INF558

Generated by Doxygen 1.8.17

Chapter 1

Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

Chapter 3

Data Structure Documentation

3.1 buffer_t Struct Reference

`#include <buffer.h>`

Data Fields

- uchar * tab
- size_t size
- size_t length

3.1.1 Field Documentation

3.1.1.1 length

`size_t buffer_t::length`

3.1.1.2 size

`size_t buffer_t::size`

3.1.1.3 tab

`uchar* buffer_t::tab`

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

- buffer.h

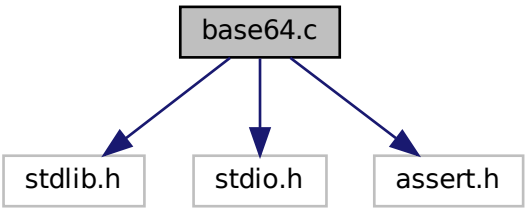
Chapter 4

File Documentation

4.1 base64.c File Reference

```
#include <stdlib.h>
#include <stdio.h>
#include <assert.h>
```

Include dependency graph for base64.c:



Typedefs

- typedef unsigned char uchar

Functions

- int CodeBase64 (uchar out[ ], const uchar in[ ], const int N)

  Code in[] in base 64: three characters of in[0..N[become four characters in out[]; if N is not a multiple of 3, we padd following the RFC.

- void DecodeBase64 (uchar ∗out, const uchar ∗in, const int N64)

  Decode in[0..N64[in base 64 with resultat in out[]. Assume that 4 | N64 and out[] has size >= 3∗N64/4. Returns the true length.

4.1.1 Detailed Description

Author

François Morain (morain@lix.polytechnique.fr)

4.1.2 Typedef Documentation

4.1.2.1 uchar

```
typedef unsigned char uchar
```

4.1.3 Function Documentation

4.1.3.1 CodeBase64()

```
int CodeBase64 (
            uchar out[],
            const uchar in[],
            const int N )
```

Code in[] in base 64: three characters of in[0..N[become four characters in out[]; if N is not a multiple of 3, we padd following the RFC.

Suppose that out has size >= 4∗N/3.

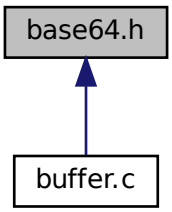
4.1.3.2 DecodeBase64()

```
void DecodeBase64 (
            uchar * out,
            const uchar * in,
            const int N64 )
```

Decode in[0..N64[in base 64 with resultat in out[]. Assume that 4 | N64 and out[] has size >= 3∗N64/4. Returns the true length.

4.2 base64.h File Reference

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



Typedefs

- typedef unsigned char uchar

Functions

- int CodeBase64 (uchar *out, const uchar *in, const int N)
- void DecodeBase64 (uchar *out, const uchar *in, const int N64)

  *Decode in[0..N64[in base 64 with resultat in out[]. Assume that 4 | N64 and out[] has size >= 3*N64/4. Returns the true length.*

4.2.1 Typedef Documentation

4.2.1.1 uchar

```
typedef unsigned char uchar
```

4.2.2 Function Documentation

4.2.2.1 CodeBase64()

```
int CodeBase64 (
            uchar * out,
            const uchar * in,
            const int N )
```

4.2.2.2 DecodeBase64()

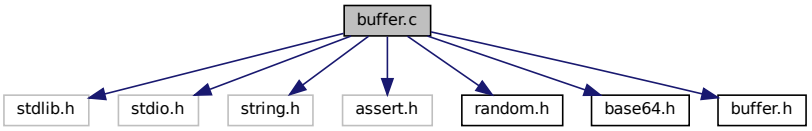
```
void DecodeBase64 (
            uchar * out,
            const uchar * in,
            const int N64 )
```

Decode in[0..N64[in base 64 with resultat in out[]. Assume that 4 | N64 and out[] has size >= 3∗N64/4. Returns the true length.

4.3 buffer.c File Reference

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <assert.h>
#include "random.h"
#include "base64.h"
#include "buffer.h"
```

Include dependency graph for buffer.c:



Macros

- #define DEBUG 0

  set DEBUG to 1 to have a complete equality test

Functions

- int buffer_init (buffer_t ∗buf, size_t size)

  initialize buf as an array of maximal size size.
- void buffer_clear (buffer_t ∗buf)

  Clears and frees the memory occuped by the buffer.
- int buffer_print (FILE ∗ofile, buffer_t ∗buf)

  Prints the content of buf to file descriptor ofile.
- int buffer_print_int (FILE ∗ofile, buffer_t ∗buf)

  Prints the content of buf as integers to file descriptor ofile.
- int buffer_resize (buffer_t ∗buf, size_t len)

  Reallocates buf->tab to size len.
- int buffer_append (buffer_t ∗buf, buffer_t ∗next)

  Appends the contents of next at the end of the contents of buf. If buf is too small, it is resized.

- int buffer_append_uchar (buffer_t *buf, uchar c)

  Appends an element at the end of the buffer.
- void buffer_reset (buffer_t *buf)

  Reset buffer to 0.
- int buffer_equality (buffer_t *buf1, buffer_t *buf2)

  Return 1 if buf1 == buf2 (same length, same content).
- void buffer_clone (buffer_t *out, buffer_t *in)

  Performs out <- in.
- int buffer_from_string (buffer_t *buf, uchar *str, size_t len)

  A string is an array of uchar's that ends with a '\0'. The '\0' is not put in the buffer. If the length is unknown give default value -1.
- uchar * string_from_buffer (buffer_t *buf)

  Creates a C string that contains buf.
- int buffer_from_file (buffer_t *buf, const char *file_name)

  Fills in buf from file if exists.
- void buffer_random (buffer_t *out, int byte_length)

  Fills in a buffer with byte_length random bytes Pseudo-random generator should have been intialised before.
- void buffer_to_base64 (buffer_t *out, buffer_t *in)
- void buffer_from_base64 (buffer_t *out, buffer_t *in)

4.3.1 Detailed Description

Author

François Morain (morain@lix.polytechnique.fr)

Alain Couvreur (alain.couvreur@lix.polytechnique.fr)

Date

September 29, 2018.

A buffer_t mimics an array of uchar of variable size. A buffer is created with a maximal *size*; each time an operation is performed, its size may be increased.

4.3.2 Macro Definition Documentation

4.3.2.1 DEBUG

```
#define DEBUG 0
```

set DEBUG to 1 to have a complete equality test

4.3.3 Function Documentation

4.3.3.1 buffer_append()

```
int buffer_append (
            buffer_t * buf,
            buffer_t * next )
```

Appends the contents of next at the end of the contents of buf. If buf is too small, it is resized.

4.3.3.2 buffer_append_uchar()

```
int buffer_append_uchar (
            buffer_t * buf,
            uchar c )
```

Appends an element at the end of the buffer.

4.3.3.3 buffer_clear()

```
void buffer_clear (
            buffer_t * buf )
```

Clears and frees the memory occuped by the buffer.

4.3.3.4 buffer_clone()

```
void buffer_clone (
            buffer_t * out,
            buffer_t * in )
```

Performs out <- in.

4.3.3.5 buffer_equality()

```
int buffer_equality (
            buffer_t * buf1,
            buffer_t * buf2 )
```

Return 1 if *buf1* == *buf2* (same length, same content).

4.3.3.6 buffer_from_base64()

```
void buffer_from_base64 (
            buffer_t * out,
            buffer_t * in )
```

4.3.3.7 buffer_from_file()

```
int buffer_from_file (
            buffer_t * buf,
            const char * file_name )
```

Fills in *buf* from *file* if exists.

4.3.3.8 buffer_from_string()

```
int buffer_from_string (
            buffer_t * buf,
            uchar * str,
            size_t len )
```

A string is an array of uchar's that ends with a '\0'. The '\0' is *not* put in the buffer. If the length is unknown give default value -1.

4.3.3.9 buffer_init()

```
int buffer_init (
            buffer_t * buf,
            size_t size )
```

initialize *buf* as an array of maximal size *size*.

4.3.3.10 buffer_print()

```
int buffer_print (
            FILE * ofile,
            buffer_t * buf )
```

Prints the content of *buf* to file descriptor *ofile*.

4.3.3.11 buffer_print_int()

```
int buffer_print_int (
            FILE * ofile,
            buffer_t * buf )
```

Prints the content of *buf* as integers to file descriptor *ofile*.

4.3.3.12 buffer_random()

```
void buffer_random (
            buffer_t * out,
            int byte_length )
```

Fills in a buffer with *byte_length* random bytes Pseudo-random generator should have been intialised before.

4.3.3.13 buffer_reset()

```
void buffer_reset (
            buffer_t * buf )
```

Reset buffer to 0.

4.3.3.14 buffer_resize()

```
int buffer_resize (
            buffer_t * buf,
            size_t len )
```

Reallocates *buf->tab* to size *len*.

4.3.3.15 buffer_to_base64()

```
void buffer_to_base64 (
            buffer_t * out,
            buffer_t * in )
```

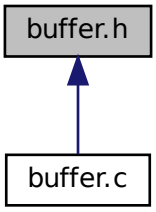
4.3.3.16 string_from_buffer()

```
uchar* string_from_buffer (
            buffer_t * buf )
```

Creates a C string that contains *buf*.

4.4 buffer.h File Reference

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



Data Structures

- struct buffer_t

Typedefs

- typedef unsigned char uchar

Functions

- int buffer_init (buffer_t *buf, size_t size)

  initialize buf as an array of maximal size size.
- void buffer_clear (buffer_t *buf)

  Clears and frees the memory occuped by the buffer.
- int buffer_print (FILE *ofile, buffer_t *buf)

  Prints the content of buf to file descriptor ofile.
- int buffer_print_int (FILE *ofile, buffer_t *buf)

  Prints the content of buf as integers to file descriptor ofile.
- int buffer_resize (buffer_t *buf, size_t len)

  Reallocates buf->tab to size len.
- int buffer_append (buffer_t *buf, buffer_t *next)

  Appends the contents of next at the end of the contents of buf. If buf is too small, it is resized.
- int buffer_append_uchar (buffer_t *buf, uchar c)

Appends an element at the end of the buffer.

- void buffer_reset (buffer_t *buf)

  Reset buffer to 0.
- int buffer_equality (buffer_t *buf1, buffer_t *buf2)

  Return 1 if buf1 == buf2 (same length, same content).
- void buffer_clone (buffer_t *out, buffer_t *in)

  Performs out <- in.
- int buffer_from_string (buffer_t *buf, uchar *str, size_t len)

  A string is an array of uchar's that ends with a '\0'. The '\0' is not put in the buffer. If the length is unknown give default value -1.
- uchar * string_from_buffer (buffer_t *buf)

  Creates a C string that contains buf.
- int buffer_from_file (buffer_t *buf, const char *file_name)

  Fills in buf from file if exists.
- void buffer_random (buffer_t *out, int byte_length)

  Fills in a buffer with byte_length random bytes Pseudo-random generator should have been intialised before.
- void buffer_to_base64 (buffer_t *out, buffer_t *in)
- void buffer_from_base64 (buffer_t *out, buffer_t *in)

4.4.1 Typedef Documentation

4.4.1.1 uchar

```
typedef unsigned char uchar
```

4.4.2 Function Documentation

4.4.2.1 buffer_append()

```
int buffer_append (
            buffer_t * buf,
            buffer_t * next )
```

Appends the contents of next at the end of the contents of buf. If buf is too small, it is resized.

4.4.2.2 buffer_append_uchar()

```
int buffer_append_uchar (
            buffer_t * buf,
            uchar c )
```

Appends an element at the end of the buffer.

4.4.2.3 buffer_clear()

```
void buffer_clear (
            buffer_t * buf )
```

Clears and frees the memory occuped by the buffer.

4.4.2.4 buffer_clone()

```
void buffer_clone (
            buffer_t * out,
            buffer_t * in )
```

Performs out <- in.

4.4.2.5 buffer_equality()

```
int buffer_equality (
            buffer_t * buf1,
            buffer_t * buf2 )
```

Return 1 if *buf1* == *buf2* (same length, same content).

4.4.2.6 buffer_from_base64()

```
void buffer_from_base64 (
            buffer_t * out,
            buffer_t * in )
```

4.4.2.7 buffer_from_file()

```
int buffer_from_file (
            buffer_t * buf,
            const char * file_name )
```

Fills in *buf* from *file* if exists.

4.4.2.8 buffer_from_string()

```
int buffer_from_string (
            buffer_t * buf,
            uchar * str,
            size_t len )
```

A string is an array of uchar's that ends with a '\0'. The '\0' is *not* put in the buffer. If the length is unknown give default value -1.

4.4.2.9 buffer_init()

```
int buffer_init (
            buffer_t * buf,
            size_t size )
```

initialize *buf* as an array of maximal size *size*.

4.4.2.10 buffer_print()

```
int buffer_print (
            FILE * ofile,
            buffer_t * buf )
```

Prints the content of *buf* to file descriptor *ofile*.

4.4.2.11 buffer_print_int()

```
int buffer_print_int (
            FILE * ofile,
            buffer_t * buf )
```

Prints the content of *buf* as integers to file descriptor *ofile*.

4.4.2.12 buffer_random()

```
void buffer_random (
            buffer_t * out,
            int byte_length )
```

Fills in a buffer with *byte_length* random bytes Pseudo-random generator should have been intialised before.

4.4.2.13 buffer_reset()

```
void buffer_reset (
            buffer_t * buf )
```

Reset buffer to 0.

4.4.2.14 buffer_resize()

```
int buffer_resize (
            buffer_t * buf,
            size_t len )
```

Reallocates *buf->tab* to size *len*.

4.4.2.15 buffer_to_base64()

```
void buffer_to_base64 (
            buffer_t * out,
            buffer_t * in )
```

4.4.2.16 string_from_buffer()

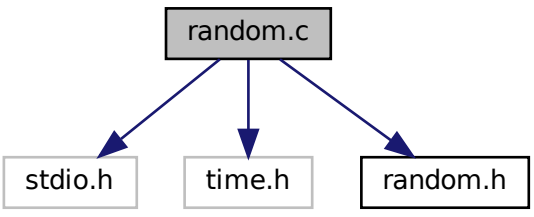
```
uchar* string_from_buffer (
            buffer_t * buf )
```

Creates a C string that contains *buf*.

4.5 random.c File Reference

```
#include <stdio.h>
#include <time.h>
#include "random.h"
```

Include dependency graph for random.c:



Functions

- unsigned int random_seed ()
  returns a random seed as random as it can be.

4.5.1 Detailed Description

Author

Alain Couvreur

4.5.2 Function Documentation

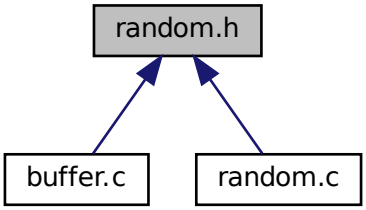
4.5.2.1 random_seed()

```
unsigned int random_seed ( )
```

returns a random seed as random as it can be.

4.6 random.h File Reference

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



Functions

- unsigned int random_seed ()
  returns a random seed as random as it can be.

4.6.1 Function Documentation

4.6.1.1 random_seed()

```
unsigned int random_seed ( )
```

returns a random seed as random as it can be.

Index