NAME

explain_putc − explain putc(3) errors

SYNOPSIS

#include <libexplain/putc.h>

const char *explain_putc(int c, FILE *fp);
const char *explain_errno_putc(int errnum, int c, FILE *fp);
void explain_message_putc(char *message, int message_size, int c, FILE *fp);
void explain_message_errno_putc(char *message, int message_size, int errnum, int c, FILE *fp);

DESCRIPTION

These functions may be used to obtain explanations for errors returned by the *putc*(3) system call.

explain_putc

const char *explain_putc(int c, FILE *fp);

The **explain_putc** function is used to obtain an explanation of an error returned by the *putc*(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 (putc(c, fp) == EOF)
{
    fprintf(stderr, "%s\n", explain_putc(c, fp));
    exit(EXIT_FAILURE);
}
```

c The original c, exactly as passed to the *putc*(3) system call.

fp The original fp, exactly as passed to the *putc*(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_putc

const char *explain_errno_putc(int errnum, int c, FILE *fp);

The **explain_errno_putc** function is used to obtain an explanation of an error returned by the *putc*(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 (putc(c, fp) == EOF)
{
    int err = errno;
    fprintf(stderr, "%s\n", explain_errno_putc(err, c, 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*.

c The original c, exactly as passed to the *putc*(3) system call.

fp The original fp, exactly as passed to the *putc*(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_putc

void explain_message_putc(char *message, int message_size, int c, FILE *fp);

The **explain_message_putc** function may be used to obtain an explanation of an error returned by the *putc*(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 (putc(c, fp) == EOF)
{
    char message[3000];
    explain_message_putc(message, sizeof(message), c, fp);
    fprintf(stderr, "%s\n", message);
    exit(EXIT_FAILURE);
}
```

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

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

c The original c, exactly as passed to the *putc*(3) system call.

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

explain_message_errno_putc

void explain_message_errno_putc(char *message, int message_size, int errnum, int c, FILE *fp);

The **explain_message_errno_putc** function may be used to obtain an explanation of an error returned by the *putc*(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 (putc(c, fp) == EOF)
{
    int err = errno;
    char message[3000];
    explain_message_errno_putc(message, sizeof(message), err, c, fp);
    fprintf(stderr, "%s\n", message);
    exit(EXIT_FAILURE);
}
```

message The location in which to store the returned message. If a suitable message return buffer is 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*.

c The original c, exactly as passed to the *putc*(3) system call.

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

SEE ALSO

putc(3) output of characters

explain_putc_or_die(3)
output of characters and report errors

COPYRIGHT