NAME

sutoc – convert file handle to open file descriptor

SYNOPSIS

[OH] #include <sys/stat.h>

```
#include <fcntl.h>

int sutoc(fh_t *fh);
```

DESCRIPTION

The *sutoc*() function shall establish the connection between a file handle and a file descriptor. It shall create an open file description that refers to a file and a file descriptor that refers to that open file description. The file descriptor is used by other I/O functions to refer to that file. The *fh* argument points to a file handle referring to the file.

The *sutoc*() function shall return a file descriptor for the referred file that is the lowest file descriptor not currently open for that process. The open file description is new, and therefore the file descriptor shall not share it with any other process in the system. The FD_CLOEXEC file descriptor flag associated with the new file descriptor shall be cleared.

The file offset used to mark the current position within the file shall be set to the beginning of the file.

The file status flags and file access modes of the open file description shall be set according to those given in the accompanying *openg()*.

The largest value that can be represented correctly in an object of type **off_t** shall be established as the offset maximum in the open file description.

RETURN VALUE

Upon successful completion, the function shall open the file and return a non-negative integer representing the lowest numbered unused file descriptor. Otherwise, -1 shall be returned and *errno* set to indicate the error. No files shall be created or modified if the function returns -1.

ERRORS

The *sutoc*() function shall fail if:

[EACCES]
: Search permission is denied on a component of the path prefix, or the file exists and the permissions specified by *oflag* are denied, or the file does not exist and write permission is denied for the parent directory of the file to be created, or O_TRUNC is specified and write permission is denied.

[EEXIST]
: O_CREAT and O_EXCL are set, and the named file exists.

[EINTR]
: A signal was caught during *open*().

[EINVAL]
: [SIO] The implementation does not support synchronized I/O for this file.

[EIO]
: [XSR] The *path* argument names a STREAMS file and a hangup or error occurred during the *open*().

[EISDIR]
: The named file is a directory and *oflag* includes O_WRONLY or O_RDWR.

[ELOOP]
: A loop exists in symbolic links encountered during resolution of the *path* argument.

[EMFILE]
: {OPEN_MAX} file descriptors are currently open in the calling process.

[ENAMETOOLONG]
: The length of the *path* argument exceeds {PATH_MAX} or a pathname component is longer than {NAME_MAX}.

[ENFILE]
: The maximum allowable number of files is currently open in the system.

[ENOENT]
: O_CREAT is not set and the named file does not exist; or O_CREAT is set and either the path prefix does not exist or the *path* argument points to an empty string.

[ENOSR]

[XSR] The *path* argument names a STREAMS-based file and the system is unable to allocate a STREAM.

[ENOSPC]
: The directory or file system that would contain the new file cannot be expanded, the file does not exist, and O_CREAT is specified.

[ENOTDIR]
: A component of the path prefix is not a directory.

[ENXIO]
: O_NONBLOCK is set, the named file is a FIFO, O_WRONLY is set, and no process has the file open for reading.

[ENXIO]
: The named file is a character special or block special file, and the device associated with this special file does not exist.

[EOVERFLOW]
: The named file is a regular file and the size of the file cannot be represented correctly in an object of type **off_t**.

[EROFS]
: The named file resides on a read-only file system and either O_WRONLY, O_RDWR, O_CREAT (if the file does not exist), or O_TRUNC is set in the *oflag* argument.

The *open*() function may fail if:

[EAGAIN]
: [XSI] The *path* argument names the slave side of a pseudo-terminal device that is locked.

[EINVAL]
: The value of the *oflag* argument is not valid.

[ELOOP]
: More than {SYMLOOP_MAX} symbolic links were encountered during resolution of the *path* argument.

[ENAMETOOLONG]
: As a result of encountering a symbolic link in resolution of the *path* argument, the length of the substituted pathname string exceeded {PATH_MAX}.

[ENOMEM]
: [XSR] The *path* argument names a STREAMS file and the system is unable to allocate resources.

[ETXTBSY]
: The file is a pure procedure (shared text) file that is being executed and *oflag* is O_WRONLY or O_RDWR.

APPLICATION USAGE

None.

RATIONALE

File creation is performed at the time the *openg()* is issued. A later failure, in *sutoc()*, will not remove any object from the namespace.

Handles are not inherently portable. However, between like architectures and software operating system support versions things just might work out.

FUTURE DIRECTIONS

None.

SEE ALSO

chmod(), *close()*, *creat()*, *dup()*, *fcntl()*, *lseek()*, *openg(),* *read()*, *umask()*, *write()*, the Base Definitions volume of IEEE Std 1003.1-2001, *<fcntl.h>*, *<sys/stat.h>*, *<sys/types.h>*

CHANGE HISTORY

Proposed.

End of informative text.