NAME

explain_accept − explain accept(2) errors

SYNOPSIS

#include <libexplain/accept.h>

const char *explain_accept(int fildes, struct sockaddr *sock_addr, socklen_t *sock_addr_size);
const char *explain_errno_accept(int errnum, int fildes, struct sockaddr *sock_addr, socklen_t *sock_addr_size);
void explain_message_accept(char *message, int message_size, int fildes, struct sockaddr *sock_addr, socklen_t *sock_addrlen);
void explain_message_errno_accept(char *message, int message_size, int errnum, int fildes, struct sockaddr *sock_addr, socklen_t *sock_addr_size);

DESCRIPTION

These functions may be used to obtain explanations for errors returned by the *accept*(2) system call.

explain_accept

const char *explain_accept(int fildes, struct sockaddr *sock_addr, socklen_t *sock_addr_size);

The **explain_accept** function is used to obtain an explanation of an error returned by the *accept*(2) 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 (accept(fildes, sock_addr, sock_addr_size) < 0)
{
    fprintf(stderr, "%s\n", explain_accept(fildes, sock_addr,
        sock_addr_size));
    exit(EXIT_FAILURE);
}
```

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

fildes The original fildes, exactly as passed to the *accept*(2) system call.

sock_addr
The original sock_addr, exactly as passed to the *accept*(2) system call.

sock_addr_size
The original sock_addr_size, exactly as passed to the *accept*(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_errno_accept

const char *explain_errno_accept(int errnum, int fildes, struct sockaddr *sock_addr, socklen_t *sock_addr_size);

The **explain_errno_accept** function is used to obtain an explanation of an error returned by the *accept*(2) 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 (accept(fildes, sock_addr, sock_addr_size) < 0)
{
    int err = errno;
    fprintf(stderr, "%s\n", explain_errno_accept(err, fildes, sock_addr,
        sock_addr_size));
    exit(EXIT_FAILURE);
}
```

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

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*.

fildes The original fildes, exactly as passed to the *accept*(2) system call.

sock_addr
The original sock_addr, exactly as passed to the *accept*(2) system call.

sock_addr_size
The original sock_addr_size, exactly as passed to the *accept*(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_accept

void explain_message_accept(char *message, int message_size, int fildes, struct sockaddr *sock_addr, socklen_t *sock_addr_size);

The **explain_message_accept** function may be used to obtain an explanation of an error returned by the *accept*(2) 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 (accept(fildes, sock_addr, sock_addr_size) < 0)
{
    char message[3000];
    explain_message_accept(message, sizeof(message), fildes, sock_addr,
        sock_addr_size);
    fprintf(stderr, "%s\n", message);
    exit(EXIT_FAILURE);
}
```

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

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.

fildes The original fildes, exactly as passed to the *accept*(2) system call.

sock_addr
The original sock_addr, exactly as passed to the *accept*(2) system call.

sock_addr_size
The original sock_addr_size, exactly as passed to the *accept*(2) system call.

explain_message_errno_accept

void explain_message_errno_accept(char *message, int message_size, int errnum, int fildes, struct sockaddr *sock_addr, socklen_t *sock_addr_size);

The **explain_message_errno_accept** function may be used to obtain an explanation of an error returned by the *accept*(2) 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 (accept(fildes, sock_addr, sock_addr_size) < 0)
{
    int err = errno;
    char message[3000];
    explain_message_errno_accept(message, sizeof(message), err, fildes,
        sock_addr, sock_addr_size);
    fprintf(stderr, "%s\n", message);
    exit(EXIT_FAILURE);
}
```

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

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*.
fildes	The original fildes, exactly as passed to the *accept*(2) system call.
sock_addr	The original sock_addr, exactly as passed to the *accept*(2) system call.
sock_addr_size	The original sock_addr_size, exactly as passed to the *accept*(2) system call.

SEE ALSO

accept(2)	accept a connection on a socket
explain_accept_or_die(3)	accept a connection on a socket and report errors

COPYRIGHT

libexplain version 1.4