HashMap

Generated by Doxygen 1.9.8

Chapter 1

File Index

1.1 File List

Here is a list of all files with brief descriptions:

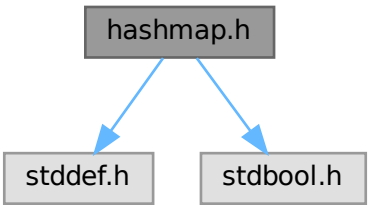
Chapter 2

File Documentation

2.1 hashmap.h File Reference

```
#include <stddef.h>
#include <stdbool.h>
```

Include dependency graph for hashmap.h:



Typedefs

- typedef struct HashMap * HashMap
- typedef size_t(* hash) (const void *key, size_t key_size)

Functions

- HashMap hm_create (size_t hm_capacity, size_t key_size, size_t value_size, hash hash_func)

  Creates a new HashMap with a hash function.
- int hm_destroy (HashMap hm)

  Destroys the HashMap.
- void * hm_get (HashMap hm, const void *key)

  Retrieves the value at the specified key.
- int hm_set (HashMap hm, const void *key, const void *value)

  Updates the value at the specified key.

- int hm_put (HashMap hm, const void *key, const void *value)

  Adds a new key-value pair to the HashMap.
- bool hm_contains (HashMap hm, const void *key)

  Tests if the HashMap contains the specified key.
- size_t hm_size (HashMap hm)

  Returns the size of the HashMap.
- int hm_remove (HashMap hm, const void *key)

  Removes the key-value pair from the HashMap.

2.1.1 Typedef Documentation

2.1.1.1 hash

```
typedef size_t(* hash) (const void *key, size_t key_size)
```

2.1.1.2 HashMap

```
typedef struct HashMap* HashMap
```

2.1.2 Function Documentation

2.1.2.1 hm_contains()

```
bool hm_contains (
            HashMap hm,
            const void * key )
```

Tests if the HashMap contains the specified key.

Parameters

hm	The HashMap
key	The key

Returns

True or Falsehood

2.1.2.2 hm_create()

```
HashMap hm_create (
            size_t hm_capacity,
            size_t key_size,
            size_t value_size,
            hash hash_func )
```

Creates a new HashMap with a hash function.

Parameters

hm_capacity	The initial capacity of the HashMap
key_size	The sizeof value of the key
value_size	The sizeof value of the value
hash_func	a custom hash function. Pass NULL for generic hashing.

Returns

HashMap

2.1.2.3 hm_destroy()

```
int hm_destroy (
            HashMap hm )
```

Destroys the HashMap.

Parameters

hm	The HashMap

Returns

Success code

2.1.2.4 hm_get()

```
void * hm_get (
            HashMap hm,
            const void * key )
```

Retrieves the value at the specified key.

Parameters

hm	The HashMap
key	The key

Returns

The value at the key

2.1.2.5 hm_put()

```
int hm_put (
            HashMap hm,
```

```
            const void * key,
            const void * value )
```

Adds a new key-value pair to the HashMap.

Parameters

hm	The HashMap
key	The key
value	The value

Returns

Success code

2.1.2.6 hm_remove()

```
int hm_remove (
            HashMap hm,
            const void * key )
```

Removes the key-value pair from the HashMap.

Parameters

hm	The HashMap
key	The key

Returns

Success code

2.1.2.7 hm_set()

```
int hm_set (
            HashMap hm,
            const void * key,
            const void * value )
```

Updates the value at the specified key.

Parameters

hm	The HashMap
key	The key
value	The value

Returns

Success code

2.1.2.8 hm_size()

```
size_t hm_size (
            HashMap hm )
```

Returns the size of the HashMap.

Parameters

hm	The HashMap

Returns

The size

2.2 hashmap.h

Go to the documentation of this file.

```
00001 //
00002 // Created by Emanuel on 02.09.2024.
00003 //
00004
00005 #ifndef HASHMAP_H
00006 #define HASHMAP_H
00007
00008 #include <stddef.h>
00009 #include <stdbool.h>
00010
00011 typedef struct HashMap *HashMap;
00012 typedef size_t (*hash)(const void *key, size_t key_size);
00013
00023 HashMap hm_create(size_t hm_capacity, size_t key_size, size_t value_size, hash hash_func);
00024
00031 int hm_destroy(HashMap hm);
00032
00040 void *hm_get(HashMap hm, const void *key);
00041
00050 int hm_set(HashMap hm, const void *key, const void *value);
00051
00060 int hm_put(HashMap hm, const void *key, const void *value);
00061
00062
00070 bool hm_contains(HashMap hm, const void *key);
00071
00078 size_t hm_size(HashMap hm);
00079
00087 int hm_remove(HashMap hm, const void *key);
00088
00089 #endif //HASHMAP_H
```

Index