Paul Prince's MPX

R1

Generated by Doxygen 1.7.3

Sat Feb 26 2011 22:02:07

Contents

1 Introduction

1.1 Repository

Version-control information is managed by Git, and hosted by GitHub:

- Website: https://github.com/pprince/cs450
- Public Repo: git://github.com/pprince/cs450.git
- Comitters: git@github.com:pprince/cs450.git

1.2 Documentation

Documentation for developers is generated by Doxygen; for detailed information about the files, functions, data structures, etc. that make up MPX and how they relate to each other, refer to:

"MPX Programmer's Manual"

which can be found in the doc/ directory. Also, in the same directory, you can find the current version of:

"MPX User's Manual"

2 Data Structure Index

2.1 Data Structures

Here are the data structures with brief descriptions:

3 File Index

3.1 File List

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

4 Data Structure Documentation

4.1 date_rec Struct Reference

Data Fields

- int **month**
- int **day**
- int **year**

4.1.1 Detailed Description

Definition at line 124 of file mpx_supt.h.

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

- mpx/mpx_supt.h

4.2 mpx_command Struct Reference

```
#include <mpx_cmds.h>
```

Data Fields

- char * **name**
- void(* **function**)(int argc, char *argv[])
- struct mpx_command * **next**

4.2.1 Detailed Description

Node type for a singly-linked list of MPX commands.

Definition at line 6 of file mpx_cmds.h.

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

- mpx/mpx_cmds.h

4.3 params Struct Reference

Data Fields

- int **op_code**
- int **device_id**
- char * **buf_p**
- int * **count_p**

4.3.1 Detailed Description

Definition at line 107 of file mpx_supt.c.

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

- mpx/mpx_supt.c

5 File Documentation

5.1 mpx/mpx.c File Reference

MPX main() Function.

```
#include "mpx_supt.h"
#include "mpx_util.h"
#include "mpx_sh.h"
#include "mpx_cmds.h"
```

Functions

- void main (int argc, char *argv[])

5.1.1 Detailed Description

MPX main() Function.

Author

Paul Prince <paul@littlebluetech.com>

Date

2011

This file contains the start-of-execution, i.e. function main(), for MPX.

Definition in file mpx.c.

5.1.2 Function Documentation

5.1.2.1 void main (int *argc*, char * *argv[]*)

This is the start-of-execution for the MPX executable.

Definition at line 45 of file mpx.c.

5.2 mpx/mpx_cmds.c File Reference

MPX User Commands.

```
#include "mpx_cmds.h"
#include "mpx_supt.h"
#include "mpx_util.h"
#include <string.h>
```

Functions

- void add_command (char *name, void(*function)(int argc, char *argv[]))
- void **dispatch_command** (char *name, int argc, char *argv[])
- void **mpxcmd_commands** (int argc, char *argv[])
- void mpxcmd_date (int argc, char *argv[])
- void **mpxcmd_exit** (int argc, char *argv[])
- void **mpxcmd_help** (int argc, char *argv[])
- void **mpxcmd_version** (int argc, char *argv[])
- void **mpxcmd_ls** (int argc, char *argv[])
- void **init_commands** (void)

Variables

- static struct mpx_command * **list_head** = NULL

5.2.1 Detailed Description

MPX User Commands. This file implements each of the user commands for MPX.

Definition in file mpx_cmds.c.

5.2.2 Function Documentation

5.2.2.1 void add_command (char * *name*, void(*)(int argc, char *argv[]) *function*)

Temporary variable for iterating through the list of commands.

Definition at line 35 of file mpx_cmds.c.

5.2.2.2 void mpxcmd_date (int *argc*, char * *argv[]*)

< Temp. storage for the return value of sys_ functions.

< Structure to hold a date (day, month, and year). Will be used for both getting and setting the MPX system date.

Definition at line 130 of file mpx_cmds.c.

5.3 mpx/mpx_sh.c File Reference

MPX Shell, aka Command Handler.

```
#include "mpx_sh.h"
#include "mpx_supt.h"
#include "mpx_util.h"
#include "mpx_cmds.h"
#include <string.h>
```

Functions

- void mpx_setprompt (char *new_prompt)
- void mpx_shell (void)

Variables

- static char * mpx_prompt_string = NULL

5.3.1 Detailed Description

MPX Shell, aka Command Handler. This file implements the user interface for MPX.

Definition in file mpx_sh.c.

5.3.2 Function Documentation

5.3.2.1 void mpx_setprompt (char * *new_prompt*)

Sets the current prompt to whatever string is given.

If new_prompt is NULL, this is a no-op.

Definition at line 41 of file mpx_sh.c.

5.3.2.2 void mpx_shell (void)

This function implements the MPX shell (command-line user interface).

mpx_shell() never returns!

A buffer to hold the command line input by the user. We include space for the ,

, and \0 characters, if any.

Buffer size argument for passing to sys_req().

Used to capture the return value of sys_req().

argc to be passed to MPX command; works just like the one passed to main().

argv array to be passed to MPX command; works almost just like the one passed to main().

But there is one caveat: argv[argc] is undefined in my implementation, not garanteed to be NULL.

Temporary pointer for use in string tokenization.

Delimiters that separate arguments in the MPX shell command-line environment.

An index for use in for(;;) loops.

An index for use in nested for(;;) loops.

Definition at line 56 of file mpx_sh.c.

5.3.3 Variable Documentation

5.3.3.1 char* mpx_prompt_string = NULL [static]

The current prompt string.

Definition at line 35 of file mpx_sh.c.

5.4 mpx/mpx_util.c File Reference

Various utility functions used by all of MPX.

```
#include "mpx_util.h"
#include "mpx_supt.h"
#include <string.h>
#include <stdio.h>
```

Functions

- int mpx_chomp (char *str)
- int **mpx_validate_date** (int year, int month, int day)
- int **mpx_cat** (char *file_name)

5.4.1 Detailed Description

Various utility functions used by all of MPX. This file contains the functions etc. to implement the user interface for MPX.

Definition in file mpx_util.c.

5.4.2 Function Documentation

5.4.2.1 int mpx_chomp (char * *str*)

Removes trailing newline, if any.

This function checks to see if the last character in a string is a newline, and, if so, removes it. Otherwise, the string is left unchanged.

The input must be a valid (allocated and null-terminated) C string, otherwise the results are undefined (but will most likley result in a segmentation fault / protection fault).

Returns the number of characters removed from the string.

Parameters

str	The string to chomp.

Definition at line 41 of file mpx_util.c.

Index