

explain_lutimes(3)

explain_lutimes(3)

NAMEexplain_lutimes – explain *lutimes*(3) errors**SYNOPSIS**

```
#include <libexplain/lutimes.h>

const char *explain_lutimes(const char *pathname, const struct timeval *data);
const char *explain_errno_lutimes(int errnum, const char *pathname, const struct timeval *data);
void explain_message_lutimes(char *message, int message_size, const char *pathname, const struct timeval *data);
void explain_message_errno_lutimes(char *message, int message_size, int errnum, const char *pathname, const struct timeval *data);
```

DESCRIPTION

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

explain_lutimes

```
const char *explain_lutimes(const char *pathname, const struct timeval *data);
```

The **explain_lutimes** function is used to obtain an explanation of an error returned by the *lutimes*(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.

pathname

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

data

The original data, exactly as passed to the *lutimes*(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.

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

```
if (lutimes(pathname, data) < 0)
{
    fprintf(stderr, "%s\n", explain_lutimes(pathname, data));
    exit(EXIT_FAILURE);
}
```

The above code example is available pre-packaged as the *explain_lutimes_or_die*(3) function.

explain_errno_lutimes

```
const char *explain_errno_lutimes(int errnum, const char *pathname, const struct timeval *data);
```

The **explain_errno_lutimes** function is used to obtain an explanation of an error returned by the *lutimes*(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.

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 *lutimes*(3) system call.

data

The original data, exactly as passed to the *lutimes*(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.

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

```
if (lutimes(pathname, data) < 0)
```



explain_lutimes(3)

explain_lutimes(3)

```

    {
        int err = errno;
        fprintf(stderr, "%s\n", explain_errno_lutimes(err, path-
            name, data));
        exit(EXIT_FAILURE);
    }

```

The above code example is available pre-packaged as the *explain_lutimes_or_die(3)* function.

explain_message_lutimes

```
void explain_message_lutimes(char *message, int message_size, const char *pathname, const struct
timeval *data);
```

The **explain_message_lutimes** function is used to obtain an explanation of an error returned by the *lutimes(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.

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.

pathname

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

data

The original data, exactly as passed to the *lutimes(3)* system call.

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

```

if (lutimes(pathname, data) < 0)
{
    char message[3000];
    explain_message_lutimes(message, sizeof(message), pathname,
        data);
    fprintf(stderr, "%s\n", message);
    exit(EXIT_FAILURE);
}

```

The above code example is available pre-packaged as the *explain_lutimes_or_die(3)* function.

explain_message_errno_lutimes

```
void explain_message_errno_lutimes(char *message, int message_size, int errnum, const char *path-
name, const struct timeval *data);
```

The **explain_message_errno_lutimes** function is used to obtain an explanation of an error returned by the *lutimes(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.

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*.

pathname

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

data

The original data, exactly as passed to the *lutimes(3)* system call.

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

```

if (lutimes(pathname, data) < 0)
{
    int err = errno;

```



explain_lutimes(3)

explain_lutimes(3)

```
    char message[3000];
    explain_message_errno_lutimes(message, sizeof(message),
    err, pathname, data);
    fprintf(stderr, "%s\n", message);
    exit(EXIT_FAILURE);
}
```

The above code example is available pre-packaged as the *explain_lutimes_or_die(3)* function.

SEE ALSO*lutimes(3)*

modify file timestamps

explain_lutimes_or_die(3)

modify file timestamps and report errors

COPYRIGHT

libexplain version 1.4

Copyright © 2013 Peter Miller

