## NAME

openproc, closeproc  – initialize process information from /proc/

## SYNOPSIS

**#include <proc/readproc.h>**

**PROCTAB\* openproc (int** *flags***, ... );**
**void closeproc (PROCTAB\*** *PT***);**

## SYNOPSIS

The **openproc** function initializes a PROCTAB structure which can be used by iterated readproc calls to get information on current processes. Depending on *flags*, openproc may need a second argument or a second and third argument (see below).

**closeproc** closes all files opened by **openproc** and deallocates the memory allocated by **openproc.**

The PROCTAB structure is defined in *<proc/readproc.h>*

## RETURN VALUE

**openproc** returns a pointer to a PROCTAB structure, or NULL if an error occurs. This usually means that */proc* cannot be read by the process.

## FLAGS

The behaviour of **openproc** is controlled by the following set of flags, which may be ORed together. There are three different kinds of flags. The first group of flags determines which information gets read from /proc/#pid for each process. The second group of flags (of which only one can be enacted for a opendir call) restricts which processes information is read for by providing a list of criteria. The third group of flags restricts this as well, but doesn't need arguments. These may be used together again.

**PROC_FILLMEM**
>       read information from */proc/#pid/statm*

**PROC_FILLCOM**
>       allocate *cmdline* part of *proc_t* and read information from */proc/#pid/cmdline*

**PROC_FILLENV**
>       allocate *environ* part of *proc_t* and read information from */proc/#pid/environ*

**PROC_FILLUSR**
>       resolve user ids to names via */etc/passwd*

**PROC_FILLGRP**
>       resolve group ids to names via */etc/group*

**PROC_FILLSTATUS**
>       read information from */proc/#pid/status*

**PROC_FILLSTAT**
>       read information from */proc/#pid/stat*

**PROC_FILLARG**
>       equivalent to PROC_FILLCOM

**PROC_FILLCGROUP**
>       alloc and fill in cgroup

**PROC_FILLSUPGRP**
>       resolve supplementary group id -> group name

**PROC_FILLOOM**
>    fill in proc_t oom_score and oom_adj

**PROC_FILLNS**
>    fill in proc_t namespace information

**PROC_FILLSYSTEMD**
>    fill in proc_t systemd information

**PROC_LOOSE_TASKS**
>    threat threads as if they were processes

**PROC_PID** (2nd argument **pid_t\*** *pidlist*)
>    lookup only processes whose pid is contained in *pidlist* (the list is terminated with 0)

**PROC_UID** (arguments **uid_t\*** *uidlist*, **int** *n*)
>    lookup only processes whose user id is contained in *uidlist* (where *n* is the number of uids contained in the list)

**PROC_EDITCGRPCVT**
>    edit cgroup as single vector

**PROC_EDITCMDLCVT**
>    edit cmdline as single vector

**PROC_EDITENVRCVT**
>    edit environ as single vector

## NOTE
>    Only one of the flags needing additional arguments (**PROC_{PID,UID}**) may be used at a time.

## SEE ALSO
>    **readproc**(3), **readproctab**(3), **/proc/**, **/usr/include/proc/readproc.h**,

## REPORTING BUGS
>    Please send bug reports to ⟨procps@freelists.org⟩