- fork()
- exec()
- wait()

fork() creates new process (child), copy of the calling process (parent)
fork() returns pid of the newly-created child, the child receives a return code of zero.

wait() - The process waits for the child process to finish
* Can return early in case of -
  - WNOHANG option is used (non-blocking behavior)
  - Interrupted by a signal (EINTR)
  - Child process changes state (stop/resume) with specific flags like `WUNTRACED' or 'WCONTINUED'
  - Child process has already exited but its status has not been reaped

(zombie process)

exec() - Runs the given executable like "wc". By loading code (and static data) basically transforming the current running program into that. Successful call to exec() never returns.

Motivating the API

Unix shell ⇒ fork() + run some code + exec
change env.

Example
prompt> wc p3.c > newfile.txt

shell → fork ①
②
close standard output
and open file

Works since in Unix system start looking for free file descriptors at zero. In this case STDOUT_FILENO will be the first available and thus get assigned when open() is called.

Unix Pipes ⇒ pipe() system call
output of one process is used as input to the next

Process Control and Users

Signals to process ⇒ ctrl-C : SIGINT (normally terminates) (interrupt)
ctrl-Z : SIGSTP (stop)
I

pausing execution can be resumed later