NAME

explain_freopen − explain freopen(3) errors

SYNOPSIS

#include <libexplain/freopen.h>
const char *explain_freopen(const char *pathname, const char *flags, FILE *fp);
const char *explain_errno_freopen(int errnum, const char *pathname, const char *flags, FILE *fp);
void explain_message_freopen(char *message, int message_size, const char *pathname, const char *flags, FILE *fp);
void explain_message_errno_freopen(char *message, int message_size, int errnum, const char *pathname, const char *flags, FILE *fp);

DESCRIPTION

These functions may be used to obtain explanations for *freopen*(3) errors.

explain_freopen

const char *explain_freopen(const char *pathname, const char *flags, FILE *fp);

The explain_freopen function is used to obtain an explanation of an error returned by the *freopen*(3) system call. The least the message will contain is the value of `strerror(errno)`, but usually it will do much better, and indicate the underlying cause in more detail.

The *errno* global variable will be used to obtain the error value to be decoded.

This function is intended to be used in a fashion similar to the following example:

```
if (!freopen(pathname, flags, fp))
{
    fprintf(stderr, ’%s0, explain_freopen(pathname, flags, fp));
    exit(EXIT_FAILURE);
}
```

pathname
The original pathname, exactly as passed to the *freopen*(3) system call.

flags The original flags, exactly as passed to the *freopen*(3) system call.

fp The original fp, exactly as passed to the *freopen*(3) system call.

Returns: The message explaining the error. This message buffer is shared by all libexplain functions which do not supply a buffer in their argument list. This will be overwritten by the next call to any libexplain function which shares this buffer, including other threads.

Note: This function is **not** thread safe, because it shares a return buffer across all threads, and many other functions in this library.

explain_errno_freopen

const char *explain_errno_freopen(int errnum, const char *pathname, const char *flags, FILE *fp);

The explain_errno_freopen function is used to obtain an explanation of an error returned by the *freopen*(3) system call. The least the message will contain is the value of `strerror(errnum)`, but usually it will do much better, and indicate the underlying cause in more detail.

This function is intended to be used in a fashion similar to the following example:

```
if (freopen(pathname, flags, fp))
{
    int err = errno;
    fprintf(stderr, ’%s0, explain_errno_freopen(err, pathname,
        flags, fp));
    exit(EXIT_FAILURE);
}
```

errnum The error value to be decoded, usually obtained from the *errno* global variable just before this function is called. This is necessary if you need to call **any** code between the system call to be explained and this function, because many libc functions will alter the value of *errno*.

pathname
The original pathname, exactly as passed to the *freopen*(3) system call.

flags The original flags, exactly as passed to the *freopen*(3) system call.

fp The original fp, exactly as passed to the *freopen*(3) system call.

Returns: The message explaining the error. This message buffer is shared by all libexplain functions which do not supply a buffer in their argument list. This will be overwritten by the next call to any libexplain function which shares this buffer, including other threads.

Note: This function is **not** thread safe, because it shares a return buffer across all threads, and many other functions in this library.

explain_message_freopen

void explain_message_freopen(char *message, int message_size, const char *pathname, const char *flags, FILE *fp);

The explain_message_freopen function is used to obtain an explanation of an error returned by the *freopen*(3) system call. The least the message will contain is the value of `strerror(errno)`, but usually it will do much better, and indicate the underlying cause in more detail.

The *errno* global variable will be used to obtain the error value to be decoded.

This function is intended to be used in a fashion similar to the following example:

```
if (!freopen(pathname, flags, fp))
{
    char message[3000];
    explain_message_freopen(message, sizeof(message), pathname, flags,
        fp);
    fprintf(stderr, ’%s0, message);
    exit(EXIT_FAILURE);
}
```

message The location in which to store the returned message. Because a message return buffer has been supplied, this function is thread safe.

message_size
The size in bytes of the location in which to store the returned message.

pathname
The original pathname, exactly as passed to the *freopen*(3) system call.

flags The original flags, exactly as passed to the *freopen*(3) system call.

fp The original fp, exactly as passed to the *freopen*(3) system call.

explain_message_errno_freopen

void explain_message_errno_freopen(char *message, int message_size, int errnum, const char *pathname, const char *flags, FILE *fp);

The explain_message_errno_freopen function is used to obtain an explanation of an error returned by the *freopen*(3) system call. The least the message will contain is the value of `strerror(errnum)`, but usually it will do much better, and indicate the underlying cause in more detail.

This function is intended to be used in a fashion similar to the following example:

```
if (!freopen(pathname, flags, fp))
{
    int err = errno;
    char message[3000];
    explain_message_errno_freopen(message, sizeof(message), err,
        pathname, flags, fp);
    fprintf(stderr, ’%s0, message);
    exit(EXIT_FAILURE);
}
```

message The location in which to store the returned message. Because a message return buffer has been supplied, this function is thread safe.

message_size
The size in bytes of the location in which to store the returned message.

errnum The error value to be decoded, usually obtained from the *errno* global variable just before this function is called. This is necessary if you need to call **any** code between the system call to be explained and this function, because many libc functions will alter the value of *errno*.

pathname
The original pathname, exactly as passed to the *freopen*(3) system call.

flags The original flags, exactly as passed to the *freopen*(3) system call.

fp The original fp, exactly as passed to the *freopen*(3) system call.

COPYRIGHT

libexplain version 1.4

AUTHOR

Written by Peter Miller <pmiller AT opensource DOT org DOT au>