myfind

1.0.0

Generated by Doxygen 1.6.1

Mon Mar 19 23:03:26 2018

Contents

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 files with brief descriptions:

Chapter 3

Class Documentation

3.1 FileNode Struct Reference

A single node in the linked list of file names. Collaboration diagram for FileNode:

Public Attributes

- char ∗ fileName

  The name of the file (or directory). This member must not be NULL.

- struct FileNode ∗ next

  A pointer to the next node in the list, or NULL if this is the last node.

3.1.1 Detailed Description

Definition at line 30 of file myfind.c.

3.1.2 Member Data Documentation

3.1.2.1 char∗ FileNode::fileName

Definition at line 33 of file myfind.c.

Referenced by AddListNode(), do_dir(), and FreeList().

3.1.2.2 struct FileNode∗ FileNode::next `[read]`

Definition at line 36 of file myfind.c.

Referenced by AddListNode(), do_dir(), and FreeList().

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

- myfind.c

Chapter 4

File Documentation

4.1 myfind.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <errno.h>
#include <assert.h>
#include <dirent.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <unistd.h>
#include <sys/queue.h>
```

Include dependency graph for myfind.c:

Classes

- struct FileNode

  A single node in the linked list of file names.

Functions

- void do_file (const char ∗file_name, const char ∗const parms[ ])
- void do_dir (const char ∗dir_name, const char ∗const parms[ ])
- char ∗ CombinePath (const char ∗path1, const char ∗path2)
- void PrintFileInformation (const char ∗filePath, const struct stat ∗fileInformation, const char ∗const parms[ ])
- struct FileNode ∗ AddListNode (struct FileNode ∗∗head, char ∗fileName)
- void FreeList (struct FileNode ∗∗head)
- int main (int argc, char ∗argv[ ])

4.1.1 Detailed Description

myfind - A simplified version of the "find" utility provided by the Linux shell.

Author:

Alexander Feldinger <`ic17b055@technikum-wien.at`>
Thomas Haberl <`ic17b021@technikum-wien.at`>
Michael Zajac <`ic17b088@technikum-wien.at`>

Date:

2018-03-19

Version:

0.1

Definition in file myfind.c.

4.1.2 Function Documentation

4.1.2.1 struct FileNode ∗ AddListNode (struct FileNode ∗∗ *head*, char ∗ *fileName*) `[read]`

Creates a new file node and adds it to the linked list.

Parameters:

head	A pointer to the head of the linked list into which the new node should be inserted.
fileName	The file name to store in the created node.

Returns:

The created file node.

Definition at line 323 of file myfind.c.

References FileNode::fileName, and FileNode::next.

Referenced by do_dir().

4.1.2.2 char ∗ CombinePath (const char ∗ *path1*, const char ∗ *path2*)

Concatenates the provided path strings into a single path, adding or removing the intermediate directory separator as necessary.

Parameters:

path1	The first path to combine.
path2	The second path to combine.

Returns:

The combined path as a newly allocated string, which needs to be released with free().

Definition at line 211 of file myfind.c.

Referenced by do_dir().

4.1.2.3 void do_dir (const char ∗ *dir_name*, const char ∗const *parms*[ ])

Enumerates the files and directories below the specified directory path and prints the information of each entry according to the actions specified in parms.

Parameters:

dir_name The path of the directory to process.

parms The array of command line arguments representing the actions used for printing the information of each file or directory entry.

Definition at line 101 of file myfind.c.

References AddListNode(), CombinePath(), do_file(), FileNode::fileName, FreeList(), and FileNode::next.

Referenced by do_file().

Here is the call graph for this function:

4.1.2.4 void do_file (const char ∗ *file_name*, const char ∗const *parms*[ ])

Recursively walks through all the files and directories below the specified path and prints the information of each entry according to the actions specified in parms.

Parameters:

file_name The path of the file or directory to process.

parms The array of command line arguments representing the actions used for printing the information of each file or directory entry.

Definition at line 74 of file myfind.c.

References do_dir(), and PrintFileInformation().

Referenced by do_dir(), and main().

Here is the call graph for this function:

4.1.2.5 void FreeList (struct FileNode ∗∗ *head*)

Frees all nodes in the provided linked list.

Parameters:

head A pointer to the head of the linked list to be freed.

Definition at line 359 of file myfind.c.

References FileNode::fileName, and FileNode::next.

Referenced by do_dir().

4.1.2.6 int main (int *argc*, char ∗ *argv*[ ])

The entry point of the application.

Parameters:

argc The number of command line arguments in argv.

argv The array of command line arguments.

Returns:

Zero if execution was successful. -1 if an unrecoverable error occurred during execution.

Definition at line 56 of file myfind.c.

References do_file().

Here is the call graph for this function:

4.1.2.7 void PrintFileInformation (const char ∗ *filePath*, const struct stat ∗ *fileInformation*, const char ∗const *parms*[ ])

Prints the information of a single file or directory.

Parameters:

filePath The path of the file to print.

fileInformation The information of the file as returned by stat().

parms The array of command line arguments representing the actions used for printing the information of each file or directory entry.

Definition at line 278 of file myfind.c.

Referenced by do_file().

Index