GETPRIORITY(2) Manuel du programmeur Linux GETPRIORITY(2)

NOM
getpriority, setpriority - Lire / écrire la priorité d'ordonnancement du processus.

SYNOPSIS
```
#include <sys/time.h>
#include <sys/resource.h>

int getpriority(int which, int who);
int setpriority(int which, int who, int prio);
```

DESCRIPTION
La priorité d'ordonnancement du processus, du groupe de processus ou de l'utilisateur, comme précisé dans which et who est lue avec getpriority et fixée avec setpriority. Which doit être l'un des éléments PRIO_PROCESS, PRIO_PGRP, ou PRIO_USER, et who est interprété en fonction de which (un ID de processus pour PRIO_PROCESS, un ID de groupe de processus pour PRIO_PGRP, et un ID d'utilisateur pour PRIO_USER).

Une valeur nulle pour who indique le processus, groupe ou utilisateur courant. Prio est une valeur dans l'intervalle -20 à 20. La priorité par défaut est 0, les priorités les plus faibles indiquant un ordonnancement le plus favorable.

La fonction getpriority retourne la plus haute priorité (la plus basse valeur numérique) dont a bénéficié l'un des processus indiqué. La fonction setpriority fixe la priorité des processus indiqués à la valeur fournie. Seul le Super-User peut diminuer la valeur numérique de la priorité (favoriser le processus).

VALEUR RENVOYÉE
Comme getpriority peut tout à fait renvoyer la valeur -1, il faut effacer la variable externe errno avant l'appel afin de vérifier si une valeur -1 indique une erreur ou une priorité légitime.

setpriority renvoie 0 s'il réussit, ou -1 s'il échoue, auquel cas errno contient le code d'erreur.

ERREURS
ESRCH Aucun processus ne correspond aux valeurs de which et who.

EINVAL Which n'était ni PRIO_PROCESS, ni PRIO_PGRP, ni PRIO_USER.

De plus setpriority peut échouer pour les erreurs suivantes :

EPERM Un processus correspond bien aux valeurs indiquées, mais ni l'UID réel, ni l'UID effectif de l'appelant ne lui correspondent.

EACCES Tentative de favoriser un processus sans être Super-User.

NOTES Les détails concernant la condition d'erreur EPERM dépendent du système. La description ci-dessus concerne ce que dit SUSv3, et qui semble être suivi par tous les systèmes de type SysV. Linux demande que l'UID réel ou effectif de l'appelant correspondent à l'UID réel du processus who (et non pas à son UID effectif). Tous les systèmes de type BSD (SunOS 4.1.3, Ultrix 4.2 BSD 4.3, FreeBSD 4.3, OpenBSD-2.5...) demandent que l'UID effectif de l'appelant corresponde à l'UID réel ou effectif du processus who.

Le véritable intervalle des priorités varie suivant les versions du noyau. Sur les systèmes Linux antérieure au 1.3.36 l'intervalle s'étendait de -infini à 15. Depuis le 1.3.43 il correspond à -20..19, et l'appel-système getpriority renvoie 40..1 pour ces valeurs (puisque les nombres négatifs sont des codes d'erreur). La routine de bibliothèque convertit N en 20-N.

L'inclusion <sys/time.h> n'est plus obligatoire, mais améliore la portabilité. (En fait, <sys/resource.h> définit la structure rusage qui contient des champs de type struct timeval défini <sys/time.h>).

CONFORMITÉ : SVr4, BSD 4.4 (Cette fonction est apparue dans BSD 4.2).

VOIR AUSSI : nice(1), fork(2), renice(8).

TRADUCTION
Christophe Blaess, 1997.

BSD 16 janvier 2002 GETPRIORITY(2)