Key:Value File Parser

Generated by Doxygen 1.8.17

Chapter 1

Class Index

1.1 Class List

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

Chapter 2

File Index

2.1 File List

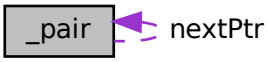
Here is a list of all documented files with brief descriptions:

Chapter 3

Class Documentation

3.1 _pair Struct Reference

Collaboration diagram for _pair:



Public Attributes

- char key [MAX_KEY_LEN+1]

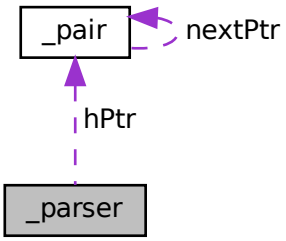
  A key:value pair.
- char **val** [MAX_VAL_LEN+1]
- struct _pair * **nextPtr**

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

- fileparser.c

3.2 _parser Struct Reference

Collaboration diagram for _parser:



Public Attributes

- char filename [MAX_FILENAME_LEN+1]

  A parser for key:value files. Supported file must adhere to this format: each line contains one and only one k:v pair, and the delimitator needs to be the same for all lines: `key<delim>value`
- char **delim** [MAX_DELIM_LEN+1]
- struct _pair * **hPtr**
- int **err**

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

- fileparser.c

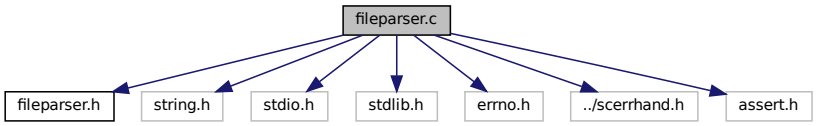
Chapter 4

File Documentation

4.1 fileparser.c File Reference

```
#include "fileparser.h"
#include <string.h>
#include <stdio.h>
#include <stdlib.h>
#include <errno.h>
#include "../scerrhand.h"
#include <assert.h>
```

Include dependency graph for fileparser.c:



Classes

- struct _pair
- struct _parser

Macros

- #define **MAX_FILENAME_LEN** 100
- #define **MAX_DELIM_LEN** 1
- #define **MAX_KEY_LEN** 50
- #define **MAX_VAL_LEN** 200

Functions

- static void **push** (struct _pair ∗∗hPtr, struct _pair ∗newPair)
- static int **parse** (Parser ∗p, FILE ∗fp)
- Parser ∗ parseFile (char ∗filename, char ∗delim)
- int parserTestErr (Parser ∗p)
- void printErrAsStr (Parser ∗p)
- int getValueFor (Parser ∗p, const char ∗key, char ∗dest, const char ∗defaultVal)
- long getLongValueFor (Parser ∗p, const char ∗key, long defaultVal, int ∗err)
- void destroyParser (Parser ∗p)

4.1.1 Function Documentation

4.1.1.1 destroyParser()

```
void destroyParser (
            Parser * p )
```

Frees every key:value pair in the parser, than frees the parser.

Parameters

p	A pointer to the parser to free.

4.1.1.2 getLongValueFor()

```
long getLongValueFor (
            Parser * p,
            const char * key,
            long defaultVal,
            int * err )
```

Utility function to get a `long` value for a key

Parameters

p	A pointer to the parser from which we want to get the value.
key	A string containing the key name
defaultVal	A string containing the default value we want to get if the requested key cannot be found in the parser
err	A pointer to an int which will contain an error value (see below)

Returns

The value associated with `key` (cast to `long`) if such a value exists and is a valid `long`, `defaultVal` if the key couldn't be found, and -1 if an error occurred (also sets err — see below)

At the end of execution, `err` will contain one of the following values:
0 no error(s) occurred
-1 requested value caused overflow or underflow
-2 the value associated with `key` wasn't a valid `long`

4.1.1.3 getValueFor()

```
int getValueFor (
            Parser * p,
            const char * key,
            char * dest,
            const char * defaultVal )
```

Attempts to get the value associated with the given key.

Parameters

p	A pointer to the parser from which we want to get the value.
key	A string containing the key name
dest	The pointer to a buffer to which the value is to be copied
defaultVal	A string containing the default value we want to get if the requested key cannot be found in the parser

Returns

1 if the key was found in the parser, 0 if no such key was found and the supplied default value was used

At the end of execution, param `dest` will contain either the `defaultVal` or the value associated with `key`.

4.1.1.4 parseFile()

```
Parser* parseFile (
            char * filename,
            char * delim )
```

Loads the key:value pairs from specified file into memory

Parameters

filename	Name of the file to parse
delim	Delimitator for key:value pairs

Returns

A pointer to a newly allocated `Parser` containing the parsed data (see `parserTestErr`)

NULL if memory for the parser could not be allocated

4.1.1.5 parserTestErr()

```
int parserTestErr (
            Parser * p )
```

Returns an int describing an error that occurred during the use of `p`

Parameters

p	a pointer to the parser to test for errors

Returns

0 if no error occurred, otherwise see below

Error codes:
-1: memory allocation for one of the k:v pairs failed (malloc error)
-2: requested file could not be opened (fopen error)
a positive integer `j`: syntax error on line `j` (parsing error)

4.1.1.6 printErrAsStr()

```
void printErrAsStr (
            Parser * p )
```

Prints a description of the error in `p->err`

Parameters

p	A pointer to the parser whose error we want to print.

Index