NAME

wait – wait for child process to stop or terminate

SYNOPSIS

#include <sys/types.h>
#include <sys/wait.h>

pid_t wait(int **stat_loc***);**

MT-LEVEL

Async-Signal-Safe

DESCRIPTION

wait( ) suspends the calling process until one of its immediate children terminates or until a child that is being traced stops because it has received a signal. The **wait( )** function will return prematurely if a signal is received. If any unawaited process stopped or terminated prior to the call on **wait( )**, return is immediate.

If **wait( )** returns because the status of a child process is available, it returns the process ID of the child process. If the calling process had specified a non-zero value for *stat_loc*, the status of the child process will be stored in the location pointed to by *stat_loc*. It may be evaluated with the macros described on **wstat**(5). In the following, *status* is the object pointed to by *stat_loc*:

> If the child process stopped, the high order 8 bits of *status* will contain the number of the signal that caused the process to stop and the low order 8 bits will be set equal to **WSTOPFLG**.
>
> If the child process terminated due to an **_exit( )** call, the low order 8 bits of *status* will be 0 and the high order 8 bits will contain the low order 8 bits of the argument that the child process passed to **_exit( );** see **exit**(2).
>
> If the child process terminated due to a signal, the high order 8 bits of *status* will be 0 and the low order 8 bits will contain the number of the signal that caused the termination. In addition, if **WCOREFLG** is set, a “core image” will have been produced; see **signal**(3C).

If **wait( )** returns because the status of a child process is available, then that status may be evaluated with the macros defined by **wstat**(5).

If a parent process terminates without waiting for its child processes to terminate, the parent process ID of each child process is set to 1. This means the initialization process inherits the child processes; see **intro**(2).

RETURN VALUES

When **wait( )** returns due to a terminated child process, the process ID of the child is returned to the calling process. Otherwise, a value of −1 is returned and **errno** is set to indicate the error.

ERRORS

wait( ) will fail if one or both of the following is true:

ECHILD The calling process has no existing unwaited-for child processes.

EINTR The function was interrupted by a signal.

SEE ALSO

intro(2), **exec**(2), **exit**(2), **fork**(2), **pause**(2), **ptrace**(2), **waitid**(2), **waitpid**(2), **signal**(3C), **signal**(5), **wstat**(5)

NOTES

See NOTES in **signal**(3C).

Since **wait( )** will block on a stopped child, if the calling process wishes to see the return results of such a **wait**, it should use **waitid**(2) or **waitpid**(2) instead of **wait( )**.