Screen or as I like to refer to it “Admin’s little helper”
Screen is a window manager that multiplexes a physical terminal between several processes

here are a couple quick reasons you’d might use screen

Lets say you have a unreliable internet connection you can use screen and if you get knocked out from your current session you can always connect back to your session.

Or let’s say you need more terminals, instead of opening a new terminal or a new tab just create a new terminal inside of screen

Here are the screen shortcuts to help you on your way Screen shortcuts

and here are some of the Top 10 Awesome Linux Screen tips urfix.com uses all the time if not daily.

1) Attach screen over ssh

```
ssh -t remote_host screen -r
```

Directly attach a remote screen session (saves a useless parent bash process)

2) Share a terminal screen with others

```
% screen -r someuser/
```

3) Triple monitoring in screen

```
tmpfile=$(mktemp) && echo -e 'startup_message off\nscreen -t top
htop\nsplit\nfocus\nscreen
-t nethogs nethogs  wlan0\nsplit\nfocus\nscreen -t iotop iotop' > $tmpfile &&
sudo screen -c $tmpfile
```

This command starts screen with ‘htop’, ‘nethogs’ and ‘iotop’ in split-screen. You have to have these three commands (of course) and specify the interface for nethogs – mine is wlan0, I could have acquired the interface from the default route extending the command but this way is simpler.

htop is a wonderful top replacement with many interactive commands and configuration options. nethogs is a program which tells which processes are using the most bandwidth. iotop tells which processes are using the most I/O.

The command creates a temporary “screenrc” file which it uses for doing the triple-monitoring. You can see several examples of screenrc files here: http://www.softpanorama.org/Utilities/Screen/screenrc_examples.shtml

4) Share a ‘screen’-session

```
screen -x
```

Ater person A starts his screen-session with `screen`, person B can attach to the srceen of person A with `screen -x`. Good to know, if you need or give support from/to others.

5) Start screen in detached mode

```
screen -d -m [<command>]
```

Start screen in detached mode, i.e., already running on background. The command is optional, but what is the purpose on start a blank screen process that way?
It’s useful when invoking from a script (I manage to run many wget downloads in parallel, for example).

6) Resume a detached screen session, resizing to fit the current terminal

```
screen -raAd.
```

By default, screen tries to restore its old window sizes when attaching to resizable terminals. This command is the command-line equivalent to typing ^A F to fit an open screen session to the window

7) use screen as a terminal emulator to connect to serial consoles

```
screen /dev/tty<device> 9600
```

Use GNU/screen as a terminal emulator for anything serial console related.

screen /dev/tty

eg.

screen /dev/ttyS0 9600

8) ssh and attach to a screen in one line.

```
ssh -t user@host screen -x <screen name>
```

If you know the benefits of screen, then this might come in handy for you. Instead of ssh'ing into a machine and then running a screen command, this can all be done on one line instead. Just have the person on the machine your ssh'ing into run something like
screen -S debug
Then you would run
ssh -t user@host screen -x debug
and be attached to the same screen session.

9) connect to all screen instances running

```
screen -ls | grep pts | gawk '{ split($1, x, "."); print x[1] }' | while read i; do gnome-terminal -e screen\ -dx\ $i; done
```

connects to all the screen instances running.

10) Quick enter into a single screen session

```
alias screenr='screen -r $(screen -ls | egrep -o -e '[0-9]+' | head -n 1)'
```

There you have ’em folks the top 10 screen commands. enjoy!