## NAME

explain_shmat − explain *shmat*(2) errors

## SYNOPSIS

#include <libexplain/shmat.h>

const char *explain_shmat(int shmid, const void *shmaddr, int shmflg);
const char *explain_errno_shmat(int errnum, int shmid, const void *shmaddr, int shmflg);
void explain_message_shmat(char *message, int message_size, int shmid, const void *shmaddr, int shmflg);
void explain_message_errno_shmat(char *message, int message_size, int errnum, int shmid, const void *shmaddr, int shmflg);

## DESCRIPTION

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

### explain_shmat

const char *explain_shmat(int shmid, const void *shmaddr, int shmflg);

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

*shmid* The original shmid, exactly as passed to the *shmat*(2) system call.

*shmaddr* The original shmaddr, exactly as passed to the *shmat*(2) system call.

*shmflg* The original shmflg, exactly as passed to the *shmat*(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.

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

```
void *result = shmat(shmid, shmaddr, shmflg);
if (!result)
{
    fprintf(stderr, "%s\n", explain_shmat(shmid, shmaddr, shm-
    flg));
     exit(EXIT_FAILURE);
}
```

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

### explain_errno_shmat

const char *explain_errno_shmat(int errnum, int shmid, const void *shmaddr, int shmflg);

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

*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*.

*shmid* The original shmid, exactly as passed to the *shmat*(2) system call.

*shmaddr* The original shmaddr, exactly as passed to the *shmat*(2) system call.

*shmflg* The original shmflg, exactly as passed to the *shmat*(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.

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

```
void *result = shmat(shmid, shmaddr, shmflg);
if (!result)
{
    int err = errno;
   fprintf(stderr, "%s\n", explain_errno_shmat(err, shmid,
   shmaddr, shmflg));
    exit(EXIT_FAILURE);
}
```

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

**explain_message_shmat**

void explain_message_shmat(char *message, int message_size, int shmid, const void *shmaddr, int shmflg);

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

*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.

*shmid* The original shmid, exactly as passed to the *shmat*(2) system call.

*shmaddr* The original shmaddr, exactly as passed to the *shmat*(2) system call.

*shmflg* The original shmflg, exactly as passed to the *shmat*(2) system call.

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

```
void *result = shmat(shmid, shmaddr, shmflg);
if (!result)
{
    char message[3000];
   explain_message_shmat(message, sizeof(message), shmid,
   shmaddr, shmflg);
    fprintf(stderr, "%s\n", message);
    exit(EXIT_FAILURE);
}
```

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

**explain_message_errno_shmat**

void explain_message_errno_shmat(char *message, int message_size, int errnum, int shmid, const void *shmaddr, int shmflg);

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

*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*.

*shmid* The original shmid, exactly as passed to the *shmat*(2) system call.

*shmaddr* The original shmaddr, exactly as passed to the *shmat*(2) system call.

*shmflg* The original shmflg, exactly as passed to the *shmat*(2) system call.

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

```
void *result = shmat(shmid, shmaddr, shmflg);
if (!result)
{
    int err = errno;
    char message[3000];
   explain_message_errno_shmat(message, sizeof(message), err,
   shmid, shmaddr, shmflg);
    fprintf(stderr, "%s\n", message);
    exit(EXIT_FAILURE);
}
```

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

## SEE ALSO

*shmat*(2)
shared memory attach

*explain_shmat_or_die*(3)
shared memory attach and report errors

## COPYRIGHT