merkle patricia trie Documentation

Release 0.1.0

Igor Aleksanov

Apr 02, 2019

API documentation:

MPT is the data structure used in Ethereum as a cryptographically authenticated key-value data storage.

This library is a Python implementation of Modified Merkle Patrica Trie with a very simple interface.

CHAPTER 1

Example

```
storage = {}
trie = MerklePatriciaTrie(storage)

trie.update(b'do', b'verb')
trie.update(b'dog', b'puppy')
trie.update(b'doge', b'coin')
trie.update(b'horse', b'stallion')

old_root = trie.root()
old_root_hash = trie.root_hash()

print("Root hash is {}".format(old_root_hash.hex()))

trie.delete(b'doge')

print("New root hash is {}".format(trie.root_hash().hex()))

trie_from_old_hash = MerklePatriciaTrie(storage, root=old_root)

print(trie_from_old_hash.get(b'doge'))

try:
    print(trie.get(b'doge'))
except KeyError:
    print('Not accessible in a new trie.')
```

1.1 mpt

1.1.1 mpt package

Submodules

mpt.mpt module

class mpt.mpt.**MerklePatriciaTrie**(*storage*, *root=None*, *secure=False*)
Bases: object

__init__(*storage*, *root=None*, *secure=False*)
Creates a new instance of MPT.

MerklePatriciaTrie works like a wrapper over provided storage. Storage must implement dict-like interface. Any data structure that implements *__getitem__* and *__setitem__* should be OK.

Parameters

- **storage** (*dict-like*) – Data structure to store all the data of MPT.
- **root** (*bytes*) – (Optional) Root node (not root hash!) of the trie. If not provided, tree will be considered empty.
- **secure** (*bool*) – (Optional) In secure mode all the keys are hashed using keccak256 internally.

Returns An instance of MPT.

Return type *MerklePatriciaTrie*

delete(*encoded_key*)
This method removes a value associtated with provided key.

Note: this method does not RLP-encode the key. If you use encoded keys, you should encode it yourself.

Parameters **encoded_key** (*bytes*) – RLP-encoded key.

Raises KeyError – KeyError is raised if there is no value assotiated with provided key.

get(*encoded_key*)
This method gets a value associtated with provided key.

Note: this method does not RLP-encode the key. If you use encoded keys, you should encode it yourself.

Parameters **encoded_key** (*bytes*) – RLP-encoded key.

Returns Stored value associated with provided key.

Return type bytes

Raises KeyError – KeyError is raised if there is no value assotiated with provided key.

root()
Returns a root node of the trie. Type is *bytes* if trie isn't empty and *None* othrewise.

root_hash()
Returns a hash of the trie's root node. For empty trie it's the hash of the RLP-encoded empty string.

update(*encoded_key*, *encoded_value*)
This method updates a provided key-value pair into the trie.

If there is no such a key in the trie, a new entry will be created. Otherwise value associtaed with key is updated. Note: this method does not RLP-encode neither key or value. If you use encoded keys, you should encode it yourself.

Parameters

- **encoded_key** (*bytes*) – RLP-encoded key.
- **encoded_value** (*bytes*) – RLP-encoded value.

Module contents

mpt

Python implementation of Merkle Patricia Trie.

copyright © 2019 by Igor Aleksanov.

license MIT, see LICENSE for more details.

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

Python Module Index

m

Index

Symbols

D

G

M

R

U