Contract Documentation

OtherWorlds

The Otherworld's contract is an ERC721 token contract that represents a draft of my upcoming collection of NFTs. As an inspiring digital artist and software engineer, it excites me to see an innovation of marketing that allows users to be paid fairly and securely without the use of a third party. This contract was written with the intent to create a maximum supply of five unique NFT's This contract holds ownership to the deployers address. As the contract owner you can create your NFT's and withdraw your funds. When this contract is running on chain, users can purchase the owners NFT's , transferring ownership of the token to the buyer, while compensating the seller via ether.

This contract inherits from several Open Zeppelin contracts: ERC721, ERC721Enumerable, Pausable, and Ownable.

State Variables

- _tokenIdCounter: A counter to keep track of the next available token ID.
- _ContractOwner: The address of the contract owner.
- Mint_Price: The price in ether required to mint an NFT.
- Max_Supply: The maximum number of NFTs that can be created.

Constructor

The constructor function initializes the contract by calling the constructor of the ERC721 contract with the token name "OtherWorlds" and symbol "HEART". It also increments the_tokenIdCounter by one.

External Functions

- **Witchdraw()**: Allows the contract owner to withdraw the contract's balance. It transfers the balance to the contract owner's address.
- **transfer(address payable _to, uint _amount):** Transfers the specified amount of ether to the given address. The to address is declared as payable.
- **sendNFT(address _to, uint256 _tokenId)**: Safely transfers the NFT with the specified token ID to the given address _to.
- **buyNFT(uint256 _tokenId):** Allows a user to buy the NFT with the specified token ID. It verifies that the token ID is valid, the correct amount of ether is sent, and the buyer is not the owner of the NFT. It transfers ownership of the NFT to the buyer and transfers the payment to the previous owner.
- **safeMint(address to):** Mints a new NFT and assigns it to the specified address to. It requires that the sent ether is greater than or equal to the Mint_Price and that the total supply of NFTs is less than Max_Supply.

- **createNFT():** Creates an NFT by the contract owner. It requires the sent ether to be equal to the Mint_Price. It increments the Max_Supply and mints a new NFT assigned to the contract owner.
- **pause():** Pauses the contract. Only the contract owner can call this function.
- **unpause():** Unpauses the contract. Only the contract owner can call this function.
- **_beforeTokenTransfer(address from, address to, uint256 tokenId, uint256 batchSize):** Hook function that is executed before transferring tokens. It ensures that the contract is not paused.
- **_baseURI():** Returns the base URI for the token metadata.

Internal Functions

- **supportsInterface(bytes4 interfaceId)**: Overrides the supportsInterface function of ERC721Enumerable to indicate which interfaces are supported by the contract.

OpenZeppelin Contracts

The OtherWorlds contract imports various OpenZeppelin contracts and uses them for additional functionalities:

ERC721: A contract that implements the ERC721 standard for non-fungible tokens.

ERC721Enumerable: A contract that adds enumerable capabilities to the ERC721 standard.

Pausable: A contract that adds pausable functionality to other contracts.

Ownable: A contract that defines an owner and provides basic authorization control.

Counters: A library for managing a counter. It is used to keep track of the next available token ID.