ncpty

Timothy Brackett

Apr 11, 2022

CONTENTS

An NCurses pseudoterminal library.

API still under initial development.

CHAPTER

ONE

API

struct ncpty_t

Public Members

int **fd**
: File descriptor for master-side of pseudoterminal.

pid_t **pid**
: Process id.

VTerm ***vt**
: Virtual terminal pointer.

PANEL ***panel**
: ncurses panel

file **ncpty.h**
: *#include <ncurses.h>#include <panel.h>#include <sys/types.h>#include <vterm.h>* Public header for ncpty - the curses pseudoterminal.

Typedefs

typedef struct *ncpty_t* **ncpty**

Functions

struct *ncpty_t* ***ncpty_new**(PANEL **panel*)
: Allocate a new ncpty object assigned with an ncurses `PANEL`.

  The pseudoterminal assumes responsibility for the `WINDOW` until a call to `ncpty_free` and any existing pointers to the `WINDOW` should be considered invalid.

  Return a newly allocated *`ncpty_t`* if successful, otherwise returns `NULL`

  Parameters

  - `[in] panel`: ncurses panel to be used by the pseudoterminal process

void **ncpty_free**(**struct** *ncpty_t* **pty*)
: Deallocate an ncpty object.

  Parameters

  - `[in] pty`: ncpty object to destroy

int **ncpty_execvp**(**struct** *ncpty_t* *pty, **const** char *file, char ***const** argv[])
Create a pseudoterminal, fork, exec and attach a child process to the child-side.

If this call fails for any reason, the ncpty object is left in a definite state.

Return 0 for success, -1 for execvp error. errno is set to indicate the specific error encountered.

Parameters

- `[out] pty`: ncpty object device
- `[in] file`: filename of file to executed (will search on $PATH)
- `[in] argv`: array of null-terminated string arguments, terminated by a null pointer

bool **ncpty_status**(**struct** *ncpty_t* *pty, int *exit_code)
Check if child process controlled by ncpty object has exited.

Return `true` if process is still running, `false` otherwise

Parameters

- `[in] pty`: ncpty object
- `[out] exit_code`:

void **ncpty_exit**(int *exit_code)
Exit program.

Convenience function. Uses `exit_curses` if built with ncurses 6.2.20191214 or later.

Parameters

- `[in] exit_code`:

file **banner.c**
#include <stdio.h>#include <string.h>#include <sys/stat.h>#include <sys/types.h>#include <sys/wait.h>#include <unistd.h>#include "ncpty.h" Simple example of ncpty use that wraps a terminal in header and footer banners.

Functions

void **print_usage**(**const** char *name)

int **fallback**(int argc, char **argv)

int **main**(int argc, char **argv)

file **ncpty.c**
#include "ncpty.h"#include <curses.h>#include <fcntl.h>#include <stdio.h>#include <stdlib.h>#include <string.h>#include <sys/ioctl.h>#include <sys/wait.h>#include <termios.h>#include <unistd.h>

Functions

struct *ncpty_t* ***ncpty_new**(PANEL **panel*)
Allocate a new ncpty object assigned with an ncurses `PANEL`.

The pseudoterminal assumes responsibility for the `WINDOW` until a call to `ncpty_free` and any existing pointers to the `WINDOW` should be considered invalid.

Return a newly allocated *`ncpty_t`* if successful, otherwise returns `NULL`

Parameters

- `[in] panel`: ncurses panel to be used by the pseudoterminal process

void **ncpty_free**(**struct** *ncpty_t* **pty*)
Deallocate an ncpty object.

Parameters

- `[in] pty`: ncpty object to destroy

int **ncpty_execvp**(**struct** *ncpty_t* **pty*, **const** char **file*, char ***const** *argv*[])
Create a pseudoterminal, fork, exec and attach a child process to the child-side.

If this call fails for any reason, the ncpty object is left in a definite state.

Return 0 for success, -1 for execvp error. errno is set to indicate the specific error encountered.

Parameters

- `[out] pty`: ncpty object device
- `[in] file`: filename of file to executed (will search on $PATH)
- `[in] argv`: array of null-terminated string arguments, terminated by a null pointer

bool **ncpty_status**(**struct** *ncpty_t* **pty*, int **exit_code*)
Check if child process controlled by ncpty object has exited.

Return `true` if process is still running, `false` otherwise

Parameters

- `[in] pty`: ncpty object
- `[out] exit_code`:

void **ncpty_exit**(int *exit_code*)
Exit program.

Convenience function. Uses `exit_curses` if built with ncurses 6.2.20191214 or later.

Parameters

- `[in] exit_code`:

file **ncpty_test.c**
#include “greatest.h”#include “ncpty.h”#include <sys/resource.h>#include <sys/queue.h>#include <sys/time.h>

Functions

TEST **invalid_status_tests** (void)

TEST **invalid_free_test** (void)

TEST **fork_fail_test** (void)

TEST **resource_starvation_test** (void)

GREATEST_MAIN_DEFS ()

int **main** (int *argc*, char ***argv*)

dir **src/bin**

dir **include**

dir **src/lib**

dir **src**

dir **src/test**

INDEX