PerpetualLicense Contract Documentation

Overview

The PerpetualLicense contract is a smart contract for managing perpetual software licenses using the ERC721 token standard. It enables users to purchase and receive licenses, as well as minting new tokens by the owner. This contract also implements royalty for the creators of the license.

ERC721 Extension

This contract extends the ERC721 token standard by implementing the royalty and access control extensions of the OpenZeppelin ERC721Royalty and Ownable contracts respectively.

Functions

The contract has the following functions:

Constructor

The constructor function sets the initial state of the contract. It takes four parameters:

- companyName: a string representing the name of the company offering the license
- licenseName: a string representing the name of the license being offered
- licensePrice: a uint256 representing the price of the license in wei
- royaltyPercentage: a uint96 representing the percentage of royalty that the creator of the license will receive on each sale.

buyToken()

Function signature: function buyToken() public payable

The buyToken function enables users to purchase a perpetual license. It takes no parameters, but requires the user to send the correct amount of ether to cover the license price. If the user sends less ether than the license price, the transaction will revert.

mintToken()

Function signature: function mintToken(address customer) public onlyOwner

The mintToken function enables the owner to mint new perpetual license tokens. It takes one parameter:

- customer: the address of the user receiving the new license.

tokenURI()
Function signature: function tokenURI(uint256 tokenId) public view override returns (string memory)

The tokenURI function returns the metadata associated with a particular token. It takes one parameter:

- tokenId: the ID of the token for which to return metadata.

_baseURI()
Function signature: function _baseURI() internal view override returns (string memory)

The _baseURI function returns the base URI for the metadata associated with the contract.

getTokenCounter()
Function signature: function getTokenCounter() public view returns (uint256)

The getTokenCounter function returns the current value of the s_tokenCounter state variable.

getLicensePrice()
Function signature: function getLicensePrice() public view returns (uint256)

The getLicensePrice function returns the current value of the s_licensePrice state variable.

withdraw()
Function signature: function withdraw() public onlyOwner

The withdraw function enables the owner to withdraw ether from the contract. It takes no parameters.

updateLicensePrice()
Function signature: function updateLicensePrice(uint256 newPrice) public onlyOwner

The updateLicensePrice function enables the owner to update the price of the perpetual license. It takes one parameter:

- newPrice: the new price of the license in wei.

Inherited Functions from ERC721 Token Standard
The contract inherits the following functions from the ERC721 token standard:

- **balanceOf(address _owner):** Returns the number of tokens owned by a particular address.
- **ownerOf(uint256 _tokenId):** Returns the owner of a particular token.

- **approve(address _approved, uint256 _tokenId):** Approves another address to transfer the given token ID.
- **getApproved(uint256 _tokenId):** Returns the approved address for a token ID, or zero if no address is approved.
- **setApprovalForAll(address _operator, bool _approved):** Enables or disables approval for a third party ("operator") to manage all of the caller's tokens.
- **isApprovedForAll(address _owner, address _operator):** Returns true if the given operator is approved to manage all of the caller's tokens.
- **transferFrom(address _from, address _to, uint256 _tokenId):** Transfers the ownership of a given token ID to another address.
- **safeTransferFrom(address _from, address _to, uint256 _tokenId, bytes memory _data):** Safely transfers the ownership of a given token ID to another address, checking first that the recipient is a contract and that it supports ERC721Receiver or ERC721ReceiverUpgradeable interfaces.
- **safeTransferFrom(address _from, address _to, uint256 _tokenId):** Safely transfers the ownership of a given token ID to another address, checking first that the recipient is a contract and that it supports ERC721Receiver or ERC721ReceiverUpgradeable interfaces with extra data.
- **royaltyInfo(uint256 _tokenId, uint256 _value):** Returns the royalty fee information for a given token ID, including the fee recipient and fee amount.

These functions are inherited from the ERC721 standard, which is extended by ERC721Royalty. For additional functionalities, please refer to the official documentation of the ERC721 token standard available here. In addition to these functions, ERC721Royalty adds a royalty fee mechanism for each token.