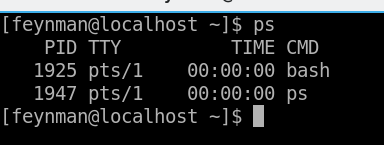
**Capitulo 1: Programación multiproceso**

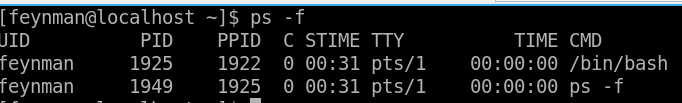
**Apuntes**

Comandos para visualización de procesos (ps en Unix, tasklist en Windows):

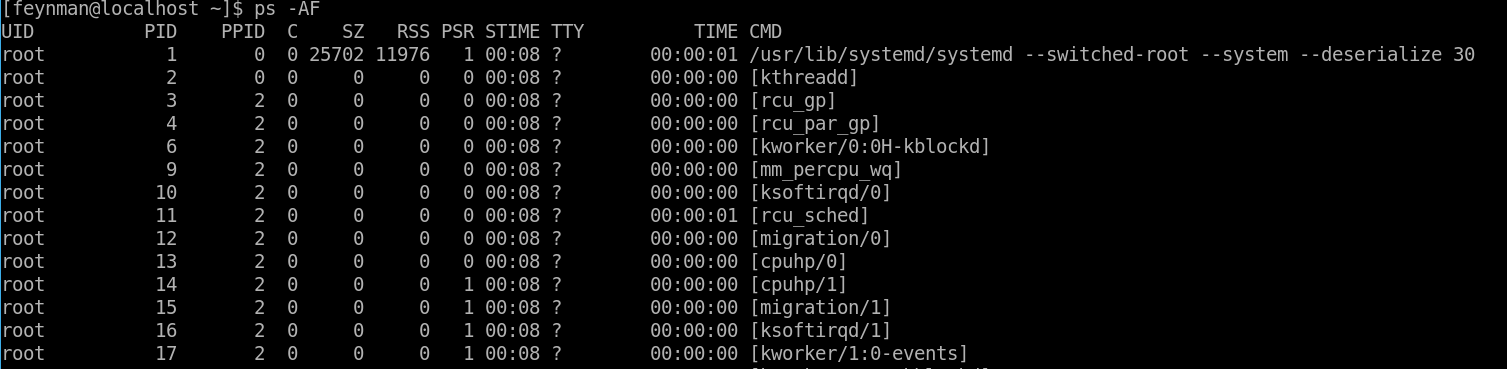
**ps**



**ps -f** (muestra más información)



**ps -AF** (muestra todos los procesos activos con todos los detalles)



**PID**: identificador del proceso.

**TTY**: Terminal asociado del que lee y al que escribe. Si no hay, aparece una interrogación.

**TIME**: Tiempo de ejecución asociado. Es la cantidad total de tiempo de CPU que el proceso ha utilizado desde que nació.

**CMD**: Nombre del proceso.

**UID**: nombre del usuario.

**PPID**: PID del padre de cada proceso.

**C**: Porcentaje de recursos de CPU utilizado por el proceso.

**STIME**: hora de inicio del proceso.

**SZ**: tamaño virtual de la imagen del proceso.

**RSS**: tamaño de la parte residente en memoria en kilobytes.

**PSR**: procesador que el proceso tiene actualmente asignado.

**C**

System(): se encuentra en la librería stdlib.h, por lo que funciona en cualquier sistema operativo que tenga un compilador de C/C++. La sintaxis es la siguiente:

Int system( const char \* cadena)

La función recibe como parámetro una cadena de caracteres que indica el comando que se desea procesar. Dicha instrucción es pasada al intérprete de comandos del ambiente en que se está trabajando y se ejecuta. Devuelve el valor -1 si ocurre un error, y el estado devuelto en caso contrario.

Fork():

Wait(): Se utiliza para que el proceso padre espere la finalización del proceso hijo. El proceso padre quedará bloqueado hasta este suceso. La sintaxis de la orden es:

pid\_t wait(int \* status);

Y la forma más básica de utilizarlo es:

Wait(NULL);