

TakoStrLib

Generated by Doxygen 1.8.17

1 Class Index	1
1.1 Class List	1
2 File Index	3
2.1 File List	3
3 Class Documentation	5
3.1 CharBuf Struct Reference	5
3.1.1 Detailed Description	5
3.1.2 Member Data Documentation	5
3.1.2.1 buf	5
3.1.2.2 size	6
3.2 StrArray Struct Reference	6
3.2.1 Detailed Description	6
3.2.2 Member Data Documentation	6
3.2.2.1 lines	6
3.2.2.2 size	6
4 File Documentation	7
4.1 /home/tako/programming/HWW/Libs/TSL/CharBuf/CharBuf.cc File Reference	7
4.1.1 Function Documentation	7
4.1.1.1 cb_init()	7
4.1.1.2 cb_destr()	8
4.1.1.3 sa_init()	8
4.1.1.4 sa_destr()	9
4.1.1.5 sa_print()	9
4.2 /home/tako/programming/HWW/Libs/TSL/CharBuf/CharBuf.hh File Reference	10
4.2.1 Typedef Documentation	10
4.2.1.1 String	10
4.2.2 Function Documentation	10
4.2.2.1 cb_init()	10
4.2.2.2 cb_destr()	11
4.2.2.3 sa_init()	11
4.2.2.4 sa_destr()	12
4.2.2.5 sa_print()	12
4.3 /home/tako/programming/HWW/Libs/TSL/tsl.cc File Reference	12
4.3.1 Detailed Description	13
4.3.2 Function Documentation	14
4.3.2.1 tsl_fputs()	14
4.3.2.2 tsl_puts()	14
4.3.2.3 tsl_strchr()	15
4.3.2.4 tsl_const_strchr()	15
4.3.2.5 tsl_strlen()	15

4.3.2.6	tsl_strcpy()	16
4.3.2.7	tsl_strncpy()	16
4.3.2.8	tsl_strcat()	17
4.3.2.9	tsl_strncat()	17
4.3.2.10	tsl_fgets()	18
4.3.2.11	tsl_strdup()	18
4.3.2.12	tsl_test()	19
4.3.2.13	tsl_split_lines()	19
4.3.2.14	tsl_cb_cmp()	20
4.3.2.15	tsl_cb_back_cmp()	20
4.4	/home/tako/programming/HWW/Libs/TSL/tsl.hh File Reference	21
4.4.1	Function Documentation	21
4.4.1.1	tsl_fputs()	22
4.4.1.2	tsl_puts()	23
4.4.1.3	tsl_strchr()	23
4.4.1.4	tsl_const_strchr()	24
4.4.1.5	tsl_strlen()	24
4.4.1.6	tsl_strcpy()	25
4.4.1.7	tsl_strncpy()	25
4.4.1.8	tsl_strcat()	26
4.4.1.9	tsl_strncat()	26
4.4.1.10	tsl_fgets()	27
4.4.1.11	tsl_strdup()	27
4.4.1.12	tsl_test()	28
4.4.1.13	tsl_split_lines()	28
4.4.1.14	tsl_cb_cmp()	28
4.4.1.15	tsl_cb_back_cmp()	29
	Index	31

Chapter 1

Class Index

1.1 Class List

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

CharBuf	Struct which keeps dynamic char array and it's size	5
StrArray	Struct to keep all strings	6

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

/home/tako/programming/HWW/Libs/TSL/ tsl.cc	
Tako's string library	12
/home/tako/programming/HWW/Libs/TSL/ tsl.hh	21
/home/tako/programming/HWW/Libs/TSL/CharBuf/ CharBuf.cc	7
/home/tako/programming/HWW/Libs/TSL/CharBuf/ CharBuf.hh	10

Chapter 3

Class Documentation

3.1 CharBuf Struct Reference

Struct which keeps dynamic char array and it's size.

```
#include <CharBuf.hh>
```

Public Attributes

- char * [buf](#)
- size_t [size](#)

3.1.1 Detailed Description

Struct which keeps dynamic char array and it's size.

Definition at line 10 of file CharBuf.hh.

3.1.2 Member Data Documentation

3.1.2.1 buf

```
char* CharBuf::buf
```

Definition at line 12 of file CharBuf.hh.

Referenced by [cb_destr\(\)](#), [cb_init\(\)](#), [sa_print\(\)](#), [tsl_cb_back_cmp\(\)](#), [tsl_cb_cmp\(\)](#), and [tsl_split_lines\(\)](#).

3.1.2.2 size

```
size_t CharBuf::size
```

Definition at line 13 of file CharBuf.hh.

Referenced by `cb_destr()`, `cb_init()`, `sa_print()`, `tsl_cb_back_cmp()`, and `tsl_cb_cmp()`.

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

- [/home/tako/programming/HWW/Libs/TSL/CharBuf/CharBuf.hh](#)

3.2 StrArray Struct Reference

Struct to keep all strings.

```
#include <CharBuf.hh>
```

Public Attributes

- [String * lines](#)
- [size_t size](#)

3.2.1 Detailed Description

Struct to keep all strings.

Definition at line 24 of file CharBuf.hh.

3.2.2 Member Data Documentation

3.2.2.1 lines

```
String* StrArray::lines
```

Definition at line 26 of file CharBuf.hh.

Referenced by `sa_destr()`, `sa_init()`, and `sa_print()`.

3.2.2.2 size

```
size_t StrArray::size
```

Definition at line 27 of file CharBuf.hh.

Referenced by `sa_destr()`, `sa_init()`, `sa_print()`, and `tsl_split_lines()`.

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

- [/home/tako/programming/HWW/Libs/TSL/CharBuf/CharBuf.hh](#)

Chapter 4

File Documentation

4.1 /home/tako/programming/HWW/Libs/TSL/CharBuf/CharBuf.cc File Reference

```
#include <stdio.h>
#include "CharBuf.hh"
```

Functions

- `CharBuf * cb_init (CharBuf *cb, size_t elem_num)`
CharBuf constructor.
- `CharBuf * cb_destr (CharBuf *cb)`
CharBuf destructor.
- `StrArray * sa_init (StrArray *sa, size_t elnum)`
StrArray constructor.
- `StrArray * sa_destr (StrArray *sa)`
StrArray destructor.
- `void sa_print (StrArray sa, FILE *fp)`
prints string array

4.1.1 Function Documentation

4.1.1.1 cb_init()

```
CharBuf* cb_init (
    CharBuf * cb,
    size_t elem_num )
```

`CharBuf` constructor.

Parameters

out	<i>cb</i>	CharBuf object
in	<i>elem_num</i>	number of elements in buffer

Returns

pointer to input [CharBuf](#) object

Definition at line 5 of file CharBuf.cc.

References [CharBuf::buf](#), and [CharBuf::size](#).

4.1.1.2 [cb_destr\(\)](#)

```
CharBuf* cb_destr (
    CharBuf * cb )
```

[CharBuf](#) destructor.

Parameters

out	<i>cb</i>	CharBuf object
-----	-----------	--------------------------------

Returns

pointer to input [CharBuf](#) object

Definition at line 15 of file CharBuf.cc.

References [CharBuf::buf](#), and [CharBuf::size](#).

4.1.1.3 [sa_init\(\)](#)

```
StrArray* sa_init (
    StrArray * sa,
    size_t elnum )
```

[StrArray](#) constructor.

Parameters

out	<i>sa</i>	StrArray object
in	<i>elnum</i>	number of lines

Returns

pointer to input object

Definition at line 25 of file CharBuf.cc.

References StrArray::lines, and StrArray::size.

4.1.1.4 sa_destr()

```
StrArray* sa_destr (
    StrArray * sa )
```

StrArray destructor.

Parameters

out	sa	StrArray object
-----	----	-----------------

Returns

pointer to input object

Definition at line 35 of file CharBuf.cc.

References StrArray::lines, and StrArray::size.

4.1.1.5 sa_print()

```
void sa_print (
    StrArray sa,
    FILE * fp = stdout )
```

prints string array

Parameters

in	sa	
----	----	--

Definition at line 45 of file CharBuf.cc.

References CharBuf::buf, StrArray::lines, CharBuf::size, and StrArray::size.

4.2 /home/tako/programming/HWW/Libs/TSL/CharBuf/CharBuf.hh File Reference

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

Classes

- struct [CharBuf](#)
Struct which keeps dynamic char array and it's size.
- struct [StrArray](#)
Struct to keep all strings.

Typedefs

- typedef [CharBuf](#) String
Alias for [CharBuf](#) struct.

Functions

- [CharBuf](#) * [cb_init](#) ([CharBuf](#) *cb, size_t elem_num)
CharBuf constructor.
- [CharBuf](#) * [cb_destr](#) ([CharBuf](#) *cb)
CharBuf destructor.
- [StrArray](#) * [sa_init](#) ([StrArray](#) *sa, size_t elnum)
StrArray constructor.
- [StrArray](#) * [sa_destr](#) ([StrArray](#) *sa)
StrArray destructor.
- void [sa_print](#) ([StrArray](#) sa, FILE *fp=stdout)
prints string array

4.2.1 Typedef Documentation

4.2.1.1 String

```
typedef CharBuf String
```

Alias for [CharBuf](#) struct.

Definition at line 19 of file CharBuf.hh.

4.2.2 Function Documentation

4.2.2.1 cb_init()

```
CharBuf* cb_init (
    CharBuf * cb,
    size_t elem_num )
```

[CharBuf](#) constructor.

Parameters

out	<i>cb</i>	CharBuf object
in	<i>elem_num</i>	number of elements in buffer

Returns

pointer to input [CharBuf](#) object

Definition at line 5 of file CharBuf.cc.

References [CharBuf::buf](#), and [CharBuf::size](#).

4.2.2.2 [cb_destr\(\)](#)

```
CharBuf* cb_destr (  
    CharBuf * cb )
```

[CharBuf](#) destructor.

Parameters

out	<i>cb</i>	CharBuf object
-----	-----------	--------------------------------

Returns

pointer to input [CharBuf](#) object

Definition at line 15 of file CharBuf.cc.

References [CharBuf::buf](#), and [CharBuf::size](#).

4.2.2.3 [sa_init\(\)](#)

```
StrArray* sa_init (  
    StrArray * sa,  
    size_t elnum )
```

[StrArray](#) constructor.

Parameters

out	<i>sa</i>	StrArray object
in	<i>elnum</i>	number of lines

Returns

pointer to input object

Definition at line 25 of file CharBuf.cc.

References StrArray::lines, and StrArray::size.

4.2.2.4 sa_destr()

```
StrArray* sa_destr (
    StrArray * sa )
```

StrArray destructor.

Parameters

out	sa	StrArray object
-----	----	-----------------

Returns

pointer to input object

Definition at line 35 of file CharBuf.cc.

References StrArray::lines, and StrArray::size.

4.2.2.5 sa_print()

```
void sa_print (
    StrArray sa,
    FILE * fp = stdout )
```

prints string array

Parameters

in	sa	
----	----	--

Definition at line 45 of file CharBuf.cc.

References CharBuf::buf, StrArray::lines, CharBuf::size, and StrArray::size.

4.3 /home/tako/programming/HWW/Libs/TSL/tsl.cc File Reference

Tako's string library.


```
#include <ctype.h>
#include <stdio.h>
#include <stdlib.h>
#include "trace.hh"
#include "tsl.hh"
```

Functions

- int [tsl_fputs](#) (const char *str, FILE *stream)
writes the string str to stream, without terminating null byte ('\0').
- int [tsl_puts](#) (const char *str)
writes the string str and a trailing newline to stdout.
- char * [tsl_strchr](#) (char *str, int ch)
returns a pointer to the first occurrence of the character c in the string str.
- const char * [tsl_const_strchr](#) (const char *str, int ch)
returns a pointer to the first occurrence of the character c in the string s.
- size_t [tsl_strlen](#) (const char *str)
calculates the length of the string pointed to by str, excluding the terminating null byte ('\0').
- char * [tsl_strcpy](#) (char *dst, const char *src)
copies the string pointed to by src, including the terminating null byte ('\0'), to the buffer pointed to by dest.
- char * [tsl_strncpy](#) (char *dst, const char *src, size_t n)
similar to [tsl_strcpy](#), except that at most n bytes of src are copied
- char * [tsl_strcat](#) (char *dst, const char *src)
appends the src string to the dest string, overwriting the terminating null byte ('\0') at the end of dest, and then adds a terminating null byte.
- char * [tsl_strncat](#) (char *dst, const char *src, size_t n)
is similar to [tsl_strcat](#), except that it will use at most n bytes from src and src does not need to be null-terminated if it contains n or more bytes.
- char * [tsl_fgets](#) (char *str, int size, FILE *stream)
reads in at most one less than size characters from stream and stores them into the buffer pointed to by str.
- char * [tsl_strdup](#) (const char *str)
returns a pointer to a new string which is a duplicate of the string str.
- int [tsl_test](#) ()
unit test for tsl functions
- [StrArray](#) [tsl_split_lines](#) ([CharBuf](#) raw)
split input buffer to lines, allocates memory in [StrArray::lines](#).
- int [tsl_cb_cmp](#) (const void *lhs, const void *rhs)
comparator for [CharBuf](#) strings
- int [tsl_cb_back_cmp](#) (const void *lhs, const void *rhs)
backward comparator for [CharBuf](#) strings

4.3.1 Detailed Description

Tako's string library.

Author

Tako

4.3.2 Function Documentation

4.3.2.1 `tsl_fputs()`

```
int tsl_fputs (
    const char * str,
    FILE * stream )
```

writes the string *str* to *stream*, without terminating null byte ('\0').

Parameters

in	<i>str</i>	
in	<i>stream</i>	

Returns

nonnegative number on success, or EOF on error.

Definition at line 14 of file `tsl.cc`.

Referenced by `tsl_puts()`.

4.3.2.2 `tsl_puts()`

```
int tsl_puts (
    const char * str )
```

writes the string *str* and a trailing newline to stdout.

Parameters

in	<i>str</i>	
----	------------	--

Returns

a nonnegative number on success, or EOF on error.

Definition at line 26 of file `tsl.cc`.

References `tsl_fputs()`.

Referenced by `tsl_test()`.

4.3.2.3 `tsl_strchr()`

```
char* tsl_strchr (  
    char * str,  
    int ch )
```

returns a pointer to the first occurrence of the character *c* in the string *str*.

Parameters

in	<i>str</i>	
in	<i>ch</i>	

Returns

pointer to the matched character or NULL if the character is not found.

Definition at line 31 of file `tsl.cc`.

4.3.2.4 `tsl_const_strchr()`

```
const char* tsl_const_strchr (  
    const char * str,  
    int ch )
```

returns a pointer to the first occurrence of the character *c* in the string *s*.

Parameters

in	<i>str</i>	
in	<i>ch</i>	

Returns

pointer to the matched character or NULL if the character is not found.

Definition at line 42 of file `tsl.cc`.

Referenced by `tsl_test()`.

4.3.2.5 `tsl_strlen()`

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

calculates the length of the string pointed to by *str*, excluding the terminating null byte (`'\0'`).

Parameters

in	<i>str</i>	
----	------------	--

Returns

the number of bytes in the string pointed to by s.

Definition at line 53 of file `tsl.cc`.

Referenced by `tsl_strdup()`, and `tsl_test()`.

4.3.2.6 `tsl_strcpy()`

```
char* tsl_strcpy (  
    char * dst,  
    const char * src )
```

copies the string pointed to by *src*, including the terminating null byte ('0'), to the buffer pointed to by *dest*.

The strings may not overlap, and the destination string *dest* must be large enough to receive the copy.

Parameters

out	<i>dst</i>	
in	<i>src</i>	

Returns

pointer to the destination string.

Definition at line 65 of file `tsl.cc`.

Referenced by `tsl_strcat()`, `tsl_strdup()`, `tsl_strncat()`, and `tsl_test()`.

4.3.2.7 `tsl_strncpy()`

```
char* tsl_strncpy (  
    char * dst,  
    const char * src,  
    size_t n )
```

similar to `tsl_strcpy`, except that at most *n* bytes of *src* are copied

Parameters

out	<i>dst</i>	
in	<i>src</i>	
in	<i>n</i>	

Returns

pointer to the destination string.

Definition at line 79 of file tsl.cc.

Referenced by `tsl_test()`.

4.3.2.8 `tsl_strcat()`

```
char* tsl_strcat (  
    char * dst,  
    const char * src )
```

appends the *src* string to the *dest* string, overwriting the terminating null byte ('\0') at the end of *dest*, and then adds a terminating null byte.

Parameters

out	<i>dst</i>	
in	<i>src</i>	

Returns

pointer to the resulting string.

Definition at line 95 of file tsl.cc.

References `tsl_strcpy()`.

4.3.2.9 `tsl_strncat()`

```
char* tsl_strncat (  
    char * dst,  
    const char * src,  
    size_t n )
```

is similar to `tsl_strcat`, except that it will use at most *n* bytes from *src* and *src* does not need to be null-terminated if it contains *n* or more bytes.

Parameters

out	<i>dst</i>	
in	<i>src</i>	
in	<i>n</i>	

Returns

pointer to the resulting string.

Definition at line 107 of file `tsl.cc`.

References `tsl_strcpy()`.

Referenced by `tsl_test()`.

4.3.2.10 `tsl_fgets()`

```
char* tsl_fgets (  
    char * str,  
    int size,  
    FILE * stream )
```

reads in at most one less than `size` characters from `stream` and stores them into the buffer pointed to by `str`.

Reading stops after an EOF or a newline. If a newline is read, it is stored into the buffer. A terminating null byte ('\0') is stored after the last character in the buffer.

Parameters

out	<i>str</i>	
in	<i>size</i>	
in	<i>stream</i>	

Returns

`str` on success, and `nullptr` on error or when end of file occurs while no characters have been read.

Definition at line 119 of file `tsl.cc`.

Referenced by `tsl_test()`.

4.3.2.11 `tsl_strdup()`

```
char* tsl_strdup (  
    const char * str )
```

returns a pointer to a new string which is a duplicate of the string `str`.

Memory for the new string is obtained with `malloc`, and can be freed with `free`.

Parameters

<i>in</i>	<i>str</i>	
-----------	------------	--

Returns

pointer to the duplicated string or nullptr in case of failure.

Definition at line 147 of file tsl.cc.

References `tsl_strcpy()`, and `tsl_strlen()`.

Referenced by `tsl_test()`.

4.3.2.12 `tsl_test()`

```
int tsl_test ( )
```

unit test for tsl functions

Returns

0 on success

Definition at line 158 of file tsl.cc.

References `tsl_const_strchr()`, `tsl_fgets()`, `tsl_puts()`, `tsl_strcpy()`, `tsl_strdup()`, `tsl_strlen()`, `tsl_strncat()`, and `tsl_strncpy()`.

4.3.2.13 `tsl_split_lines()`

```
StrArray tsl_split_lines (
    CharBuf raw )
```

split input buffer to lines, allocates memory in [StrArray::lines](#).

Do not forget to free

Parameters

<i>in</i>	<i>raw</i>	input char buffer
-----------	------------	-------------------

Returns

[StrArray](#)

Definition at line 194 of file `tsl.cc`.

References `CharBuf::buf`, and `StrArray::size`.

4.3.2.14 `tsl_cb_cmp()`

```
int tsl_cb_cmp (
    const void * lhs,
    const void * rhs )
```

comparator for [CharBuf](#) strings

Parameters

in	<i>lhs</i>	
in	<i>rhs</i>	

Returns

`== 0` if elements are equal
`> 0` if `lhs > rhs`
`< 0` if `lhs < rhs`

Definition at line 241 of file `tsl.cc`.

References `CharBuf::buf`, and `CharBuf::size`.

4.3.2.15 `tsl_cb_back_cmp()`

```
int tsl_cb_back_cmp (
    const void * lhs,
    const void * rhs )
```

backward comparator for [CharBuf](#) strings

Parameters

in	<i>lhs</i>	
in	<i>rhs</i>	

Returns

`== 0` if elements are equal
`> 0` if `lhs > rhs`
`< 0` if `lhs < rhs`

Definition at line 267 of file tsl.cc.

References CharBuf::buf, and CharBuf::size.

4.4 /home/tako/programming/HWW/Libs/TSL/tsl.hh File Reference

```
#include "CharBuf.hh"
```

Functions

- int [tsl_fputs](#) (const char *str, FILE *stream)
writes the string str to stream, without terminating null byte ('\0').
- int [tsl_puts](#) (const char *str)
writes the string str and a trailing newline to stdout.
- char * [tsl_strchr](#) (char *str, int ch)
returns a pointer to the first occurrence of the character c in the string str.
- const char * [tsl_const_strchr](#) (const char *str, int ch)
returns a pointer to the first occurrence of the character c in the string s.
- size_t [tsl_strlen](#) (const char *str)
calculates the length of the string pointed to by str, excluding the terminating null byte ('\0').
- char * [tsl_strcpy](#) (char *dst, const char *src)
copies the string pointed to by src, including the terminating null byte ('\0'), to the buffer pointed to by dest.
- char * [tsl_strncpy](#) (char *dst, const char *src, size_t n)
similar to [tsl_strcpy](#), except that at most n bytes of src are copied
- char * [tsl_strcat](#) (char *dst, const char *src)
appends the src string to the dest string, overwriting the terminating null byte ('\0') at the end of dest, and then adds a terminating null byte.
- char * [tsl_strncat](#) (char *dst, const char *src, size_t n)
is similar to [tsl_strcat](#), except that it will use at most n bytes from src and src does not need to be null-terminated if it contains n or more bytes.
- char * [tsl_fgets](#) (char *str, int size, FILE *stream)
reads in at most one less than size characters from stream and stores them into the buffer pointed to by str.
- char * [tsl_strdup](#) (const char *str)
returns a pointer to a new string which is a duplicate of the string str.
- int [tsl_test](#) ()
unit test for tsl functions
- [StrArray](#) [tsl_split_lines](#) ([CharBuf](#) raw)
split input buffer to lines, allocates memory in [StrArray::lines](#).
- int [tsl_cb_cmp](#) (const void *lhs, const void *rhs)
comparator for [CharBuf](#) strings
- int [tsl_cb_back_cmp](#) (const void *lhs, const void *rhs)
backward comparator for [CharBuf](#) strings

4.4.1 Function Documentation

4.4.1.1 `tsl_fputs()`

```
int tsl_fputs (
    const char * str,
    FILE * stream )
```

writes the string `str` to `stream`, without terminating null byte (`'\0'`).

Parameters

in	<i>str</i>	
in	<i>stream</i>	

Returns

nonnegative number on success, or EOF on error.

Definition at line 14 of file tsl.cc.

Referenced by `tsl_puts()`.

4.4.1.2 `tsl_puts()`

```
int tsl_puts (  
    const char * str )
```

writes the string `str` and a trailing newline to `stdout`.

Parameters

in	<i>str</i>	
----	------------	--

Returns

a nonnegative number on success, or EOF on error.

Definition at line 26 of file tsl.cc.

References `tsl_fputs()`.

Referenced by `tsl_test()`.

4.4.1.3 `tsl_strchr()`

```
char* tsl_strchr (  
    char * str,  
    int ch )
```

returns a pointer to the first occurrence of the character `c` in the string `str`.

Parameters

in	<i>str</i>	
in	<i>ch</i>	

Returns

pointer to the matched character or NULL if the character is not found.

Definition at line 31 of file `tsl.cc`.

4.4.1.4 `tsl_const_strchr()`

```
const char* tsl_const_strchr (  
    const char * str,  
    int ch )
```

returns a pointer to the first occurrence of the character `c` in the string `s`.

Parameters

in	<i>str</i>	
in	<i>ch</i>	

Returns

pointer to the matched character or NULL if the character is not found.

Definition at line 42 of file `tsl.cc`.

Referenced by `tsl_test()`.

4.4.1.5 `tsl_strlen()`

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

calculates the length of the string pointed to by `str`, excluding the terminating null byte (`'\0'`).

Parameters

in	<i>str</i>	
----	------------	--

Returns

the number of bytes in the string pointed to by `s`.

Definition at line 53 of file `tsl.cc`.

Referenced by `tsl_strdup()`, and `tsl_test()`.

4.4.1.6 `tsl_strcpy()`

```
char* tsl_strcpy (
    char * dst,
    const char * src )
```

copies the string pointed to by *src*, including the terminating null byte ('\0'), to the buffer pointed to by *dest*.

The strings may not overlap, and the destination string *dest* must be large enough to receive the copy.

Parameters

out	<i>dst</i>	
in	<i>src</i>	

Returns

pointer to the destination string.

Definition at line 65 of file `tsl.cc`.

Referenced by `tsl_strcat()`, `tsl_strdup()`, `tsl_strncat()`, and `tsl_test()`.

4.4.1.7 `tsl_strncpy()`

```
char* tsl_strncpy (
    char * dst,
    const char * src,
    size_t n )
```

similar to `tsl_strcpy`, except that at most *n* bytes of *src* are copied

Parameters

out	<i>dst</i>	
in	<i>src</i>	
in	<i>n</i>	

Returns

pointer to the destination string.

Definition at line 79 of file `tsl.cc`.

Referenced by `tsl_test()`.

4.4.1.8 `tsl_strcat()`

```
char* tsl_strcat (
    char * dst,
    const char * src )
```

appends the *src* string to the *dest* string, overwriting the terminating null byte ('\0') at the end of *dest*, and then adds a terminating null byte.

Parameters

out	<i>dst</i>	
in	<i>src</i>	

Returns

pointer to the resulting string.

Definition at line 95 of file `tsl.cc`.

References `tsl_strcpy()`.

4.4.1.9 `tsl_strncat()`

```
char* tsl_strncat (
    char * dst,
    const char * src,
    size_t n )
```

is similar to `tsl_strcat`, except that it will use at most *n* bytes from *src* and *src* does not need to be null-terminated if it contains *n* or more bytes.

Parameters

out	<i>dst</i>	
in	<i>src</i>	
in	<i>n</i>	

Returns

pointer to the resulting string.

Definition at line 107 of file `tsl.cc`.

References `tsl_strcpy()`.

Referenced by `tsl_test()`.

4.4.1.10 `tsl_fgets()`

```
char* tsl_fgets (
    char * str,
    int size,
    FILE * stream )
```

reads in at most one less than *size* characters from *stream* and stores them into the buffer pointed to by *str*.

Reading stops after an EOF or a newline. If a newline is read, it is stored into the buffer. A terminating null byte ('\0') is stored after the last character in the buffer.

Parameters

out	<i>str</i>	
in	<i>size</i>	
in	<i>stream</i>	

Returns

str on success, and `nullptr` on error or when end of file occurs while no characters have been read.

Definition at line 119 of file `tsl.cc`.

Referenced by `tsl_test()`.

4.4.1.11 `tsl_strdup()`

```
char* tsl_strdup (
    const char * str )
```

returns a pointer to a new string which is a duplicate of the string *str*.

Memory for the new string is obtained with `malloc`, and can be freed with `free`.

Parameters

in	<i>str</i>	
----	------------	--

Returns

pointer to the duplicated string or `nullptr` in case of failure.

Definition at line 147 of file `tsl.cc`.

References `tsl_strcpy()`, and `tsl_strlen()`.

Referenced by `tsl_test()`.

4.4.1.12 `ttl_test()`

```
int ttl_test ( )
```

unit test for ttl functions

Returns

0 on success

Definition at line 158 of file ttl.cc.

References `ttl_const_strchr()`, `ttl_fgets()`, `ttl_puts()`, `ttl_strcpy()`, `ttl_strdup()`, `ttl_strlen()`, `ttl_strncat()`, and `ttl_strncpy()`.

4.4.1.13 `ttl_split_lines()`

```
StrArray ttl_split_lines (
    CharBuf raw )
```

split input buffer to lines, allocates memory in [StrArray::lines](#).

Do not forget to free

Parameters

in	<i>raw</i>	input char buffer
----	------------	-------------------

Returns

[StrArray](#)

Definition at line 194 of file ttl.cc.

References `CharBuf::buf`, and `StrArray::size`.

4.4.1.14 `ttl_cb_cmp()`

```
int ttl_cb_cmp (
    const void * lhs,
    const void * rhs )
```

comparator for [CharBuf](#) strings

Parameters

in	<i>lhs</i>	
in	<i>rhs</i>	

Returns

== 0 if elements are equal
> 0 if lhs > rhs
< 0 if lhs < rhs

Definition at line 241 of file tsl.cc.

References CharBuf::buf, and CharBuf::size.

4.4.1.15 tsl_cb_back_cmp()

```
int tsl_cb_back_cmp (  
    const void * lhs,  
    const void * rhs )
```

backward comparator for [CharBuf](#) strings

Parameters

in	<i>lhs</i>	
in	<i>rhs</i>	

Returns

== 0 if elements are equal
> 0 if lhs > rhs
< 0 if lhs < rhs

Definition at line 267 of file tsl.cc.

References CharBuf::buf, and CharBuf::size.

Index

[/home/tako/programming/HWWH/Libs/TSL/CharBuf/CharBuf.c](#)[size, 6](#)
[7](#) [String](#)
[/home/tako/programming/HWWH/Libs/TSL/CharBuf/CharBuf.h](#)[CharBuf.h, 10](#)
[10](#)

[/home/tako/programming/HWWH/Libs/TSL/tsl.c](#), [12](#) [tsl.c](#)
[/home/tako/programming/HWWH/Libs/TSL/tsl.h](#), [21](#) [tsl.cc](#)

[buf](#)
 [CharBuf](#), [5](#)

[cb_destr](#)
 [CharBuf.c](#), [8](#)
 [CharBuf.h](#), [11](#)

[cb_init](#)
 [CharBuf.c](#), [7](#)
 [CharBuf.h](#), [10](#)

[CharBuf](#), [5](#)
 [buf](#), [5](#)
 [size](#), [5](#)

[CharBuf.c](#)
 [cb_destr](#), [8](#)
 [cb_init](#), [7](#)
 [sa_destr](#), [9](#)
 [sa_init](#), [8](#)
 [sa_print](#), [9](#)

[CharBuf.h](#)
 [cb_destr](#), [11](#)
 [cb_init](#), [10](#)
 [sa_destr](#), [12](#)
 [sa_init](#), [11](#)
 [sa_print](#), [12](#)
 [String](#), [10](#)

[lines](#)
 [StrArray](#), [6](#)

[sa_destr](#)
 [CharBuf.c](#), [9](#)
 [CharBuf.h](#), [12](#)

[sa_init](#)
 [CharBuf.c](#), [8](#)
 [CharBuf.h](#), [11](#)

[sa_print](#)
 [CharBuf.c](#), [9](#)
 [CharBuf.h](#), [12](#)

[size](#)
 [CharBuf](#), [5](#)
 [StrArray](#), [6](#)

[StrArray](#), [6](#)
 [lines](#), [6](#)

[tsl_cb_back_cmp](#), [20](#)
 [tsl_cb_cmp](#), [20](#)
 [tsl_const_strchr](#), [15](#)
 [tsl_fgets](#), [18](#)
 [tsl_fputs](#), [14](#)
 [tsl_puts](#), [14](#)
 [tsl_split_lines](#), [19](#)
 [tsl_strcat](#), [17](#)
 [tsl_strchr](#), [14](#)
 [tsl_strcpy](#), [16](#)
 [tsl_strdup](#), [18](#)
 [tsl_strlen](#), [15](#)
 [tsl_strncat](#), [17](#)
 [tsl_strncpy](#), [16](#)
 [tsl_test](#), [19](#)

[tsl.h](#)
 [tsl_cb_back_cmp](#), [29](#)
 [tsl_cb_cmp](#), [28](#)
 [tsl_const_strchr](#), [24](#)
 [tsl_fgets](#), [26](#)
 [tsl_fputs](#), [21](#)
 [tsl_puts](#), [23](#)
 [tsl_split_lines](#), [28](#)
 [tsl_strcat](#), [25](#)
 [tsl_strchr](#), [23](#)
 [tsl_strcpy](#), [24](#)
 [tsl_strdup](#), [27](#)
 [tsl_strlen](#), [24](#)
 [tsl_strncat](#), [26](#)
 [tsl_strncpy](#), [25](#)
 [tsl_test](#), [27](#)

[tsl_cb_back_cmp](#)
 [tsl.c](#), [20](#)
 [tsl.h](#), [29](#)

[tsl_cb_cmp](#)
 [tsl.c](#), [20](#)
 [tsl.h](#), [28](#)

[tsl_const_strchr](#)
 [tsl.c](#), [15](#)
 [tsl.h](#), [24](#)

[tsl_fgets](#)
 [tsl.c](#), [18](#)
 [tsl.h](#), [26](#)

[tsl_fputs](#)
 [tsl.c](#), [14](#)

- [tsl.hh](#), [21](#)
- [tsl_puts](#)
 - [tsl.cc](#), [14](#)
 - [tsl.hh](#), [23](#)
- [tsl_split_lines](#)
 - [tsl.cc](#), [19](#)
 - [tsl.hh](#), [28](#)
- [tsl_strcat](#)
 - [tsl.cc](#), [17](#)
 - [tsl.hh](#), [25](#)
- [tsl_strchr](#)
 - [tsl.cc](#), [14](#)
 - [tsl.hh](#), [23](#)
- [tsl_strcpy](#)
 - [tsl.cc](#), [16](#)
 - [tsl.hh](#), [24](#)
- [tsl_strdup](#)
 - [tsl.cc](#), [18](#)
 - [tsl.hh](#), [27](#)
- [tsl_strlen](#)
 - [tsl.cc](#), [15](#)
 - [tsl.hh](#), [24](#)
- [tsl_strncat](#)
 - [tsl.cc](#), [17](#)
 - [tsl.hh](#), [26](#)
- [tsl_strncpy](#)
 - [tsl.cc](#), [16](#)
 - [tsl.hh](#), [25](#)
- [tsl_test](#)
 - [tsl.cc](#), [19](#)
 - [tsl.hh](#), [27](#)