Process Control

When you type in a command on the command line and hit the Enter key, the operating system initiates a process, creating an instance of the command you specified and passing the arguments to that instance of the command. This is like what happens when you choose a program to run off the Windows Start menu or double-click on an icon on your Windows desktop. However, most command-line programs do not persist like you are used to seeing with graphical programs you invoke in Windows; command-line programs generally will run until they are finished, and then terminate themselves, and the shell will respond by displaying a prompt and wait for your next input.

You can run multiple processes at once, maintaining control of them all, and being able to determine the status of each process. Today's lab is about managing multiple processes from the command line.

Jobs

Jobs are processes that you invoke using commands at the shell's command line.

Invoking a Process/Job

Type the following at the prompt and press the Enter key:

```
% sleep 60
```

As we saw earlier, this will cause your shell to wait for 60 seconds, and then generate another command prompt and wait for your input.

Suspending a Running Job

We won't wait the full 60 seconds. What we will do is to suspend this process. Do this by hitting control-Z before the 60 seconds have passed; you'll see the following.

```
^Z[1] + Suspended                 sleep 60
%
```

Hit control-Z by holding down the **Ctrl** key while hitting the **Z** key. Don't hit **shift-6**, then the **Z** key, or just the **Z** key.

The sleep command you started created a sleep process, which is now suspended. It is not doing anything right now. You now have one job. To see a list of the jobs you have, type:

```
% jobs
[1] + Suspended                sleep 60
%
```

This shows you have one job, the sleep process you suspended.

Each job has a number. The number of the suspended job is 1; you can see the job's number at the left

side of the line describing the job, which includes:

- its number
- an optional +/- designator (we'll get to this in just a minute)
- a status indicator
- the command associated with the job

The 60 second limit will pass, and the process will still be there. Try the jobs command after 60 seconds to demonstrate this.

Resuming a Suspended Job

To resume the job you suspended, type the following:

```
% fg %1
sleep 60
%
```

This told the operating system to resume execution of the sleep command (job number 1 in the list of jobs you saw with the jobs command).

Run the jobs command:

```
% jobs
%
```

You have no jobs. The sleep command has completed its operation and has terminated.

The Foreground Job

Try this again. Start a 60-second sleep again. Notice you do not have a prompt; the shell is executing the sleep command and will not wait for a new command line from you until the sleep command terminates. Hit control-Z to suspend the process. Run the jobs command to determine the number of the suspended job. Type fg %# at the prompt to resume the job, where # is the number of the suspended job. Notice how the shell is still not accepting new commands from you while the sleep command is actually running; this is the case when a command is being run in the *foreground*. That is what the **fg** command stands for: foreground.

Background Jobs

You can only run one job in the foreground at a time. You can run one or more jobs in the background. You can do this by:

```
% sleep 60
^Z[1] + Suspended                sleep 60
% jobs
```

```
[1] + Suspended                 sleep 60
% bg %1
[1] sleep 60
% jobs
[1] + Running                   sleep 60
%
```

The bg command resumed your suspended job, but it is now running in the background.

Notice the change in status of your job: it is now marked as running. Yet, the shell is presenting you with the command prompt, which means you could start another job even though another one is running in the background.

Hit the Enter key a few times now. You should eventually see the following:

```
%
[1]   Done                      sleep 60
%
```

Just before presenting a command prompt, the shell will check the list of jobs for any jobs that have changed status, and will report those to you. Here, the shell lets you know that your background job has terminated.

Terminating the Foreground Job

Start another sleep 60 command. Type control-C. You will see:

```
% sleep 60
^C%
```

This terminated the job in the foreground.

Terminating a Suspended or Background Job

(Not all shells support this; I think the one you are using does.) Start another sleep 60 command. Suspend it. Run it in the background. Type the following at the command prompt:

```
% kill %1
[1]    Terminated                    sleep 60
%
```

This terminated your job prior to the job completing on its own.

Invoking a Job Directly Into the Background

Try the following:

```
% sleep 60 &
[1] 21544
%
```

This started a sleep command, but the presence of the ampersand at the end of the command line indicates to the shell that you want to start the job in the background. Run the jobs command:

```
% jobs
[1]  + Running                 sleep 60
%
```

Logging Out With Suspended Jobs

Start a sleep 60. Suspend it. Attempt to log out.

```
% exit
There are suspended jobs.
%
```

The shell warns you if you attempt to exit/logout and you have suspended jobs. If you repeat the exit command as your next command, the shell will allow you to exit/logout even though there are suspended jobs; the shell will terminate the jobs before exiting.

Additional Practice With Jobs

Try creating a number of processes using the sleep command; use different amounts of sleep time to distinguish them. Start them in the foreground or the background.

Suspend one, some, or all of them.

Obtain the list of jobs and their status.

Run one, some, or all of them in the background.

Terminate a foreground job.

Terminate one or more background jobs.

Moving a Job from One State to Another

To start a job in the foreground, type the command you want to run and press Enter.

To start a job in the background, type the command you want to run, put an ampersand & at the end, and press enter.

To suspend a job running in the foreground, press control-Z.

To run a suspended job in the background, run the *jobs* command to find out the number of the suspended job, then run the *bg* command as `bg %#`, substituting the number of the suspended job for the

#.

To run a suspended or background job in the foreground, run the *jobs* command to find out the number of the suspended or background job, then run the *fg* command as `fg %#`, substituting the number of the suspended or background job for the #.

Processes

Processes are "workers" that the operating system creates and manages for you in order to perform various operations. In addition to processes associated with shell jobs, processes may be started at boot time, started at other times but not by commands given to the shell, or started by other processes that make calls to the kernel API.

Listing Processes

Type the following – you'll see something like this:

```
% ps
  PID  TT  STAT      TIME COMMAND
21589  p2  S      0:00.06 csh
%
```

The ps command gives you process status. It lists your processes. Each process has an identifying number that is unique among all active processes, called its *pid* (process ID).

Terminating a Process

You can terminate a process by using the *kill* command, supplying the PID of the process you want to kill as the argument.

More Practice

Start up some *sleep* commands. Suspend them or put them in the background. Run the *ps* command to see the information reported about these processes. Kill a few of them.

Another Way of Logging Out

This uses the *kill* command. What you will do is to kill your login shell. First, find out the process ID of your login shell by running the *ps* command, and looking for the PID associated with the *csh* command. Then, run the following command:

```
% kill -KILL pid
```

replacing *pid* with the actual PID of your login shell you obtained from the *ps* command.