- 如果 pid>0，那么等待集合就是一个单独的子进程，它的进程 ID 等于 pid。
- 如果 pid=-1，那么等待集合就是由父进程所有的子进程组成的。

waitpid 函数还支持其他类型的等待集合，包括 Unix 进程组，对此我们将不做讨论。

2. 修改默认行为

可以通过将 options 设置为常量 WNOHANG、WUNTRACED 和 WCONTINUED 的各种组合来修改默认行为：

- WNOHANG：如果等待集合中的任何子进程都还没有终止，那么就立即返回(返回值为 0)。默认的行为是挂起调用进程，直到有子进程终止。在等待子进程终止的同时，如果还想做些有用的工作，这个选项会有用。
- WUNTRACED：挂起调用进程的执行，直到等待集合中的一个进程变成已终止或者被停止。返回的 PID 为导致返回的已终止或被停止子进程的 PID。默认的行为是只返回已终止的子进程。当你想要检查已终止和被停止的子进程时，这个选项会有用。
- WCONTINUED：挂起调用进程的执行，直到等待集合中一个正在运行的进程终止或等待集合中一个被停止的进程收到 SIGCONT 信号重新开始执行。(8.5 节会解释这些信号。)

可以用或运算把这些选项组合起来。例如：

- WNOHANG | WUNTRACED：立即返回，如果等待集合中的子进程都没有被停止或终止，则返回值为 0；如果有一个停止或终止，则返回值为该子进程的 PID。

3. 检查已回收子进程的退出状态

如果 statusp 参数是非空的，那么 waitpid 就会在 status 中放上关于导致返回的子进程的状态信息，status 是 statusp 指向的值。wait.h 头文件定义了解释 status 参数的几个宏：

- WIFEXITED(status)：如果子进程通过调用 exit 或者一个返回(return)正常终止，就返回真。
- WEXITSTATUS(status)：返回一个正常终止的子进程的退出状态。只有在 WIFEXITED()返回为真时，才会定义这个状态。
- WIFSIGNALED(status)：如果子进程是因为一个未被捕获的信号终止的，那么就返回真。
- WTERMSIG(status)：返回导致子进程终止的信号的编号。只有在 WIFSIGNALED()返回为真时，才定义这个状态。
- WIFSTOPPED(status)：如果引起返回的子进程当前是停止的，那么就返回真。
- WSTOPSIG(status)：返回引起子进程停止的信号的编号。只有在 WIFSTOPPED()返回为真时，才定义这个状态。
- WIFCONTINUED(status)：如果子进程收到 SIGCONT 信号重新启动，则返回真。

4. 错误条件

如果调用进程没有子进程，那么 waitpid 返回－1，并且设置 errno 为 ECHILD。如果 waitpid 函数被一个信号中断，那么它返回－1，并设置 errno 为 EINTR。

旁注　和 Unix 函数相关的常量

像 WNOHANG 和 WUNTRACED 这样的常量是由系统头文件定义的。例如，WNOHANG 和 WUNTRACED 是由 wait.h 头文件(间接)定义的：