Process Control and Multitasking

Concept: The UNIX kernel can keep track of many processes at once, dividing its time between the jobs submitted to it. Each process submitted to the kernel is given a unique process ID.

Single-tasking operating systems, like DOS, or the Macintosh System, can only perform one job at a time. A user of a single-tasking system can switch to different windows, running different applications, but only the application that is currently being used is active. Any other task that has been started is suspended until the user switches back to it. A suspended job receives no operating system resources, and stays just as it was when it was suspended. When a suspended job is reactivated, it begins right where it left off, as if nothing had happened.

The UNIX operating system will simultaneously perform multiple tasks for a single user. Activating an application does not have to cause other applications to be suspended.

Actually, it only appears that UNIX is performing the jobs simultaneously. In reality, it is running only one job at a time, but quickly switching between all of its ongoing tasks. The UNIX kernel will execute some instructions from job A, and then set job A aside, and execute instructions from job B. The concept of switching between queued jobs is called process scheduling.

Viewing processes

UNIX provides a utility called ps (process status) for viewing the status of all the unfinished jobs that have been submitted to the kernel. The ps command has a number of options to control which processes are displayed, and how the output is formatted.
Example: Type the command

Code:

```
ps
```

to see the status of the "interesting" jobs that belong to you. The output of the ps command, without any options specified, will include the process ID, the terminal from which the process was started, the amount of time the process has been running, and the name of the command that started the process.
Example: Type the command

Code:

```
ps -ef
```

to see a complete listing of all the processes currently scheduled. The -e option causes ps to include all processes (including ones that do not belong to you), and the -f option causes ps to give a long listing. The long listing includes the process owner, the process ID, the ID of the parent process, processor utilization, the time of submission, the process's terminal, the total time for the process, and the command that started the process.

Example: Use the ps command, and the grep command, in a pipeline to find all the processes owned by you.

Explanation: The command

Code:

```
ps -ef | grep yourusername
```

where "yourusername" is replaced by your user name, will cause the output of the ps -ef command to be filtered for those entries that contain your username.

Killing processes

Occasionally, you will find a need to terminate a process. The UNIX shell provides a utility called kill to terminate processes. You may only terminate processes that you own (i.e., processes that you started). The syntax for the kill command is kill [-options] process-ID.
To kill a process, you must first find its process ID number using the ps command. Some processes refuse to die easily, and you can use the "-9" option to force termination of the job.

Example: To force termination of a job whose process ID is 111, enter the command

Source: http://www.go4expert.com/articles/process-control-multitasking-t840/