Fork

Permite crear un proceso hijo.

Padre:	PID del hijo
Hijo:	0
Error:	-1

exec

Pone en ejecución un programa. No cambia la identidad del proceso.

execlp	argumentos en lista
execvp	argumentos en vector a punteros (argv[])
herencia	
PID,PPID,PGID,UID GID	variables de entorno
señales pendientes	tabla de descriptores
Diferente	
EUID,EGID	comportamiento ante señales

execlp

execlp(comando,comando,argumento1,....,0)

Comando dos veces

execvp

execvp(argv[1],&argv[1])

strtok(s,"_")

La primera vez devuelve el puntero donde empieza la primera palabra y escribe un \0 donde acaba

Siguientes veces usar null en vez de s.

wait // waitpid(pid_hijo,p_estado,options)

error	-1 (si no hay hijos) // errno =10
Terminación	
voluntaria	byte + significativo
proceso zombie	

wait // waitpid(pid_hijo,p_estado,options) (cont)

Cuando el padre no ha hecho wait

proceso huérfano

Cuando el padre se cierra dejando hijios activos. Son adoptados por el init(pid=1)

Signal(señal,comportamiento)

Señal:	Nombre o número
comportamiento:	
SIG_DFL	0
SIG_IGN	1
rutina_captura	
	puntero a función de parametro entero que devuelve void
Devuelve:	Comportamiento anterior
Capturar señal	Solo una vez

#define SIG_ERR(void(*)(int))-1

kill(pid_señalado,señal)

pid_señalado	
>0	pid destino
=0	todos los del grupo
=-1	Todos los procesos(SU)
<-1	grupo de procesos con valor absoluto
matar a todos	
kill -9 -1	
error:	-1

alarm(sec)

Programa la recepción de sIG_ALARM dentro de sec segundos

alarm(0)	Cancela la alarma que haya (Solo puede haber una)
devuelve:	tiempo de la anterior alarma

Recuerda capturar antes de hacer alarm

pause()

Suspende al proceso hasta la llegada de una señal capturada.

devuelve -1 al capturar señal

Señales seguras

sigset_t	sigemptyset (todo vacío)
	sigfillset(todo lleno
	sigaddset(añadir)
	sigdelset(borrar)
	sigismember(sigset_t,int signo)
int sigprocmask(int how, sigset_t *set,sigset_t* oset)	
how	SIG_BLOCK //añade set al conjunto
	SIG_UNBLOCK // quita set al conjunto
	SIG_SETMASK // pone nuevo set como conjunto
oset	devuelve el conjunto previo de señales
set	devuelve el nuevo
int sigpending(sigset_t*set)	
int sigsuspend(sigset_t* sigmask)	
	operacion atomica para resolver problemas de sincro.
	bloqueo sigusr1
	kill(, sigusr1)
	sigsuspend(desbloqueo)

Identificadores

Usuario	
UID:	id de usuario(0 para su)
GID:	id del grupo
Fichero	número de inodo
i-nodo	
UID:	propietario
GUID:	grupo propietario

By **Cesarblancg**

cheatography.com/cesarblancg/

Published 1st February, 2019.
Last updated 1st February, 2019.

Sponsored by **Readable.com**
Measure your website readability!
https://readable.com

Identificadores (cont)

permisos de acceso(user-group-other)	
set-user-id	usuario efectivo
set-group-id	grupo efectivo
sticky bit	gestión en directorio compartido(\tmp)
Proceso	
PID	pid_t getpid() //id padre
PGID	pid_t getpgrp() //id del grupo
UID	uid_t getuid() // usuario real
GID	gid_t getgid() // id del grupo usuario
EUID	uid_t geteuid() //usuario efectivo
	si(set-user-id==1) then EUID=UID
EGID	gid_t getegid // grupo efectivo

Señales

Por defecto termina el programa	
ctrl+c o ctrl+y no se pueden controlar	
Terminación proceso hijo no se puede capturar	
SIGINT	2
SIGQUIT	3
SIGPIPE	13
SIGALRM	14
SIGTERM	15
SIGUSR[1,2]	16,17
SIGCHLD	18
SIGSTOP	23
SIGCONT	25
SIGSEGV	11

Llamadas sistema

int creat(char*path,mode_t mode)

Crea un fichero o reescribe uno existente. Abierto solo para escritura. Si el fichero existe pierdes los datos.

Devuelve Descriptor de fichero o -1

int open(char*path,int oflag, [mode])

Permite abrir un archivo como lectura, escritura o ambas

[oflag]

O_RDONLY 0

O_WRONLY 1

O_RDWR 2

[mode]

O_APPEND(cursor al final)

O_CREAT(crea fichero)

int close(int fd)

Elimina la entrada TDF y TFA y decrementa nºopens en TI

Devuelve 0 o -1

By **Cesarblancg**

cheatography.com/cesarblancg/

Published 1st February, 2019.
Last updated 1st February, 2019.

Sponsored by **Readable.com**
Measure your website readability!
https://readable.com