Hash table

0.1

Generated by Doxygen 1.8.17

Chapter 1

Module Index

1.1 Modules

Here is a list of all modules:

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 3

File Index

3.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 4

Module Documentation

4.1 (chaining)

Item of hash table.

Classes

- struct hash_table_t

  Hash table data structure.

Typedefs

- typedef struct hash_table_t hash_table_t

  Hash table data structure.

Functions

- void hash_table_init (hash_table_t ∗table, size_t max_count)

  Init hash table data structure.
- size_t hash_table_count (hash_table_t ∗table)

  Returns count elements of hash table.
- size_t hash_table_max_count (hash_table_t ∗table)

  Returns max count elements of hash table.
- void hash_table_free (hash_table_t ∗table)

  Frees memory in data structure.
- unsigned long elf_hash (const unsigned char ∗s, size_t max_size)

  Frees memory in data structure.
- void hash_table_add (hash_table_t ∗table, char ∗key, int value)

  Add the given key and object to hash table. If key exists, update the value.
- unsigned short hash_table_is_exist (hash_table_t ∗table, char ∗key)

  Returns true if the given key exists in the table.
- int ∗ hash_table_is_get (hash_table_t ∗table, char ∗key)

  Returns the value associated with the given key, or NULL if it doesn't exist.
- void hash_table_remove (hash_table_t ∗table, char ∗key)

  Removes the value associated with key from the table.

4.1.1 Detailed Description

Item of hash table.

Warning

This structure created only for educational goals

4.1.2 Typedef Documentation

4.1.2.1 hash_table_t

```
typedef struct hash_table_t hash_table_t
```

Hash table data structure.

()

Warning

This structure created only for educational goals

4.1.3 Function Documentation

4.1.3.1 elf_hash()

```
unsigned long elf_hash (
             const unsigned char * s,
             size_t max_size )
```

Frees memory in data structure.

()

Parameters

table	Pointer to hash table data structure.

4.1.3.2 hash_table_add()

```
void hash_table_add (
             hash_table_t * table,
             char * key,
             int value )
```

Add the given key and object to hash table. If key exists, update the value.

()

Parameters

table	Pointer to hash table data structure.
key	Key for value.
value	Value by key.

4.1.3.3 hash_table_count()

```
size_t hash_table_count (
            hash_table_t * table )
```

Returns count elements of hash table.

()

Parameters

table	Pointer to hash table data structure.

Returns

count elements of hash table.

4.1.3.4 hash_table_free()

```
void hash_table_free (
            hash_table_t * table )
```

Frees memory in data structure.

()

Parameters

table	Pointer to hash table data structure.

4.1.3.5 hash_table_init()

```
void hash_table_init (
            hash_table_t * table,
            size_t max_count )
```

Init hash table data structure.

()

Parameters

table	Pointer to hash table data structure.
max_count	Max count elements of hash table

4.1.3.6 hash_table_is_exist()

```
unsigned short hash_table_is_exist (
            hash_table_t * table,
            char * key )
```

Returns true if the given key exists in the table.

()

Parameters

table	Pointer to hash table data structure.
key	Key for search.

Returns

true if key exist else false

4.1.3.7 hash_table_is_get()

```
int* hash_table_is_get (
            hash_table_t * table,
            char * key )
```

Returns the value associated with the given key, or NULL if it doesn't exist.

()

Parameters

table	Pointer to hash table data structure.
key	Key for search.

Returns

value(pointer) associated with the given key, or NULL if it doesn't exist

4.1.3.8 hash_table_max_count()

```
size_t hash_table_max_count (
            hash_table_t * table )
```

Returns max count elements of hash table.

()

Parameters

table	Pointer to hash table data structure.

Returns

max count elements of hash table.

4.1.3.9 hash_table_remove()

```
void hash_table_remove (
            hash_table_t * table,
            char * key )
```

Removes the value associated with key from the table.

()

Parameters

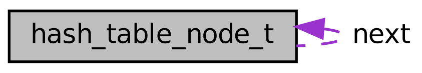
table	Pointer to hash table data structure.
key	Key for remove.

Chapter 5

Class Documentation

5.1 hash_table_node_t Struct Reference

Collaboration diagram for hash_table_node_t:



Public Attributes

- char ∗ **key**
- int **value**
- struct hash_table_node_t ∗ **next**

The documentation for this struct was generated from the following file:

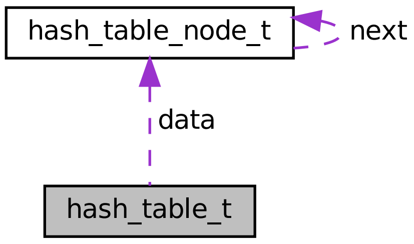
- include/hash_table.h

5.2 hash_table_t Struct Reference

Hash table data structure.

```
#include <hash_table.h>
```

Collaboration diagram for hash_table_t:



Public Attributes

- node_t ** **data**
- size_t max_count

  max count elements of table
- size_t count

  current count elements of table

5.2.1 Detailed Description

Hash table data structure.

()

Warning

This structure created only for educational goals

The documentation for this struct was generated from the following file:

- include/hash_table.h

Chapter 6

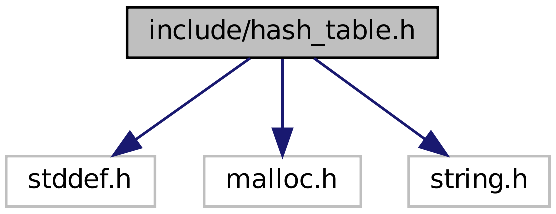
File Documentation

6.1 include/hash_table.h File Reference

Header file for hash table This file contains the definition of the data structure hash table.

```
#include <stddef.h>
#include <malloc.h>
#include <string.h>
```

Include dependency graph for hash_table.h:



Classes

- struct hash_table_node_t
- struct hash_table_t

  Hash table data structure.

Typedefs

- typedef struct hash_table_node_t **node_t**
- typedef struct hash_table_t hash_table_t

  Hash table data structure.

Functions

- void hash_table_init (hash_table_t ∗table, size_t max_count)

  Init hash table data structure.
- size_t hash_table_count (hash_table_t ∗table)

  Returns count elements of hash table.
- size_t hash_table_max_count (hash_table_t ∗table)

  Returns max count elements of hash table.
- void hash_table_free (hash_table_t ∗table)

  Frees memory in data structure.
- unsigned long elf_hash (const unsigned char ∗s, size_t max_size)

  Frees memory in data structure.
- void hash_table_add (hash_table_t ∗table, char ∗key, int value)

  Add the given key and object to hash table. If key exists, update the value.
- unsigned short hash_table_is_exist (hash_table_t ∗table, char ∗key)

  Returns true if the given key exists in the table.
- int ∗ hash_table_is_get (hash_table_t ∗table, char ∗key)

  Returns the value associated with the given key, or NULL if it doesn't exist.
- void hash_table_remove (hash_table_t ∗table, char ∗key)

  Removes the value associated with key from the table.

6.1.1 Detailed Description

Header file for hash table This file contains the definition of the data structure hash table.

Index