## NAME

explain_stat − explain stat(2) errors

## SYNOPSIS

#include <libexplain/stat.h>
const char *explain_stat(const char *pathname, const struct stat *buf);
void explain_message_stat(char *message, int message_size, const char *pathname, const struct stat *buf);
const char *explain_errno_stat(int errnum, const char *pathname, const struct stat *buf);
void explain_message_errno_stat(char *message, int message_size, int errnum, const char *pathname, const struct stat *buf);

## DESCRIPTION

These functions may be used to obtain explanations for *stat*(2) errors .

### explain_errno_stat

const char *explain_errno_stat(int errnum, const char *pathname, const struct stat *buf);

The explain_errno_stat function is used to obtain an explanation of an error returned by the *stat*(2) function. 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 (stat(pathname, &buf) < 0)
{
    int err = errno;
    fprintf(stderr, "%s\n", explain_errno_stat(err, pathname, &buf));
    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 *stat*(2) system call.

*buf* The original buf, exactly as passed to the *stat*(2) 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_errno_stat

void explain_message_errno_stat(char *message, int message_size, int errnum, const char *pathname, const struct stat *buf);

The explain_message_errno_stat function is used to obtain an explanation of an error returned by the *stat*(2) function. 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 (stat(pathname, &buf) < 0)
{
    int err = errno;
    char message[3000];
    explain_message_errno_stat(message, sizeof(message), err,
        pathname, &buf);
    fprintf(stderr, "%s\n", 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 *stat*(2) system call.

*buf* The original buf, exactly as passed to the *stat*(2) system call.

**explain_message_stat**
void explain_message_stat(char *message, int message_size, const char *pathname, const struct stat *buf);

The explain_message_stat function is used to obtain an explanation of an error returned by the *stat*(2) function. 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 (stat(pathname, &buf) < 0)
{
    char message[3000];
    explain_message_stat(message, sizeof(message), pathname, &buf);
    fprintf(stderr, "%s\n", 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 *stat*(2) system call.

*buf* The original buf, exactly as passed to the *stat*(2) system call.

**explain_stat**
const char *explain_stat(const char *pathname, const struct stat * buf);

The explain_stat function is used to obtain an explanation of an error returned by the *stat*(2) function. 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 (stat(pathname, &buf) < 0)
{
    fprintf(stderr, "%s\n", explain_stat(pathname, &buf));
    exit(EXIT_FAILURE);
}
```

*pathname*
The original pathname, exactly as passed to the *stat*(2) system call.

*buf* The original buf, exactly as passed to the *stat*(2) 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.

## COPYRIGHT



## AUTHOR

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