NAME
uselib − load shared library

SYNOPSIS
#include <unistd.h>

int uselib(const char ***library);**

Note: No declaration of this system call is provided in glibc headers; see NOTES.

DESCRIPTION
The system call **uselib**() serves to load a shared library to be used by the calling process. It is given a pathname. The address where to load is found in the library itself. The library can have any recognized binary format.

RETURN VALUE
On success, zero is returned. On error, −1 is returned, and *errno* is set appropriately.

ERRORS
In addition to all of the error codes returned by open(2) and mmap(2), the following may also be returned:

EACCES
> The library specified by *library* does not have read or execute permission, or the caller does not have search permission for one of the directories in the path prefix. (See also path_resolution(7).)

ENFILE
> The system-wide limit on the total number of open files has been reached.

ENOEXEC
> The file specified by *library* is not an executable of a known type; for example, it does not have the correct magic numbers.

CONFORMING TO
uselib() is Linux-specific, and should not be used in programs intended to be portable.

NOTES
This obsolete system call is not supported by glibc. No declaration is provided in glibc headers, but, through a quirk of history, glibc versions before 2.23 did export an ABI for this system call. Therefore, in order to employ this system call, it was sufficient to manually declare the interface in your code; alternatively, you could invoke the system call using syscall(2).

In ancient libc versions, **uselib**() was used to load the shared libraries with names found in an array of names in the binary.

Since libc 4.3.2, startup code tries to prefix these names with "/usr/lib", "/lib" and "" before giving up. In libc 4.3.4 and later these names are looked for in the directories found in **LD_LIBRARY_PATH**, and if not found there, prefixes "/usr/lib", "/lib" and "/" are tried.

From libc 4.4.4 on only the library "/lib/ld.so" is loaded, so that this dynamic library can load the remaining libraries needed (again using this call). This is also the state of affairs in libc5.

glibc2 does not use this call.

Since Linux 3.15, this system call is available only when the kernel is configured with the **CONFIG_USELIB** option.

SEE ALSO
ar(1), **gcc(1),** ld(1), ldd(1), mmap(2), open(2), dlopen(3), capabilities(7), ld.so(8)

COLOPHON
This page is part of release 4.16 of the Linux *man-pages* project. A description of the project, information about reporting bugs, and the latest version of this page, can be found at https://www.kernel.org/doc/man−pages/.