser

Rutgers, FinTech Bootcamp, Project 3

Dmitry Kochnov

Contents

Description

About

This presentation about a smart contract I built called NFTParser. NFTParser is a tool that can help developers and NFT collectors extract metadata from NFTs listed on two popular platforms: OpenSea and Blur Network.

This contract defines a smart contract that can parse non-fungible tokens and return their name, symbol, and price in Ether. The contract uses the OpenZeppelin library to interact with ERC721 tokens and the Chainlink library to fetch the latest ETH price.

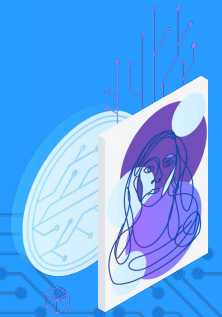
This Solidity code appears to parse NFT information from two different websites (OpenSea and Blur) and determines which NFT is cheaper based on their prices.

Description

How does it work

The getNFTDetailsFromOpenSea() function retrieves the name, symbol, and current price of an NFT from OpenSea using the given URL and the OpenSea Registry contract address. Similarly, the getNFTDetailsFromBlur() function retrieves the name, symbol, and price of an NFT from Blur using the given URL.

The getNFTDetails() function takes the URLs of the NFT from OpenSea and Blur, calls the above two functions to retrieve their details, and returns the name, symbol, and price of the NFT that is cheaper. The buyCheapestNFT() function uses the convertToEth() function to convert the price of the NFT from OpenSea to ETH and then compares it with the price of the NFT from Blur. If the NFT from Blur is cheaper, it buys the NFT from Blur using the buyNFTFromBlur() function, and if the NFT from OpenSea is cheaper or the same price, it buys the NFT from OpenSea using the buyNFTFromOpenSea() function.



Technologies used

Language

For this project, I used the Solidity programming language to develop a smart contract that runs on the Ethereum blockchain.

This contract defines a smart contract that can parse non-fungible tokens (NFTs) and return their name, symbol, and price in Ether. The contract uses the OpenZeppelin library to interact with ERC721 tokens and the Chainlink library to fetch the latest ETH price.

I deployed it on the Ganache test network.

Overall, using Solidity and deploying on Ganache made it easy for us to develop and test our smart contract, ensuring that it was secure and functional before we released it to the public Ethereum network.

Technologies used

Functions that was used

- getNFTName: returns the name of an ERC721 token.
- getNFTSymbol: returns the symbol of an ERC721 token.
- getNFTDetails: given an OpenSea and a Blur URL, returns the name, symbol, and price of the NFT.
- getOpenSeaAssetId: returns the asset ID of an NFT from its OpenSea URL.

- getNFTDetailsFromOpenSea: given an OpenSea URL and the OpenSea registry address, returns the name, symbol, and price of the NFT.
- getNFTPriceFromBlur: given a Blur URL, returns the price of the NFT.
- parseFixedPoint: parses a fixed-point number represented as a string.
- getNFTDetailsFromBlur: given a Blur URL, returns the name, symbol, and price of the NFT.

- startsWith: returns true if a string starts with a given prefix.
- substring: returns a substring of a string.
- getNFTTokenIdFromBlur: given a Blur URL, returns the token ID of the NFT.
- indexOf: finds the first occurrence of a string within another string.

- hexStringToBytes: converts a hexadecimal string to a bytes memory array.
- hexCharToUint: converts a single hexadecimal character to a uint8.
- hexCharToByte: converts a single hexadecimal character to a uint8.
- getNFTAddress: given a URL, returns the address of the NFT.

Conclusion

Conclusion

In conclusion, my NFT parser written in Solidity is a powerful tool for working with NFTs on the Ethereum blockchain. Its ability to retrieve owner, metadata, and token URI information for any given NFT ID, as well as all NFT IDs owned by a specific address, makes it a valuable asset for developers looking to build NFT-related applications.

Conclusion

Demo

To demonstrate the functionality of my NFT parser, I will walk through a simple example using an ERC721 NFT contract. First, I will deploy the contract on Remix and generate some test NFTs. Then, I will use the parser to retrieve information about a specific NFT ID, including the owner, metadata, and token URI. Finally, I will show how the parser can be used to retrieve all NFT IDs owned by a specific address.

Thanks !

The code for NFTParser is open-source and available on Github, so feel free to check it out and contribute if you like.