Implementing libc

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 Memory_management

Header file for malloc This file contains functions implementation from libs (malloc.h)

Classes

- struct memory_control_block

  Memory block characteristic.

Typedefs

- typedef struct memory_control_block mcb

  Memory block characteristic.

Functions

- void ∗ simple_malloc (unsigned long size)

  Allocates size bytes of uninitialized storage.
- void ∗ simple_calloc (unsigned long num, unsigned long size)

  Allocates memory for an array of num objects of size size and initializes it to all bits zero.
- void ∗ simple_realoc (void ∗ptr, unsigned long size)

  Reallocates the given area of memory..

4.1.1 Detailed Description

Header file for malloc This file contains functions implementation from libs (malloc.h)

4.1.2 Function Documentation

4.1.2.1 simple_calloc()

```
void* simple_calloc (
            unsigned long num,
            unsigned long size )
```

Allocates memory for an array of num objects of size size and initializes it to all bits zero.

Parameters

num	Number of objects.
size	Size of each object.

Returns

On success, returns the pointer to the beginning of newly allocated memory.

4.1.2.2 simple_malloc()

```
void* simple_malloc (
            unsigned long size )
```

Allocates size bytes of uninitialized storage.

Parameters

size	Number of bytes to allocate.

Returns

On success, returns the pointer to the beginning of newly allocated memory.

if block is free and size is good -> get current block

allocate first or no available memory

get memory from os

4.1.2.3 simple_realoc()

```
void* simple_realoc (
            void * ptr,
            unsigned long size )
```

Reallocates the given area of memory..

Parameters

num	Number of objects.
ptr	Pointer to the memory area to be reallocated
new_size	New size of the array in bytes.

Returns

On success, returns the pointer to the beginning of newly allocated memory.

4.2 String

Header file for string This file contains functions implementation from libs (string.h)

Functions

- void ∗ str_memcpy (void ∗dest, const void ∗src, size_t n)

  Copies bytes between buffers. From src to dest.
- void ∗ str_memset (void ∗buf, char ch, size_t count)

  Sets buffers to a specified character.
- int str_cmp (const char ∗str1, const char ∗str2)

  Compare strings.
- char ∗ str_cat (char ∗dest, const char ∗src)

  Appends a string.
- size_t str_len (const char ∗str)

  Gets the length of a string.
- char ∗ str_cpy (char ∗dest, const char ∗src)

  Copies a string.
- char ∗ str_cpyn (char ∗dest, const char ∗src, size_t num)

  Copies the first num characters of source to destination.
- size_t str_spn (const char ∗str, char ∗accept)

  Returns the length of the initial portion of str1 which consists only of characters that are part of accept.
- size_t str_cspn (const char ∗str, char ∗not_accept)

  Scans str1 for the first occurrence of any of the characters that are part of str2, returning the number of characters of str1 read before this first occurrence.
- char ∗ str_ch (char ∗str, char ch)

  Returns a pointer to the first occurrence of character in the C string str.

4.2.1 Detailed Description

Header file for string This file contains functions implementation from libs (string.h)

4.2.2 Function Documentation

4.2.2.1 str_cat()

```
char∗ str_cat (
            char ∗ dest,
            const char ∗ src )
```

Appends a string.

Parameters

buf	Null-terminated destination string.
ch	Null-terminated source string.

Returns

destination is returned.

4.2.2.2 str_ch()

```
char* str_ch (
            char * str,
            char ch )
```

Returns a pointer to the first occurrence of character in the C string str.

Parameters

str	pointer to the object to fill.
ch	Character to be located. It is passed as its int promotion, but it is internally converted back to char for the comparison.

Returns

A pointer to the first occurrence of character in str.

4.2.2.3 str_cmp()

```
int str_cmp (
            const char * str1,
            const char * str2 )
```

Compare strings.

Parameters

str1	Null-terminated string to compare.
str2	Null-terminated string to compare.

Returns

{The return value for each of these functions indicates the ordinal relation of str1, str2.

$<$ 0 str1 is less than str2 0 str1 is identical to str2

0 str1 is greater than str2}

4.2.2.4 str_cpy()

```
char* str_cpy (
            char * dest,
            const char * src )
```

Copies a string.

Parameters

dest	Destination string.
src	Null-terminated source string.

Returns

destination is returned.

4.2.2.5 str_cpyn()

```
char* str_cpyn (
            char * dest,
            const char * src,
            size_t num )
```

Copies the first num characters of source to destination.

Parameters

dest	Destination string.
src	Null-terminated source string.
num	Maximum number of characters to be copied from source. size_t is an unsigned integral type.

Returns

destination is returned.

4.2.2.6 str_cspn()

```
size_t str_cspn (
            const char * str,
            char * not_accept )
```

Scans str1 for the first occurrence of any of the characters that are part of str2, returning the number of characters of str1 read before this first occurrence.

Parameters

str	Null-terminated source string to be scanned.
not_accept	Null-terminated source string containing the characters to match.

Returns

The length of the initial part of str not containing any of the characters that are part of not_accept.

4.2.2.7 str_len()

```
size_t str_len (
            const char * str )
```

Gets the length of a string.

Parameters

str	Null-terminated string.

Returns

length of a string.

4.2.2.8 str_memcpy()

```
void* str_memcpy (
            void * dest,
            const void * src,
            size_t n )
```

Copies bytes between buffers. From src to dest.

Parameters

dest	Destination area.
src	Source area.
n,Bytes	count.

Returns

Pointer to dest.

4.2.2.9 str_memset()

```
void* str_memset (
            void * buf,
            char ch,
            size_t count )
```

Sets buffers to a specified character.

Parameters

buf	Pointer to the object to fill.
ch	Fill byte.
count	Number of bytes to fill.

Returns

Pointer to buf.

4.2.2.10 str_spn()

```
size_t str_spn (
            const char * str,
            char * accept )
```

Returns the length of the initial portion of str1 which consists only of characters that are part of accept.

Parameters

str	Null-terminated string.
accept	Null-terminated string containing the characters to match.

Returns

The length of the initial portion of str1 containing only characters that appear in accept. size_t is an unsigned integral type.

Chapter 5

Class Documentation

5.1 memory_control_block Struct Reference

Memory block characteristic.

```
#include <memory.h>
```

Public Attributes

- unsigned long size

  Memory block size.
- short is_available

  1 is free else 0

5.1.1 Detailed Description

Memory block characteristic.

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

- include/memory.h

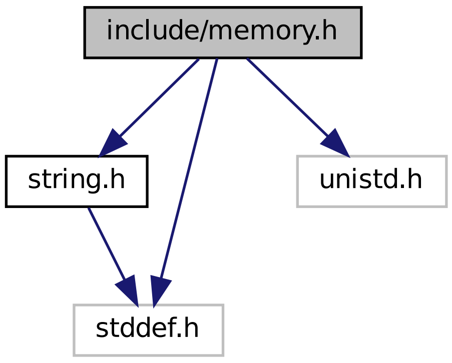
Chapter 6

File Documentation

6.1 include/memory.h File Reference

```
#include "string.h"
#include <stddef.h>
#include <unistd.h>
```

Include dependency graph for memory.h:



Classes

- struct memory_control_block

  Memory block characteristic.

Typedefs

- typedef struct memory_control_block mcb

  Memory block characteristic.

Functions

- void ∗ simple_malloc (unsigned long size)

  Allocates size bytes of uninitialized storage.
- void simple_free (void ∗ptr)

  Deallocates the space previously allocated by malloc(), calloc().
- void ∗ simple_calloc (unsigned long num, unsigned long size)

  Allocates memory for an array of num objects of size size and initializes it to all bits zero.
- void ∗ simple_realoc (void ∗ptr, unsigned long size)

  Reallocates the given area of memory..

6.1.1 Function Documentation

6.1.1.1 simple_free()

```
void simple_free (
            void * ptr )
```

Deallocates the space previously allocated by malloc(), calloc().

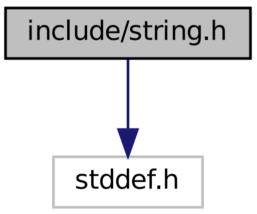
Parameters

ptr	Pointer to the memory to deallocate.

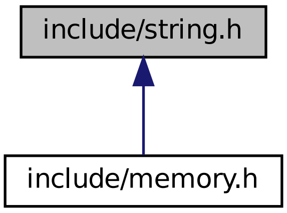
6.2 include/string.h File Reference

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

Include dependency graph for string.h:



This graph shows which files directly or indirectly include this file:



Functions

- void ∗ str_memcpy (void ∗dest, const void ∗src, size_t n)

  Copies bytes between buffers. From src to dest.
- void ∗ str_memset (void ∗buf, char ch, size_t count)

  Sets buffers to a specified character.
- int str_cmp (const char ∗str1, const char ∗str2)

  Compare strings.
- char ∗ str_cat (char ∗dest, const char ∗src)

  Appends a string.
- size_t str_len (const char ∗str)

  Gets the length of a string.
- char ∗ str_cpy (char ∗dest, const char ∗src)

  Copies a string.
- char ∗ str_cpyn (char ∗dest, const char ∗src, size_t num)

  Copies the first num characters of source to destination.
- size_t str_spn (const char ∗str, char ∗accept)

  Returns the length of the initial portion of str1 which consists only of characters that are part of accept.
- size_t str_cspn (const char ∗str, char ∗not_accept)

  Scans str1 for the first occurrence of any of the characters that are part of str2, returning the number of characters of str1 read before this first occurrence.
- char ∗ str_ch (char ∗str, char ch)

  Returns a pointer to the first occurrence of character in the C string str.

Index