

# HBTrust.domain.token Manual

## Overview

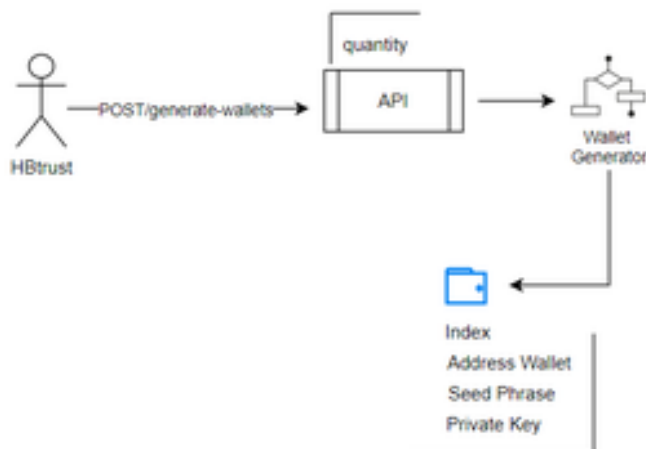
The HBTrust.domain.token API provides the necessary services to manage and execute the PIT Token smart contract functions, in addition to the generation of PoP wallets and balance check.

### PIT Token

API has PIT token management services: management of Minter, Burner and Admin roles (add, remove and query), management of Total Supply of PIT token (mint token PIT, burn token PIT) in addition to token balance consultation PIT.

### Generate Wallet

API has the algorithm to generate an **Address**, a **Private Key** and a **Seed Phrase (mnemonic)** based on one or more input parameters.



## Token Administration

To administer the Token PIT it is necessary to have a PoP wallet. With the address of that wallet the contract owner can add that wallet as the administrator of the smart contract Token PIT.

owner	The address of the wallet that publishes the smart contract on the network. This address is inserted in all smart contract roles at the time of deployment.
carteira	Wallets allow PoP users to store tokens and interact with smart contracts on the PoP network.
address	PoP addresses are unique identifiers derived from public keys.  Private Key -> Public Key -> PoP Address
smart contract	A smart contract is the application code published on the PoP blockchain network. Executes the business rules defined for the Token.
token	Digital assets developed on the PoP Blockchain.
Total Supply	The total number of Tokens that are allocated in the smart contract regardless of the owner.

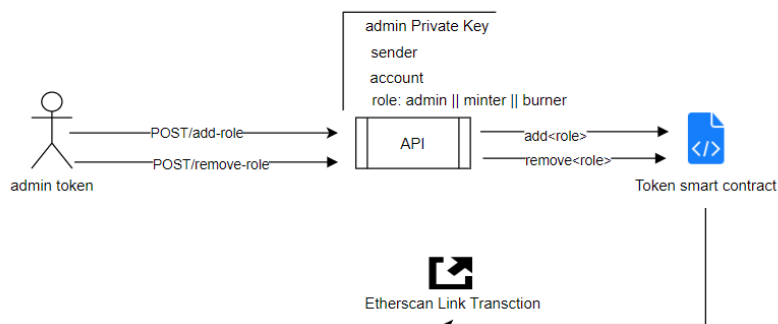
## Administrative functions

Roles define the roles performed by each entity.

### Admin role

- Address of the wallet responsible for controlling the **add** and **remove** functions of the minter, burner and admin roles.

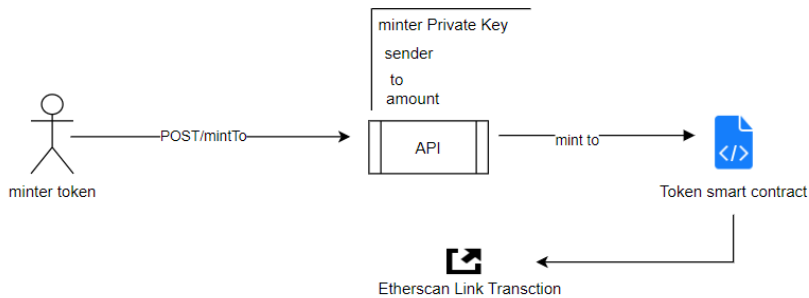
API Method	Smart Contract Function
POST / Add Role	addAdmin, addMinter, addBurner
POST / Remove Role	removeAdmin, removeMinter , removeBurner



## Minter Role

- Address of the wallet responsible for minting (creating) new PIT Tokens. Each new Token created is added to the Total Supply.

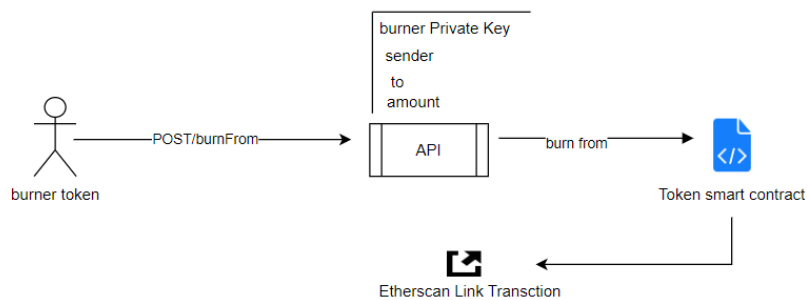
API Method	Smart Contract Function
POST / Mint To	<code>mintTo</code>



## Burner Role

- Address of the wallet responsible for eliminating PIT Tokens. Each "burned" Token is subtracted from Total Supply

API Method	Smart Contract Function
POST / Burn From	<code>burnFrom</code>



# General Functions

## Token PIT

- Given the function sender's address (msg.sender) he can transfer PIT tokens from his wallet to a second wallet and check the balance in PIT token.

API Method	Smart Contract Function
POST / Transfer	transfer
GET / Balance	balanceOf

## PoP Blockchain

### POST / Generate Wallet

---

- Method used to generate PoP wallets:

<b>BIP-39</b>	Generate Address and Seed Phrase.
<b>HDKey</b>	Generate Private Key
<b>ETHUtil</b>	Transform Address.
<b>Web3</b>	Wallet test

### GET / Balance PoP

---

- Given the sender's address for the function (msg.sender) he can view the balance in PoP of his wallet.

# Run a method using swagger

swagger	<a href="http://54.90.5.188:3000/swagger/">http://54.90.5.188:3000/swagger/</a>
Rinkeby	<a href="https://rinkeby.etherscan.io/address/0x5a093dd37b635f4ae8d9b3be65d3f4e7b3dd187c">https://rinkeby.etherscan.io/address/0x5a093dd37b635f4ae8d9b3be65d3f4e7b3dd187c</a>

On the swagger page you can select a GET / POST / PUT method and click the *try it out* button.

GET methods can be performed by entering the parameters in the required fields.

POST / PUT methods can be executed by editing the body with the necessary parameters.

After editing, click **Execute** and the method will be executed.

Method	Description
<b>POST</b>	The HTTP method <code>POST</code> is used to create resources or to send information that will be processed. For example, mint tokens, adding an address to a role.
<b>GET</b>	The HTTP method <code>GET</code> is used to query existing resources. For example, query for roles.

## Methods

Role

**GET /verify-role/{address}**

---

Through this method it is possible to check which role an address belongs to (admin, minter or burner).

### Request

- Inform in PATH the address of the wallet being verified

address \* required

string

(path)

Address

0xf11777b305567F72D3c73f29986630B1242

### Response

- Return true / false for each role

```
{
  "success": true,
  "code": "100-7012",
  "message": "Wallet address has no role.",
  "admin": false,
  "minter": false,
  "burner": false
}
```

## POST/add-role

---

The methods called roles need to be signed by address that have admin permission within the Token smart contract. This signature is called a private key.

Through this method it is possible to add an address in one of the roles: Admin, Minter or Burner.

### Request

- Private Key - whoever carries out this transaction needs to be in the "admin" role. This private key must be from a PIT Token smart contract administrator
- sender - PIT Token smart contract admin address
- account - Wallet address that will be added to the role
- role - role name: admin, burner or minter

```
{
  "privateKey": "C12BBFEA10C5DB51DF680BF11A7C0909067B38924E9E8C763E5721039D588A55",
  "_sender": "0x3A0d038BFe0573BcdC7d2766E3eC3f3A2f68a52f",
  "_account": "0xf11777b305567F72D3c73f2998663081242461aF",
  "_role": "admin"
}
```

### Response

- link - Blockexplorer link for the transaction

```
{
  "success": true,
  "code": "100-7015",
  "message": "Wallet address has been added to role admin.",
  "hash": "0xcce25c0ff89941c996a5ead0809b746aded8c4977f12568f03ce399c859bf952e",
  "link": "https://rinkeby.etherscan.io/tx/0xcce25c0ff89941c996a5ead0809b746aded8c4977f12568f03ce399c859bf952e"
}
```

## POST/remove-role

---

The methods called roles need to be signed by address that have admin permission within the Token smart contract. This signature is called a private key.

Through this method it is possible to remove an address from one of the roles: Admin, Minter or Burner.


### Request

- Private Key - Whoever carries out this transaction needs to be in the "admin" role. This private key must be from a PIT Token smart contract administrator

- sender - PIT Token smart contract admin address
- account - wallet address that will be added to the role
- role - role name: admin, burner or minter

```
{
  "privateKey": "C12BBFEA10C5DB51DF6808F11A7C0909067B38924E9E8C763E5721039D588A55",
  "_sender": "0x3A0d038BFe0573BcdC7d2766E3eC3f3A2f68a52f",
  "_account": "0xf11777b305567F72D3c73f2998663081242461aF",
  "_role": "admin"
}
```

## Response

- link - Blockexplorer link for the transaction 

```
{
  "success": true,
  "code": "100-7014",
  "message": "Wallet address has been removed from role admin.",
  "hash": "0x4b2dfeaa696a8df87c7eaac9abd44e6212333fa9dd466d47a908e9082606ff9e"
}
```

## Mint Token PIT

### POST/mintTo

---

Through this method it is possible to mint, that is, create PIT Tokens in a wallet.

This method needs to be signed by an address that has the minter permission within the Token smart contract. This signature is called a private key.

## Request

- Private Key - Whoever carries out this transaction must be in the role "minter". This private key must be from a PIT Token smart contract minter
- sender - PIT Token smart contract minter address
- account - Wallet address that will receive the PIT Tokens
- amount- Amount of PIT tokens that will be created.

```
{
  "privateKey": "C12BBFEA10C5DB51DF6808F11A7C0909067B38924E9E8C763E5721039D588A55",
  "_sender": "0x3A0d038BFe0573BcdC7d2766E3eC3f3A2f68a52f",
  "_to": "0xf11777b305567F72D3c73f2998663081242461aF",
  "_amount": 2000
}
```

## Response

- link - Blockexplorer link for the transaction

```
{
  "success": true,
  "code": "100-1000",
  "message": "Sucess!",
  "hash": "0x69acc87bb5841000f0d5cd64bd76fc764a4021d0164fca0473e9bf80008bb344",
  "link": "https://rinkeby.etherscan.io/tx/0x69acc87bb5841000f0d5cd64bd76fc764a4021d0164fca0473e9bf80008bb344"
}
```

## Burn Token HBT

### POST/burnFrom

---

Through this method it is possible to burn tokens, that is, to destroy PIT Tokens from a wallet.

This method must be signed by an address that has burner permission within the Token smart contract. This signature is called a private key.

#### Requisição

- Private Key - whoever carries out this transaction must be in the role “burner”. This private key must be from a PIT Token smart contract burner
- sender - Token PIT smart contract burner address
- account - Wallet address that will have the PIT Tokens eliminated
- amount- amount of HBT tokens that will be eliminated.

```
{
  "privateKey": "C12BBFEA10C5DB51DF680BF11A7C0909067B38924E9E8C763E5721039D588A55",
  "_sender": "0x3A0d038BFe0573BcdC7d2766E3eC3f3A2f68a52f",
  "_to": "0xf11777b305567F72D3c73f29986630B1242461aF",
  "_amount": 2000
}
```

#### Response

- link - Blockexplorer link for the transaction

```
{
  "success": true,
  "code": "100-1000",
  "message": "Sucess!",
  "hash": "0x33b5d62e5801348695967ed4deca27b7c41858d7c0071160b333b83256963a4d",
  "link": "https://rinkeby.etherscan.io/tx/0x33b5d62e5801348695967ed4deca27b7c41858d7c0071160b333b83256963a4d"
}
```



## Transfer

### POST/transfer

---

Through this method it is possible to transfer PIT Tokens from one wallet to another.

#### Request

- Private Key - Who carries out the transfer transaction (the sender).
- sender - sender address of PIT tokens
- account - wallet address that will receive the PIT tokens
- amount- amount of PIT tokens to be sent.

```
{
  "privateKey": "C12BBFEA10C5DB51DF680BF11A7C0909067B38924E9E8C763E5721039D588A55",
  "_sender": "0x3A0d03BBFe0573BcdC7d2766E3eC3f3A2f68a52f",
  "_to": "0xf11777b305567F72D3c73f29986630B1242461aF",
  "_amount": 1000
}
```

#### Response

- link - Blockexplorer link for the transaction

```
{
  "success": true,
  "code": "100-1000",
  "message": "Sucess!",
  "hash": "0x886dfb7560dbfb2bd969a56effec2c010a60cbccc99342ee533103b4e1444c07",
  "link": "https://rinkeby.etherscan.io/tx/0x886dfb7560dbfb2bd969a56effec2c010a60cbccc99342ee533103b4e1444c07"
}
```

## Balance

### GET/balance/{owner}

---

Through this method it is possible to check the amount of PIT Token in a wallet.

#### Request

- address - Wallet to be checked

<b>owner</b> <small>* required</small>	Address
string (path)	<input type="text" value="0x3A0d03BBFe0573BcdC7d2766E3eC3f3A2"/>

## Response

- balance - amount of PIT tokens

```
{
  "success": true,
  "code": "100-1000",
  "message": "Success!",
  "balance": 13300,
  "link": "https://rinkeby.etherscan.io/tx/undefined"
}
```

## GET/balancePoP/{owner}

---

Through this method it is possible to check the amount of PoP of a wallet.

## Request

- address - Wallet to be checked

<b>owner</b> <small>* required</small>	Address
string (path)	<input type="text" value="0x3A0d03BBFe0573BcdC7d2766E3eC3f3Az"/>

## Response

- balance - Amount of PoP

```
{
  "success": true,
  "code": "100-1000",
  "message": "Success!",
  "balance": "18.748332339",
  "link": "https://rinkeby.etherscan.io/tx/undefined"
}
```

## Generate Wallet

### POST/generate-wallets

---

Through this method it is possible to create wallets.

#### Request

- quantity - Number of wallets to be created

```
{  
  "quantity": 2  
}
```

#### Response

- index - index of the generated wallet
- mnemonic - group of words that give access to wallet. It needs to be treated as your private key because with mnemonic anyone can access your wallet
- address - your wallet address
- private Key - your subscription for transactions made on PoP

```
{  
  "success": true,  
  "result": [  
    {  
      "index": 1,  
      "mnemonic": "hill enable region spirit more travel hero festival garbage cycle gaze toddler",  
      "address": "0xde2cf0104882eA3b8F0c0582ae93C498339fED1A",  
      "privateKey": "1a58cb7badfc75d97e1d148a385b4b3655a1d47c2ede27b40e41c442b641eb2b"  
    },  
    {  
      "index": 2,  
      "mnemonic": "inside fly ready barely flame gas future foot huge prevent measure split",  
      "address": "0xd5a4f35d2F43937821645f0F49Fe2Ac2d80837f7",  
      "privateKey": "d2e1549de3a6f6c6cf027d9282dba03fd4555c0f25b6b6f75b0cd62a5c60b8ba"  
    }  
  ]  
}
```