SDTP Documentation – String-Tools

This is the documentation of the C API of the SDTP String-Tools library, which has been developed to extend the standard string and char functions by several other functions, which were found to be useful within SDTP, and are generally enough to be potentially useful in other projects. The name of the header file is `string_tools.h`.

Functions for checking certain groups of chars

Similar to the standard functions in the `ctype.h` header, these functions are used to check if a char is a member of a certain group of chars. These are two functions which extend the test for alphabetical or alphanumerical char to also allowing underscores, which is commonly allowed in SDTP as a char in variable naming or other symbols.

Both functions return 0 if the function is not satisfied, or `1`, if it is. (ATTENTION: Unlike these functions, the `ctype.h` are only specified to return a value unequal to `0` although *de facto* all implementations return `1`)

int isalpha_uscore (**int** c) – Checks, if char `c` is alphabetic or underscores

int isalnum_uscore (**int** c) – Checks, if char `c` is alphanumeric or underscore

Count number of char occurrences in a string

This function returns the number of occurrences of the char `c` in string `s`:

size_t chrn (**char*** s, **char** c)

The absolute number of the chars occurrences is returned. If the char doesn‘t appear at all, the function will return `0`. (NOTICE: To count letters case-insensitive, one may use a copy created with `strtocfn`, where consistently uppercase or lowercase is in use; As an alternative, it is possible to call the function for both the upper- and lower case letter and add the two results.)

Remove white space

Using `wwipe`, a modified copy of `src` is copied to `dest`, where all non-graphical chars according to the `ctype.h` function **int** isgraph(**int**) outside of double quotation marks are removed:

void wwipe (**char*** dest, **char*** src)

In SDTP, this function is used for example to remove white space before parsing the file of a steganographic generator, which is present for better human readability. Spaces withing double quotation marks are, however, preserved, as they are used to express spacing in sentences or separated-written words.

Application of char transformation on entire string (especially useful for setting string upper- or lowercase)

Beneath the destination memory pointer `dest` and the input string `src` this function expects a function as an argument, which takes an integer and returns an integer. This is the prototype for a general function which takes a character, transforms it according to specified rules, and returns it.

```
void strtocfn (char* dest, char* src, int(*tfn)(int))
```

Although many use cases are possible, the function has especially been developed for transforming entire string to upper- or lowercase using the standardized functions `tolower` or `toupper`. It is of advantage, that those chars, which don‘t have to be modified at all (digits, punctuation, …) is simply returned without modification. Likewise, its possible to specify a new function which inverts the case (upper → lower and vice verse) or substitutes characters not supported for a certain kind of string with a exclamation mark.

Apply test function for single char on all chars of a string

This function checks, whether all chars of a given string are satisfying a test function. For example, the standard function `isdigit` can be used to check if a string only contains decimal digits.

```
int strctestfn (char* test, int(*tfn)(int))
```

Here, `test` is a pointer to the string, which characters have to be checked for satisfying the function `tfn`. The function is expected to take an integer as argument and return an integer, which is `0`, if the tested char didn‘t satisfy the conditions. Otherwise, a positive number should be returned. The function itself returns `0`, if at least one char did not satisfy the functions conditions, otherwise, its `1`.

Together with the standard functions or the functions from this library (with underscore included), it can for example be checked, if a given string only contains digits, hexadecimal digits, letters, graphical symbols etc. Likewise, functions from other libraries or self-defined may be given as an argument for performing several kinds of testing entire strings.

Clear sentence from white space

This function modifies a string so that there is no white space before the first or behind the last graphical character and no multiple white space characters within the string.

```
void strstrp (char* dest, char* src);
```

`src` is a pointer the string to be cleared and `dest` is a pointer to the destination, where the cleared string will be included.