

Re: parsing ps -ax output and killing processes

Source: <http://unix.derkeiler.com/Newsgroups/comp.unix.shell/2005-02/0607.html>

From: Icarus Sparry (usenet_at_icarus.freeuk.com)

Date: 02/15/05

Date: Tue, 15 Feb 2005 12:42:28 GMT

On Tue, 15 Feb 2005 11:58:00 +0000, Rob wrote:

> *After the success of writing my first script yesterday (thanks guys!) I'm*
> *motivated to move on to something more complex today. How would I go about*
> *doing something like a*
>
> *ps -ax | grep xxxx*
>
> *and then killing any processes listed (except for the one running the*
> *command of course). There may not be any processes, I was thinking to*
> *detect that I could write out to a file and check the file size, but I*
> *guess that doesn't really help me if there are any processes listed.*
> *Basically I run a shutdown script which doesn't always work, then need to*
> *verify that the script did work by seeing if processes are still running*
> *and manually killing them if they aren't.*

It is more normal to use 'awk' here, rather than grep, as you need to not only select particular lines, you then need to manipulate them.

Awk is pretty easy to use, and has all the searching of grep built in.

However since this is your second script, let us proceed in small steps.

You already know you want to do something like 'ps -ax | grep something' and then manipulate the result. If there is nothing left then do nothing, otherwise run kill on it. The 'test' command (also known as '[') can tell if a variable is null or not, and it can also test to see if a file is empty. Modern shells such as ksh, bash, zsh have a syntax element called '[' which handles things a little better.

So one thing you could do is

```
tokill=$(ps -ax | grep something | extract_just_the_first_column)
if [[ -n $tokill ]]
then
    kill $tokill
fi
```

comp.unix.shell: Re: parsing ps -ax output and killing processes

A command for extract_just_the_first_column is
awk '{print \$1}'

You also say you don't want to kill off the commands doing the killing. A way of doing this is to use 'grep -v pattern' which selects the lines which do not match. putting this together

```
tokill=$(ps -ax | grep something | grep -v grep | awk '{print $1}')
```

If you want the "something" to be fixed, then you have it. Otherwise you can use the positional parameters.

```
#!/bin/sh
tokill=$(ps -ax | grep -- "$1" | grep -v grep | awk '{print $1}')
if [[ -n $tokill ]]
then
    kill $tokill
fi
```