The power of the pipe

Chris Bidmead takes some pipeline code on a driving test: manoeuvre, signal, reverse, find a space...

The other morning I needed to come up with a string representing the name of the newest file in a particular directory. From the command line, the simplest way to achieve this is by using:

```
ls -t <directory> | head -1
```

In other words, list the files in <directory>, newest files first (ls -t) and pipe the result into the head utility, chopping off all but the first line.

There's a nice opportunity here to explore the power of the pipe at the Unix command line. The example above is oversimplified, because there's a chance, of course, that the most recent file – the one at the top of the ls -t list – isn't a file. It could be a directory.

An easy way to distinguish files from directories is by using the -l switch to create a 'long' listing. This gives you an expanded list that includes the attribute flags as well as the date-last-modified of each file and its size (see Fig 1).

Now if you just want to list the directories, you can do:

```
ls -l | grep ^d
```

In other words, pipe the full listing to the grep filter utility and only allow through the lines beginning with a letter d (the caret represents the beginning of the line).

To do the reverse – filter out the directories – it is possible to negate the grep with the -v switch (think inVert). By the way, the discussion here relates to the GNU version of these common Unix tools. On proprietary systems your mileage may vary.

Now we can expand our original short pipeline a bit:

```
ls -lt | grep -v ^d | head -1
```

This effectively says: 'Give us a long listing, in inverse date order, filter out the directories, and then filter out everything except the first line.'

However, it doesn't quite work in the way that you might expect, because the -l switch creates a summary first line, which gives a total of the number of files:

```
total 1198
```

This means we need to extend the pipeline a step further, increasing the head lines to two and then use tail (the inverse of head) to isolate the second line:

```
ls -lt /etc | grep -v ^d | ↙
head -2 | tail -1
```

giving:

```
-rw-r--r--   1 root    root ↙
     1650 Oct 15 08:15 issue
```

(*Key: ↙ code string continues*)

Another way of achieving the same thing would be to change our grep pattern and look for that -r which starts the flag array for normal files. This is complicated by the need to protect the hyphen with a backslash so that grep doesn't think it's a command switch, and then wrap the pattern in single or double quotes to insulate it from the shell.

```
ls -lt /etc | grep '\-r' | ↙
head -1
```

The new grep pattern slightly changes the rules. Links (where the flag array begins with an 'l') are now excluded, whereas a negated '^d' still allowed them. And we're also now insisting that the file should be readable. We probably don't want links and we do want a readable file, so let's stick with this.

Another way of achieving the same thing would be to change the grep pattern

But we're not done yet. We wanted just the filename. But because we're using the -l switch to ls we end up with a line full of other stuff such as flags, ownership and date. Is there an easy way to strip this away?

The heavy duty way of breaking up a line is with awk, but there's a lightweight utility we can use here called cut. Cut can operate in various ways, but in the present context we'll think of the line as a series of fields. As you can see, the filename is in the ninth field, so we might expect the cut command to be something such as:

```
cut -f 9
```

But, we're not quite there yet, because we have to tell cut what delimiter we're using to separate the fields. The default is a tab, but there are no tabs in the line. We'd like to use a regex such as /[ ]+/ (one or more spaces), but, alas, cut doesn't understand regexs. Cut lets us use only a single character to indicate the field boundaries, so in a line whose fields are broken by one or more spaces we have no way of counting to the ninth field.

We could isolate the first field very nicely with:

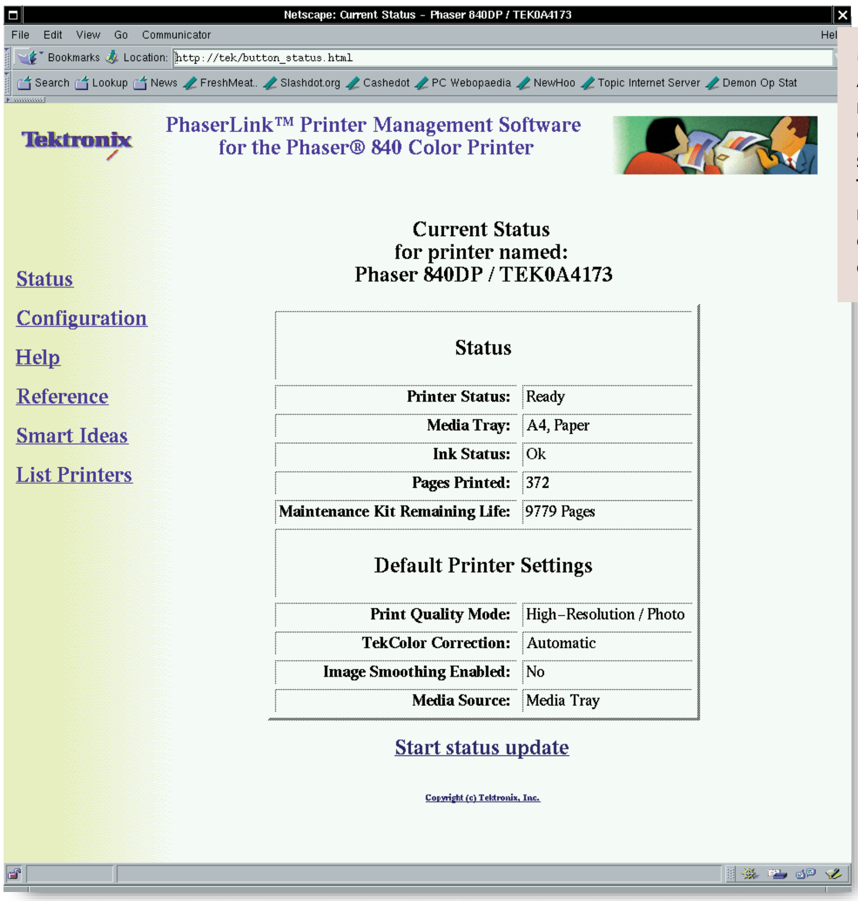
```
cut -f 1 -d " "
```

This is because we know there are no spaces in the flag field, so the first space must demark the end of the field.

Let's change the exercise and say I want to end up with a string that contains the flag field of the newest

(FIG 1)

```
drwxr-xr-x   3 root     root   1024  Jul 19 18:13 CORBA
-rw-r--r--   1 root     root   2241  Aug 11 20:53 DIR_COLORS
-rw-r--r--   1 root     root     24  Sep  8 09:33 HOSTNAME
-rw-r--r--   1 root     root     41  Aug  6 18:29 MACHINE.SID
-rw-r--r--   1 root     root   5441  May  9 19:14 Muttrc
drwxr-xr-x  15 root     root   1024  Aug 11 11:26 X11
-rw-r--r--   1 root     root     10  May 26 01:55 adjtime
-rw-r--r--   1 root     root    732  Apr 13  1999 aliases
-rw-r--r--   1 root     root  16384  Jul 19 18:21 aliases.db
-rw-r--r--   1 root     root    406  Apr 27 14:19 anacrontab
-rw-------   1 root     root      1  May 18 23:28 at.deny
-rw-r--r--   1 root     root    302  May 25 13:27 bashrc
-rw-r--r--   1 root     root    130  Oct 12 08:16 bc.conf
```



Netscape: Current Status – Phaser 840DP / TEK0A4173

File Edit View Go Communicator Help

Bookmarks Location: http://tek/button_status.html

Search Lookup News FreshMeat.. Slashdot.org Cashedot PC Webopaedia NewHoo Topic Internet Server Demon Op Stat

Tektronix

PhaserLink™ Printer Management Software
for the Phaser® 840 Color Printer

Status
Configuration
Help
Reference
Smart Ideas
List Printers

Current Status
for printer named:
Phaser 840DP / TEK0A4173

Status

Printer Status:	Ready
Media Tray:	A4, Paper
Ink Status:	Ok
Pages Printed:	372
Maintenance Kit Remaining Life:	9779 Pages

Default Printer Settings

Print Quality Mode:	High-Resolution / Photo
TekColor Correction:	Automatic
Image Smoothing Enabled:	No
Media Source:	Media Tray

Start status update

Copyright (c) Tektronix, Inc.

Most network printers can be administered remotely, but this typically requires special software running on the client machines. The great joy of a web server is that it's totally cross-platform. The web page in the screenshot is from the latest Tektronix Phaser 840, which is now capable of printing photo-quality colour on both sides of the paper

readable plain file in a directory. Here's the pipeline to do that:

```
ls -lt /etc | grep "\-r" | ✓
head -1 | cut -f 1 -d " "
```

QED. Er... but as you've noticed, I cheated by changing the goal. What we really want is the last field.

But, hey, the last field is the first field – it just depends on how you look at it. The utility for looking at a line backwards is called rev. If we pipe our output through rev just before the cut operation, we get:

```
eussi 51:80 51 tc0 0561    ✓
toor
toor 1
--r--r-wr-
```

So now we just apply the cut as before and pipe the output from that through another rev – to get the filename the right way round:

```
ls -lt /etc | grep "\-r" | ✓
head -1 | rev | cut -f 1 -d ✓
" " | rev
```

Of course there's one pesky case where this fails – if the filename contains a space, as under Unix it's perfectly entitled to do. I'm not too bothered, because pipelines such as this are typically quick fixes used in '80 per cent solutions'. In any case this has been an exercise in pipelining, not an attempt at rock solid program building.

But if you can think of an improvement that covers this case I'd love to hear from you. Stick to pipelining simple, standard utilities – Perl is cheating (isn't it always?).

Stick to pipelining simple, standard utilities – Perl is cheating (isn't it always?)

The best I can do for now improves on the above by taking us into the murky reaches of sed, the streaming text editor. But this relies on the fact that the filename doesn't contain a colon followed by a couple of digits. It's a more remote eventuality than an embedded space, but by no means a watertight assumption.

```
ls -lt | head -2 | tail -1 | ✓
sed -ne 's/^.*:[0-9][0-9][ ✓
]*//; p'
```

My FIFO-LIFO email queue

Spencer Lavery (species1@playaz.greatxscape.net) wrote to me this month with a question that keeps cropping up about 'any sites where I could either download Linux (preferably the latest version) or order a CD for free'.

He was mailing from a Windows system using Microsoft Outlook Express, and, like many beginners, had it set to do HTML by default. I mentioned the unnecessary crud this adds in my reply to Spencer, and also told him – as regular readers will already know – that The Linux Emporium (http://linuxemporium.co.uk) is a great place to get all kinds of Linux (and other) CDs at a range of prices, from zero pounds upwards. John Winters who runs the outfit has a laudable policy of giving away older versions of the GPL distributions.

Spencer wrote back (in plain ASCII, which was cool): 'I just want to say thank you again for your amazingly quick and helpful response. I was unsure if you would reply as it was an old issue I was reading, but I am pleasantly surprised'.

Other readers who've emailed me may also be wondering about that 'amazingly quick' response. I revealed my secret to Spencer:

Well, I have a queue of incoming email for the column that's about a month long. This in itself isn't impressive, and if I just keep answering all my mail a month late that isn't too impressive either. So I cheat and alternately answer mail at the beginning of the queue and at the end. That way I build half a reputation for responding quickly to email... :-)

Web servers everywhere

They say that in a few years' time our fridges and freezers will all have web servers in them, so we can quiz them about their contents. Back in the summer of 1997 I showed you the web server in my Tektronix Phaser solid inkjet printer. This lets you check the status and change parameters remotely, and also contains links to online documentation.

Network printers should be comfortable with any operating system. However, the Tektronix Phaser range has a particular empathy with Unix, because it contains its own BSD-like print queue and you can treat it like any remote Unix machine with a local printer attached to it.

This snippet from my /etc/printcap file – which was created using the Red Hat printtool utility – illustrates the point:

```
##PRINTTOOL3## REMOTE ↙
POSTSCRIPT 600x600 a4 {} ↙
PostScript Default 1
lp:\
     :sd=/var/spool/lpd/lp:\
     :mx#0:\
     :sh:\
     :rm=tek:\
     :rp=AUTO:\
     :if=/var/spool/lpd/↙
lp/filter:
```

The printer's internal queue, called AUTO, includes a magic filter than can distinguish between PostScript and PCL. This means that the input filter line (:if above) added by printtool is unnecessary in this context.

The BJ-10e isn't a PostScript printer. Well, yes it is – Ghostscript just made it one

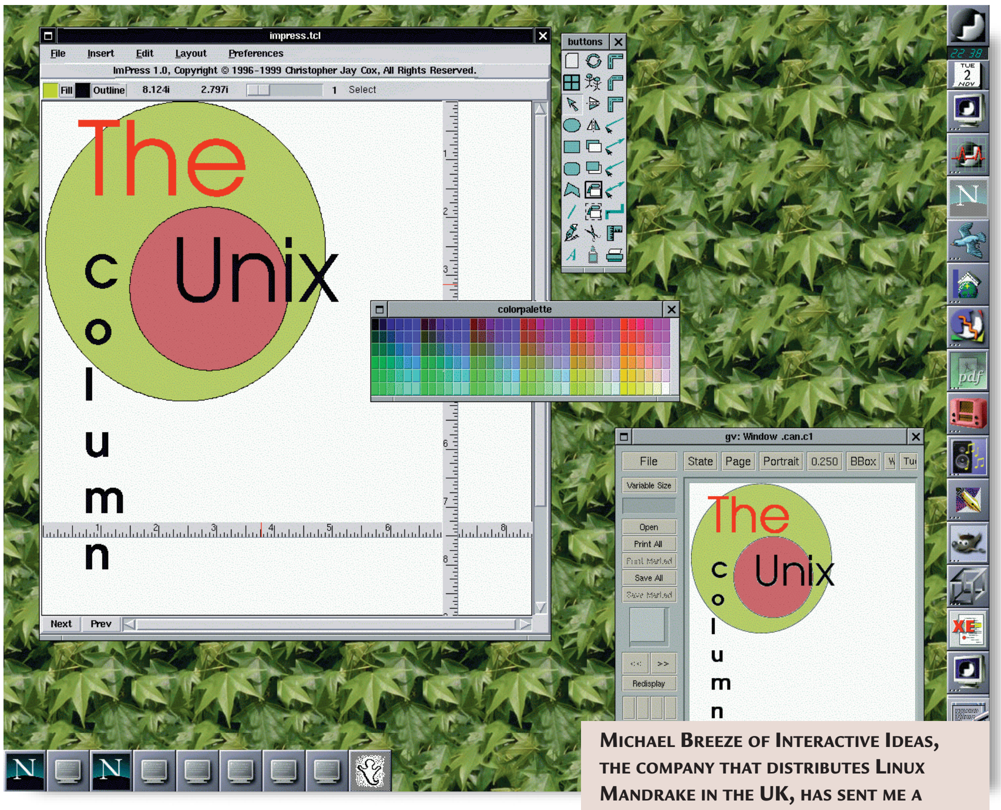
But of course not everyone has a network PostScript printer. It's when you're working with a low-cost printer attached to a Unix machine that this input filter comes into its own. For example, I used it to turn a modest old Canon BJ-10e bubblejet into something reasonably equivalent to a PostScript laser printer.

To do this you need Ghostscript, the free PostScript emulator that comes with most Linux (and many other free Unix) distributions. Full details about how to set this up are in the Printing HOW-TO and the Printing Usage HOW-TO (which you can find at, for example, www.metalab.unc.edu). But the bare bones of it go like this...

The output that your application directs to the printer gets stacked in a local queue, typically in a directory somewhere below /var/spool/lpd/. Once there, it is intercepted by the filter, which analyses it in order to guess what kind of printer it's meant for. The filter (strictly speaking a set of filter scripts and utilities that interact) will then rewrite the file as PostScript.

But the BJ-10e, which you could probably pick up in a charity shop these days for about £5, isn't a PostScript printer. Well, yes it is, because Ghostscript just made it one. Instead of sending the file directly to the printer, the input filter mechanism diverts it to Ghostscript, which in turn converts the PostScript page-by-page into a set of low-level graphics instructions the BJ-10e can handle.

This depends on Ghostscript knowing about the internal codes used by your printer. You can check the list of known devices – you'll find it bundled with your distribution in a file called devices.txt. A more recent version of Ghostscript – including drivers for more printers, but under a less liberal licence – can be found at: www.cs.wisc.edu/~ghost.



impress.tcl
File Insert Edit Layout Preferences
ImPress 1.0, Copyright © 1996-1999 Christopher Jay Cox, All Rights Reserved.
Fill Outline 8.124i 2.797i 1 Select
The
Unix
c o l u m n
Next Prev
buttons
colorpalette
gv: Window .can.c1
File State Page Portrait 0.250 BBox
Variable Size
Open
Print All
Print Marked
Save All
Save Marked
Redisplay

Michael Breeze of Interactive Ideas, the company that distributes Linux Mandrake in the UK, has sent me a stack of Linux utility CDs put together by a German outfit called Hemming. On one of them, Graphic, Sound and Multimedia for X, I found something called Impress, a vector-based layout program for creating presentations and PostScript documents. It took only a moment to knock up this logo. The bottom right-hand window is GhostView, a front end to Ghostscript, which is displaying the PostScript file created by Impress

PCW CONTACTS

Chris Bidmead welcomes your comments on the Unix column. Contact him via the PCW editorial office or email unix@pcw.co.uk

The Hemming CDs cost £4.99 from Interactive Ideas (0208) 805 1000 or you can download Impress directly from www.ntlug.org/~ccox/impress