NAME

perror, **strerror**, **strerror_l**, **strerror_r**, **sys_errlist**, **sys_nerr** - system error messages

LIBRARY

Standard C Library (libc, -lc)

SYNOPSIS

#include <stdio.h>

void
perror(*const char *string*);

*extern const char * const sys_errlist[]*;
extern const int sys_nerr;

#include <string.h>

*char **
strerror(*int errnum*);

*char **
strerror_l(*int errnum, locale_t*);

int
strerror_r(*int errnum, char *strerrbuf, size_t buflen*);

DESCRIPTION

The **strerror**(), **strerror_l**(), **strerror_r**(), and **perror**() functions look up the error message string corresponding to an error number.

The **strerror**() function accepts an error number argument *errnum* and returns a pointer to the corresponding message string in the current locale. **strerror**() is not thread-safe. It returns a pointer to an internal static buffer that could be overwritten by a **strerror**() call from another thread.

The **strerror_l**() function accepts *errnum* error number and *locale* locale handle arguments and returns a pointer to a string corresponding to the specified error in the given locale. **strerror_l**() is thread-safe, its result can be only overwritten by another call to **strerror_l**() from the current thread.

The **strerror_r**() function renders the same result into *strerrbuf* for a maximum of *buflen* characters and returns 0 upon success.

The **perror**() function finds the error message corresponding to the current value of the global variable *errno* (intro(2)) and writes it, followed by a newline, to the standard error file descriptor. If the argument *string* is non-NULL and does not point to the null character, this string is prepended to the message string and separated from it by a colon and space (": "); otherwise, only the error message string is printed.

If the error number is not recognized, these functions return an error message string containing "Unknown error: " followed by the error number in decimal. The **strerror**() and **strerror_r**() functions return EINVAL as a warning. Error numbers recognized by this implementation fall in the range 0 < *errnum* < *sys_nerr*. The number 0 is also recognized, although applications that take advantage of this are likely to use unspecified values of *errno*.

If insufficient storage is provided in *strerrbuf* (as specified in *buflen*) to contain the error string, **strerror_r**() returns ERANGE and *strerrbuf* will contain an error message that has been truncated and NUL terminated to fit the length specified by *buflen*.

The message strings can be accessed directly using the external array *sys_errlist*. The external value *sys_nerr* contains a count of the messages in *sys_errlist*. The use of these variables is deprecated; **strerror**(), **strerror_l**(), or **strerror_r**() should be used instead.

EXAMPLES

The following example shows how to use **perror**() to report an error.

```
#include <fcntl.h>
#include <stdio.h>
#include <stdlib.h>

int
main(void)
{
        int fd;

        if ((fd = open("/nonexistent", O_RDONLY)) == -1) {
                perror("open()");
                exit(1);
        }
        printf("File descriptor: %d\n", fd);
        return (0);
}
```

When executed, the program will print an error message along the lines of ‘open(): No such file or directory’.

SEE ALSO

intro(2), err(3), psignal(3)

STANDARDS

The **perror**() and **strerror**() functions conform to ISO/IEC 9899:1999 ("ISO C99"). The **strerror_r**() function conforms to IEEE Std 1003.1-2001 ("POSIX.1"). The **strerror_l**() function conforms to IEEE Std 1003.1-2008 ("POSIX.1").

HISTORY

The **strerror**() and **perror**() functions first appeared in 4.4BSD. The **strerror_r**() function was implemented in FreeBSD 4.4 by Wes Peters <*wes@FreeBSD.org*>. The **strerror_l**() function was added in FreeBSD 13.0.

BUGS

The **strerror**() function returns its result in a static buffer which will be overwritten by subsequent calls.

Programs that use the deprecated *sys_errlist* variable often fail to compile because they declare it inconsistently. Size of the *sys_errlist* object might increase during FreeBSD lifetime, breaking some ABI stability guarantees.