ioctl(2) — Linux manual page

NAME | SYNOPSIS | DESCRIPTION | RETURN VALUE | ERRORS | CONFORMING TO | NOTES | SEE ALSO | COLOPHON

Search online pages

IOCTL(2) Linux Programmer's Manual IOCTL(2)

NAME top

ioctl - control device

SYNOPSIS top

#include <sys/ioctl.h>

int ioctl(int *fd*, **unsigned long** *request*, **...);**

DESCRIPTION top

The **ioctl**() system call manipulates the underlying device parameters of special files. In particular, many operating characteristics of character special files (e.g., terminals) may be controlled with **ioctl**() requests. The argument *fd* must be an open file descriptor.

The second argument is a device-dependent request code. The third argument is an untyped pointer to memory. It's traditionally **char** **argp* (from the days before **void *** was valid C), and will be so named for this discussion.

An **ioctl**() *request* has encoded in it whether the argument is an *in* parameter or *out* parameter, and the size of the argument *argp* in bytes. Macros and defines used in specifying an **ioctl**() *request* are located in the file *<sys/ioctl.h>*. See NOTES.

RETURN VALUE top

Usually, on success zero is returned. A few **ioctl**() requests use the return value as an output parameter and return a nonnegative value on success. On error, -1 is returned, and *errno* is set to indicate the error.

ERRORS top

EBADF *fd* is not a valid file descriptor.

EFAULT *argp* references an inaccessible memory area.

EINVAL *request* or *argp* is not valid.

ENOTTY *fd* is not associated with a character special device.

ENOTTY The specified request does not apply to the kind of object that the file descriptor *fd* references.

CONFORMING TO top

No single standard. Arguments, returns, and semantics of **ioctl**() vary according to the device driver in question (the call is used as a catch-all for operations that don't cleanly fit the UNIX stream I/O model).

The **ioctl**() system call appeared in Version 7 AT&T UNIX.

NOTES top

In order to use this call, one needs an open file descriptor. Often the open(2) call has unwanted side effects, that can be avoided under Linux by giving it the **O_NONBLOCK** flag.

ioctl structure

Ioctl command values are 32-bit constants. In principle these constants are completely arbitrary, but people have tried to build some structure into them.

The old Linux situation was that of mostly 16-bit constants, where the last byte is a serial number, and the preceding byte(s) give a type indicating the driver. Sometimes the major number was used: 0x03 for the **HDIO_*** ioctls, 0x06 for the **LP*** ioctls. And sometimes one or more ASCII letters were used. For example, **TCGETS** has value 0x00005401, with 0x54 = 'T' indicating the terminal driver, and **CYGETTIMEOUT** has value 0x00435906, with 0x43 0x59 = 'C' 'Y' indicating the cyclades driver.

Later (0.98p5) some more information was built into the number. One has 2 direction bits (00: none, 01: write, 10: read, 11: read/write) followed by 14 size bits (giving the size of the argument), followed by an 8-bit type (collecting the ioctls in groups for a common purpose or a common driver), and an 8-bit serial number.

The macros describing this structure live in *<asm/ioctl.h>* and are **_IO(type,nr)** and **{_IOR,_IOW,_IOWR}(type,nr,size)**. They use *sizeof(size)* so that size is a misnomer here: this third argument is a data type.

Note that the size bits are very unreliable: in lots of cases they are wrong, either because of buggy macros using *sizeof(sizeof(struct))*, or because of legacy values.

Thus, it seems that the new structure only gave disadvantages: it does not help in checking, but it causes varying values for the various architectures.

SEE ALSO top

execve(2), fcntl(2), ioctl_console(2), ioctl_fat(2), ioctl_ficlonerange(2), ioctl_fideduperange(2), ioctl_fslabel(2), ioctl_getfsmap(2), ioctl_iflags(2), ioctl_ns(2), ioctl_tty(2), ioctl_userfaultfd(2), open(2), sd(4), tty(4)

COLOPHON top

This page is part of release 5.13 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/.

Linux **2021-03-22** **IOCTL(2)**

Pages that refer to this page: apropos(1), man(1), whatis(1), getsockopt(2), ioctl_console(2), ioctl_fat(2), ioctl_ficlonerange(2), ioctl_fideduperange(2), ioctl_fslabel(2), ioctl_getfsmap(2), ioctl_iflags(2), ioctl_ns(2), ioctl_tty(2), ioctl_userfaultfd(2), ioctl_xfs_ag_geometry(2), ioctl_xfs_bulkstat(2), ioctl_xfs_fsbulkstat(2), ioctl_xfs_fscounts(2), ioctl_xfs_fsgeometry(2), ioctl_xfs_fsgetxattr(2), ioctl_xfs_fsinumbers(2), ioctl_xfs_getbmapx(2), ioctl_xfs_getresblks(2), ioctl_xfs_goingdown(2), ioctl_xfs_inumbers(2), ioctl_xfs_scrub_metadata(2), open(2), perf_event_open(2), prctl(2), read(2), seccomp_unotify(2), socket(2), syscalls(2), timerfd_create(2), userfaultfd(2), write(2), errno(3), if_nameindex(3), if_nametoindex(3), openpty(3), dsp56k(4), fd(4), loop(4), lp(4), random(4), rtc(4), sd(4), smartpqi(4), st(4), tty(4), vcs(4), arp(7), capabilities(7), inotify(7), namespaces(7), pipe(7), pty(7), signal(7), socket(7), tcp(7), termio(7), udp(7), unix(7), systemd-makefs@.service(8)

HTML rendering created 2021-08-27 by Michael Kerrisk, author of *The Linux Programming Interface*, maintainer of the Linux *man-pages* project.