NAME

signal, sigset, sighold, sigrelse, sigignore, sigpause – simplified signal management for application processes

SYNOPSIS

#include <signal.h>

void (∗signal (int *sig*, **void (∗***disp***)(int)))(int);**

void (∗sigset (int *sig*, **void (∗***disp***)(int)))(int);**

int sighold(int *sig***);**

int sigrelse(int *sig***);**

int sigignore(int *sig***);**

int sigpause(int *sig***);**

DESCRIPTION

These functions provide simplified signal management for application processes. See **signal**(5) for an explanation of general signal concepts.

signal( ) and **sigset( )** are used to modify signal dispositions. *sig* specifies the signal, which may be any signal except **SIGKILL** and **SIGSTOP**. *disp* specifies the signal's disposition, which may be **SIG_DFL**, **SIG_IGN**, or the address of a signal handler. If **signal( )** is used, *disp* is the address of a signal handler, and *sig* is not **SIGILL**, **SIGTRAP**, or **SIGPWR**, the system first sets the signal's disposition to **SIG_DFL** before executing the signal handler. If **sigset( )** is used and *disp* is the address of a signal handler, the system adds *sig* to the calling process's signal mask before executing the signal handler; when the signal handler returns, the system restores the calling process's signal mask to its state prior to the delivery of the signal. In addition, if **sigset( )** is used and *disp* is equal to **SIG_HOLD**, *sig* is added to the calling process's signal mask and the signal's disposition remains unchanged.

sighold( ) adds *sig* to the calling process's signal mask.

sigrelse( ) removes *sig* from the calling process's signal mask.

sigignore( ) sets the disposition of *sig* to **SIG_IGN**.

sigpause( ) removes *sig* from the calling process's signal mask and suspends the calling process until a signal is received.

RETURN VALUES

On success, **signal( )** returns the signal's previous disposition. On failure, it returns **SIG_ERR** and sets **errno** to indicate the error.

On success, **sigset( )** returns **SIG_HOLD** if the signal had been blocked or the signal's previous disposition if it had not been blocked. On failure, it returns **SIG_ERR** and sets **errno** to indicate the error.

All other functions return zero on success. On failure, they return –1 and set **errno** to indicate the error.

ERRORS

These functions fail if any of the following are true:

EINTR A signal was caught during the function **sigpause()**.

EINVAL The value of the *sig* argument is not a valid signal or is equal to **SIGKILL** or **SIGSTOP**.

SEE ALSO

kill(2), **pause**(2), **sigaction**(2), **sigsend**(2), **wait**(2), **waitid**(2), **signal**(5)

NOTES

sighold() in conjunction with **sigrelse()** or **sigpause()** may be used to establish critical regions of code that require the delivery of a signal to be temporarily deferred.

If **signal()** or **sigset()** is used to set **SIGCHLD**'s disposition to a signal handler, **SIGCHLD** will not be sent when the calling process's children are stopped or continued.

If any of the above functions are used to set **SIGCHLD**'s disposition to **SIG_IGN**, the calling process's child processes will not create zombie processes when they terminate (see **exit**(2)). If the calling process subsequently waits for its children, it blocks until all of its children terminate; it then returns a value of −1 with **errno** set to **ECHILD** (see **wait**(2), **waitid**(2)).