Registry Contract

Introduction

This Registry contract is a central contract which keeps the record of all the contracts in the ecosystem. It stores the contract name associated with the contract addresses. Anyone can get the addresses of any contract.

The Registry also stores the configurations for the ecosystem ex. Execution fee, supported tokens, extension period etc.

Contract version

pragma solidity 0.8.0

Contract constructor

In this contract the initialize() method is used in place of the constructor to allow the contract to be upgradable via proxy.

Method Name:

- initialize

Method Usage:

- The initialize method is used to initialize the Registry contract and transfers the ownership of the registry to the admin who deploys the registry.

Method Signature:

- function initialize() external;

Method Parameters:

- No parameters required.

Contract Methods

setAddressFor()

Method name:

- setAddressFor

Method detail:

- This method registers the given contract address with the given name.
- Only the registry owner can register the contract.

Method signature:

- function setAddressFor(string calldata identifier, address addr) external override

Method parameters:

- **_identifier:**
  - Indicates the name of the contract in the ecosystem that we want to register.
  - Identifier is specified in the readable string format
- **_addr:**
  - Indicates the address of the contract.

Returned data:

- This method does not return any data.

getAddressForOrDie()

Method name:

- getAddressForOrDie

Method detail:

- This method retrieves the contract`s address associated with the given identifier hash.
- If it does not find the contract`s address for a given identifier hash, it throws an error.

Method signature:

- function getAddressForOrDie(bytes32 _identifierHash)
  external
  override
  view
  returns (address)

Method parameters:

- **_identifierHash:**
  - Indicates the contract name in bytes32 format.

Returned data:

- This method returns the address of the contract associated with the identifier.

getAddressFor()

Method name:

- getAddressFor

Method detail:

- This method retrieves the contract address for the given identifier hash.
- If it does not get the address for the given identifier, it returns the default zero address.

Method signature:

- function getAddressFor(bytes32 _identifierHash)
  external
  override
  view
  returns (address)

Method parameters:

- **_identifierHash:**
  - Indicates the pullPayment contract name in bytes32 format.

Returned data:

- This method returns the pullPayment contract address if it exists or it returns the zero-address.

getAddressForStringOrDie()

Method name:

- getAddressForStringOrDie

Method detail:

- This method returns the contract address for the given contract name.
- It throws an error if it does not find the address for the given contract name.

Method signature:

- function getAddressForStringOrDie(string calldata _identifier)
  external
  view
  override
  returns (address)

Method parameters:

- **_identifier:**
  - Indicates the pullPayment contract name in string format.

Returned data:

- This method returns the pullPayment contract address if it exists.

getAddressForString()

Method name:

- getPPAddressForString

Method detail:

- This method retrieves the address of the pullPayment contract for a given contract name.
- It returns the zero-address if it does not find the contract address associated with the given contract name.

Method signature:

- function getAddressForString(string calldata _identifier)
  external
  view
  override
  returns (address)

Method parameters:

- **_identifier:**
  - Indicates the name of the pullPayment contract in string format.

Returned data:

- This method returns the address of the pullPayment contract.

isOneOf()

Method name:

- isOneOf

Method detail:

- This method tells you whether the given address is registered in the registry or not.
- We need to provide the list of contract identifier hashes and the sender address whose membership to be verified.

Method Signature:

- function isOneOf(bytes32[] calldata identifierHashes, address sender)
  external
  override
  view
  returns (bool)

Method parameters:

- **identifierHashes:**
  - Indicates the list of contract identifier hashes.
- **sender:**
  - Indicates the contract address.

Returned data:

- This method returns true if the given address is one of the registered contract otherwise it returns false.

addToken()

Method name:

- addToken

Method detail:

- This method allows the contract owner to add new tokens for the pullPayments.
- Only the owner can add the supported tokens.
- This method emits a SupportedTokenAdded event.

Method signature:

- function addToken(address tokenAddress) external override;

Method parameters:

1. **tokenAddress:**
   - Indicates the token contract address which is to add for the pullPayments

Returned data:

- This method does not return any data.

removeToken()

Method name:

- removeToken

Method detail:

- This method allows the contract owner to remove from the list of supported tokens.
- Only the owner can remove the supported tokens.
- This method emits a **SupportedTokenRemoved** event

Method signature:

- function removeToken(address tokenAddress) external override;

Method parameters:

2. **tokenAddress:**
   - Indicates the token contract address which is to be removed.

Returned data:

- This method does not return any data.

updateExecutionFeeReceiver()

Method name:

- updateExecutionFeeReceiver

Method detail:

- This method allows the admin to update the execution fee receiver address.
- This method emits a **UpdatedExecutionFeeReceiver** event

Method signature:

- function updateExecutionFeeReceiver(address _newReceiver) public virtual

Method parameters:

1. **_newReceiver:**
    - Indicates the new execution fee receiver`s address.

Returned data:

- This method does not return any data.

updateExecutionFee()

Method name:

- updateExecutionFeeReceiver

Method detail:

- This method allows the admin to update the execution fee which is used to calculate the amount of PMA tokens to charge for the pullpayment execution.
- This method emits a **UpdatedExecutionFee** event.

Method signature:

- function updateExecutionFee(uint256 _newFee) public virtual

Method parameters:

1. **_newFee:**
    - Indicates the new execution fee.
    - New fee must be less than 10000. i.e 100%
    - Here, 100 equals 1%.

Returned data:

- This method does not return any data.

updateExtensionPeriod()

Method name:

- updateExtensionPeriod

Method detail:

- This method allows admin to update the extension period duration.
- The extension period is the duration in seconds given to the subscribers in order to top up their account in case of low balance subscription.

Method signature:

- function updateExtensionPeriod(uint256 _newPeriod) external virtual

Method parameters:

2. **_newPeriod:**
    - Indicates the new extension period in seconds.

Returned data:

- This method does not return any data.

getSupportedTokens()

Method name:

- getSupportedTokens

Method detail:

- This method retrieves the list of supported token addresses.

Method signature:

- function getSupportedTokens() external override view returns (address[] memory)

Method parameters:

- This method does not take any parameters.

Returned data:

- This method returns the list of token addresses which supports the pullPayment.

getPMAToken()

Method name:

- getPMAToken

Method detail:

- This method returns the address of the PMA token.

Method signature:

- function getPMAToken() public view virtual returns (address)

getWBNBToken()

Method name:

- getWBNBToken

Method detail:

- This method returns the address of the WBNB token.

Method signature:

- function getWBNBToken() public view virtual returns (address)

getExecutor()

Method name:

- getExecutor

Method detail:

- This method returns the address of the Executor contract.

Method signature:

- function getExecutor() public view virtual returns (address)

getUniswapFactory()

Method name:

- getUniswapFactory

Method detail:

- This method returns the address of the Dex Factory contract.

Method signature:

- function getUniswapFactory() public view virtual returns (address)

getUniswapRouter()

Method name:

- getUniswapRouter

Method detail:

- This method returns the address of the Dex Router contract.

Method signature:

- function getUniswapRouter() public view virtual returns (address)

getPullPaymentRegistry()

Method name:

- getPullPaymentRegistry

Method detail:

- This method returns the address of the PullPayment Registry contract.

Method signature:

- function getPullPaymentRegistry() public view virtual returns (address)

getKeeperRegistry()

Method name:

- getKeeperRegistry

Method detail:

- This method returns the address of the Keeper Registry contract.

Method signature:

- function getKeeperRegistry() public view virtual returns (address)

getTokenConverter()

Method name:

- getTokenConverter

Method detail:

- This method returns the address of theToken Converter contract.

Method signature:

- function getTokenConverter() public view virtual returns (address)

executionFeeReceiver()

Method name:

- executionFeeReceiver

Method detail:

- This method returns the address of the execution fee receiver.

Method signature:

- function executionFeeReceiver() public view returns (address)

executionFee()

Method name:

- executionFee

Method detail:

- This method returns the execution fee percentage.

Method signature:

- function executionFee() public view returns (uint256)

extensionPeriod()

Method name:

- extensionPeriod

Method detail:

- This method returns the extension period value in seconds.

Method signature:

- function extensionPeriod() public view returns (uint256)

Events

RegistryUpdated

Event Name:

- RegistryUpdated

Event detail:

- This event is emitted when a new contract is registered using the **setAddressFor()** method.

Event signature:

- event RegistryUpdated(
  string identifier,
  bytes32 indexed identifierHash,
  address indexed addr
  );

Event data:

- **identifier:**
  - Indicates the name of the contract.
- **identifierHash:**
  - Indicates the contract name in bytes32 format.
- **Addr:**
  - Indicates the address of the registered contract.

SupportedTokenAdded

Event Name:

- SupportedTokenAdded

Event detail:

- This event is emitted when a supported ERC20/BEP20 token is added in the list using the **addToken()** method

Event signature:

- event SupportedTokenAdded(address indexed _token);

Event data:

- **_token:**
  - Indicates the supported token`s address.

SupportedTokenRemoved

Event Name:

- SupportedTokenRemoved

Event detail:

- This event is emitted when a supported ERC20/BEP20 token is removed from the list using the **removeToken()** method

Event signature:

- event SupportedTokenRemoved(address indexed _token);

Event data:

- **_token:**
  - Indicates the supported token`s address.

UpdatedExecutionFeeReceiver

Event Name:

- UpdatedExecutionFeeReceiver

Event detail:

- This event is emitted when the admin updates the execution fee receiver address using the **updateExecutionFeeReceiver()** method.

Event signature:

- event UpdatedExecutionFeeReceiver(address indexed _newReceiver);

Event data:

- _newReceiver**:**
  - Indicates the address of the new execution fee receiver.

UpdatedExecutionFee

Event Name:

- UpdatedExecutionFee

Event detail:

- This event is emitted when the admin updates the execution fee using the **updateExecutionFee** method.

Event signature:

- event UpdatedExecutionFee(uint256 indexed _newFee);

Event data:

- **_newFee:**
  - Indicates the new execution fee.