Blockchain for Industrial Engineers: Decentralized Application Development

บล็อกเชนสำหรับวิศวกรอุตสาหการ: การพัฒนาแอปพลิเคชันแบบกระจายศูนย์

Non-Fungible Token (NFT)

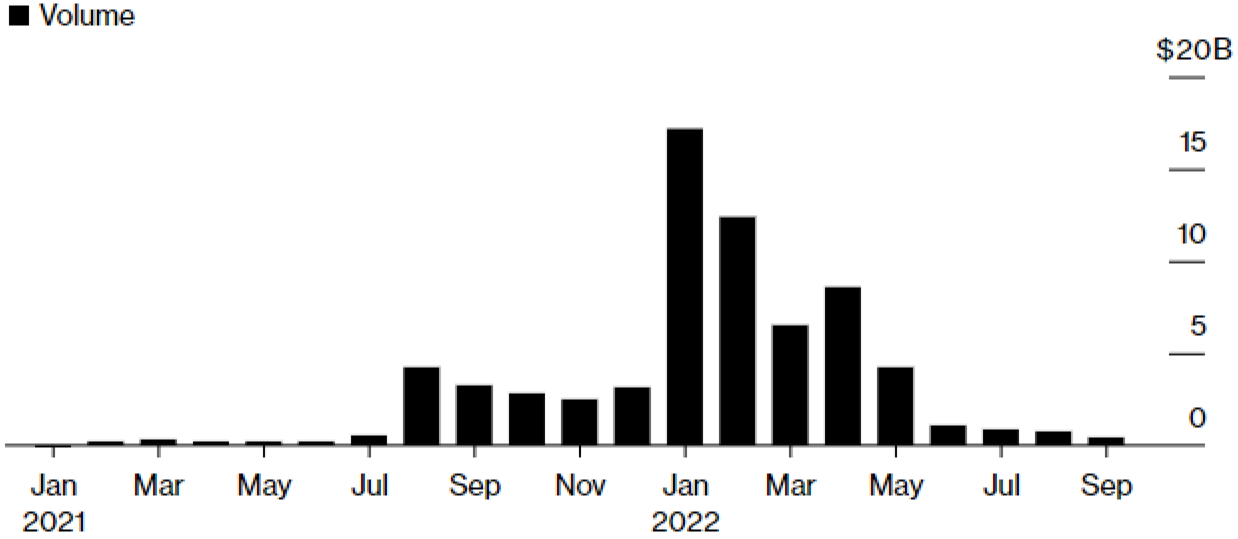
Non-Fungible Token (NFT)?

- Cryptographic assets on a blockchain with unique identification codes and metadata that distinguish them from each other.
- Cannot be traded or exchanged at equivalency.
- NFTs can represent real-world items like artwork and real estate.

Hype?

Volume Drop

NFT monthly volume has dropped 97% from 2022 highs



Source: Dune Analytics; Dashboard by @hildobby
Note: Cumulative data from OpenSea, NFTX, LarvaLabs, LooksRare, SuperRare, Rarible, Foundation

Standard

- ERC-721 standard
    - Most common
- The ERC-1155 standard takes the concept further.
    - Reducing the transaction and storage costs required for NFTs.
    - Natching multiple types of non-fungible tokens into a single contract.

NFTs in action

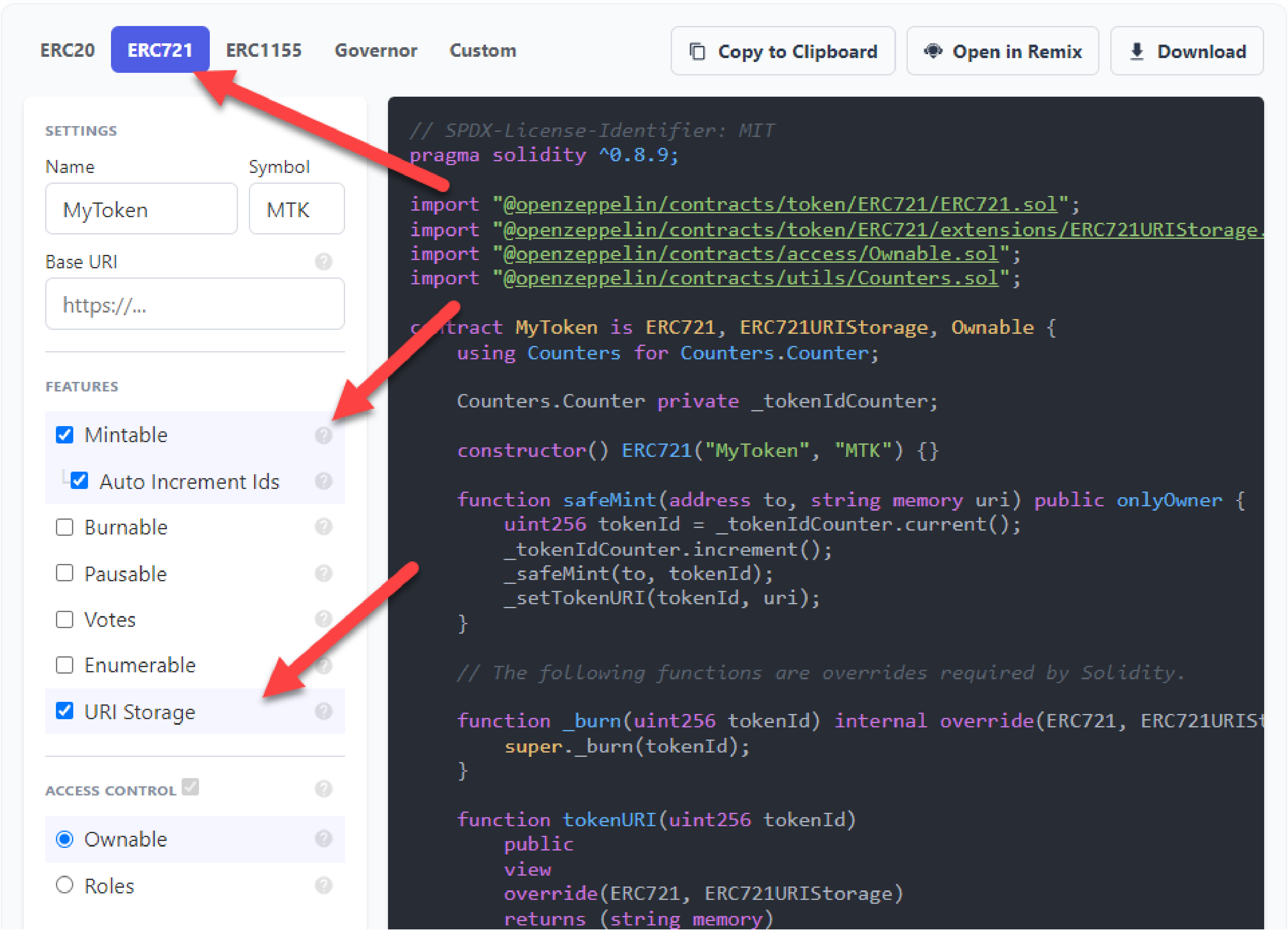
- Mutant Hounds #1775
  - Contract
  - Token URI
  - Image

Create your own NFTs (simple)

OpenSea Testnet

Implementing ERC721

- OpenZeppelin



SETTINGS

Name: MyToken

Symbol: MTK

Base URI: https://...

FEATURES

☑ Mintable

☑ Auto Increment Ids

☐ Burnable

☐ Pausable

☐ Votes

☐ Enumerable

☑ URI Storage

ACCESS CONTROL ☑

● Ownable

○ Roles

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.9;

import "@openzeppelin/contracts/token/ERC721/ERC721.sol";
import "@openzeppelin/contracts/token/ERC721/extensions/ERC721URIStorage.
import "@openzeppelin/contracts/access/Ownable.sol";
import "@openzeppelin/contracts/utils/Counters.sol";

contract MyToken is ERC721, ERC721URIStorage, Ownable {
    using Counters for Counters.Counter;

    Counters.Counter private _tokenIdCounter;

    constructor() ERC721("MyToken", "MTK") {}

    function safeMint(address to, string memory uri) public onlyOwner {
        uint256 tokenId = _tokenIdCounter.current();
        _tokenIdCounter.increment();
        _safeMint(to, tokenId);
        _setTokenURI(tokenId, uri);
    }

    // The following functions are overrides required by Solidity.

    function _burn(uint256 tokenId) internal override(ERC721, ERC721URISt
        super._burn(tokenId);
    }

    function tokenURI(uint256 tokenId)
        public
        view
        override(ERC721, ERC721URIStorage)
        returns (string memory)
```

Source codes

- ERC721
- ERC721 Interface
  - EIP-721
- Ownable.sol
  - `owner` variable, for example.

Function overloading

In some programming languages, function overloading or method overloading is the ability to create multiple functions of the same name with different implementations.

Create your own NFTs (real)

1. **Create image**
2. **Upload to IPFS**

   2.1 Get `image_url`
3. **Create metadata**

   3.1 Save as `JSON` file
4. **Upload metadata file to IPFS**

   4.1 Get `token_uri`
5. **Deploy contract**
6. **Mint token**

Metadata

```json
{
  "name": "<<name>>",
  "description": "<<description>>",
  "external_url": "<<external_link>>",
  "image": "<<image_url>>",
  "attributes": [
    {
      "trait_type": "Personality",
      "value": "Sad"
    },
    {
      "trait_type": "Level",
      "value": 5
    },
    {
      "display_type": "boost_number",
      "trait_type": "Power",
      "value": 40
    }
  ]
}
```