NAME

vfs_getopt, **vfs_getopts**, **vfs_flagopt**, **vfs_scanopt**, **vfs_copyopt**, **vfs_filteropt**, **vfs_setopt**, **vfs_setopt_part**, **vfs_setopts -** manipulate mount options and their values

SYNOPSIS

#include <sys/param.h>
#include <sys/mount.h>

int
vfs_getopt(*struct vfsoptlist *opts*, *const char *name*, *void **buf*, *int *len*);

*char **
vfs_getopts(*struct vfsoptlist *opts*, *const char *name*, *int *error*);

int
vfs_flagopt(*struct vfsoptlist *opts*, *const char *name*, *uint64_t *flags*, *uint64_t flag*);

int
vfs_scanopt(*struct vfsoptlist *opts*, *const char *name*, *const char *fmt*, *...*);

int
vfs_copyopt(*struct vfsoptlist *opts*, *const char *name*, *void *dest*, *int len*);

int
vfs_filteropt(*struct vfsoptlist *opts*, *const char **legal*);

int
vfs_setopt(*struct vfsoptlist *opts*, *const char *name*, *void *value*, *int len*);

int
vfs_setopt_part(*struct vfsoptlist *opts*, *const char *name*, *void *value*, *int len*);

int
vfs_setopts(*struct vfsoptlist *opts*, *const char *name*, *const char *value*);

DESCRIPTION

The **vfs_getopt**() function sets *buf* to point to the value of the named mount option, and sets *len* to the length of the value if it is not NULL. The *buf* argument will point to the actual value, and does not need to be freed or released (and probably should not be modified).

The **vfs_getopts**() function returns the value of the specified option if it is a string (i.e., NUL terminated).

The **vfs_flagopt**() function determines if an option exists. If the option does exist, and *flags* is not NULL, *flag* is added to those already set in *flags*. If the option does not exist, and *flags* is not NULL, *flag* is removed from those already set in *flags*. An example of typical usage is:

```
if (vfs_flagopt(mp->mnt_optnew, "wormlike", NULL, 0))
        vfs_flagopt(mp->mnt_optnew, "appendok", &(mp->flags), F_APPENDOK);
```

The **vfs_scanopt**() function performs a vsscanf(3) with the option's value, using the given format, into the specified variable arguments. The value must be a string (i.e., NUL terminated).

The **vfs_copyopt**() function creates a copy of the option's value. The *len* argument must match the length of the option's value exactly (i.e., a larger buffer will still cause **vfs_copyout**() to fail with EINVAL).

The **vfs_filteropt**() function ensures that no unknown options were specified. A option is valid if its name matches one of the names in the list of legal names. An option may be prefixed with 'no', and still be considered valid.

The **vfs_setopt**() and **vfs_setopt_part**() functions copy new data into the option's value. In **vfs_setopt**(), the *len* argument must match the length of the option's value exactly (i.e., a larger buffer will still cause **vfs_copyout**() to fail with EINVAL).

The **vfs_setopts**() function copies a new string into the option's value. The string, including NUL byte, must be no longer than the option's length.

RETURN VALUES

The **vfs_getopt**() function returns 0 if the option was found; otherwise, ENOENT is returned.

The **vfs_getopts**() function returns the specified option if it is found, and is NUL terminated. If the option was found, but is not NUL terminated, *error* is set to EINVAL and NULL is returned. If the option was not found, *error* is set to 0, and NULL is returned.

The **vfs_flagopt**() function returns 1 if the option was found, and 0 if it was not.

The **vfs_scanopt**() function returns 0 if the option was not found, or was not NUL terminated; otherwise, the return value of vsscanf(3) is returned. If vsscanf(3) returns 0, it will be returned unchanged; therefore, a return value of 0 does not always mean the option does not exist, or is not a valid string.

The **vfs_copyopt**() and **vfs_setopt**() functions return 0 if the copy was successful, EINVAL if the option was found but the lengths did not match, and ENOENT if the option was not found.

The **vfs_filteropt**() function returns 0 if all of the options are legal; otherwise, EINVAL is returned.

The **vfs_setopts**() function returns 0 if the copy was successful, EINVAL if the option was found but the string was too long, and ENOENT if the option was not found.

AUTHORS

This manual page was written by Chad David <*davidc@FreeBSD.org*> and Ruslan Ermilov <*ru@FreeBSD.org*>.