asciiprint.ps
Ben Cranston <zben@umd2.umd.edu>

In article <17862@rpp386.cactus.org> Woodrow Baker <woody@rpp386.cactus.org> caterwauls:

> [printing vanilla ascii files on PS printers] is so common, that we need
> to find a simple shorty ... I propose we sort of run an informal contest
> here to find who can write the shortest file printer in PS. At a minimum
> it should understand cr/lf's, use courier as its font, expand tabs to blanks.

I actually wrote this several months ago as part of a project to print output from an evil IBM VM/CMS system on LaserWriters. The stupid code (see below) in the LW input routine that converts carriage returns to linefeeds makes it impossible to support overprinting on a line-by-line basis by using bare CR as a move to column one. Because of this the project decided to use a C filter on the Unix machine to do the job, and I deleted the machine readable of this in disgust and anger. I redeveloped it this afternoon as an entry to the informal contest.

It does FF, LF, CR, TAB, and BS, in that heirarchical order. If you can get the damn CR in, it will do CR to column one for overprinting. The BS support means ul-bs-letter or letter-bs-ul underlining will work too.

Charles B. (Ben) Cranston
Network Infrastructures Group
Computer Science Center
University of Maryland at College Park
of Ulm

Tabs:
```
1       22      333     4444    55555   666666  7777777 88888888        9999
```

Underlining type 1:

Underlining type 2:

There is a formfeed at the end of this line --->

The operating principle can be understood just as the source program can be parsed: as a heirarchy of control character handlers. All handlers have this same basic form. First the remaining string is searched for the control character, but the result of the search is immediately buried on the stack while the {remainder of the buffer|string before the control} is processed. After this, the boolean result of the search is used to control exit from the loop. If control remains in the loop, the control character was indeed found, so the remainder of the loop does the specific action for the control character and then resumes the loop.

This organization engenders certain consequences. For example, the model of handling TAB heirarchically over BS means that an attempt to backspace over the space created by a TAB actually causes a backspace one normal character width INTO the space created by the TAB. This case is sufficiently weird that I did not consider it a problem.

The hierarchy is:

1. FF (form feed) causes a break to the first character position on a new page. If FF is given at the very end of a file a blank page is produced.

2. LF (line feed) causes a break to the first character position of a new line of the page, except when that would be below a certain point, in which case a break to the top of the next page occurs instead (unmarked "if").

3. CR (carriage return) causes a break back to the first character position of the CURRENT line. The intent was to allow underlining and overprinting on a line by line basis by using a bare CR between overprint segments. However, the LaserWriter designers, in their infinite wisdom and bearing in mind everything they had ever been taught about the operating system standing between the user and the hardware*, saw fit to convert CRs to LFs when read by the serial chip. Strangely enough, this does not happen during feed from the Appletalk(tm) network. However, the Macintosh uses CRs instead of LFs between lines, so the code doesn’t work with the Mac SendPS programs either...

4. TAB (horizontal tab) causes a break to the next multiple of 8 characters from the left margin. The magic constant 248 is 0xF8 which throws away the bottom 7 bits. The tab spacing could probably be changed if one were to make the appropriate adjustment to this number.

5. BS (backspace) causes the print position to move one character width back to the left. Note interaction with TAB described above.

If none of these control characters is found, the rest of the string is typeset at the current position. This means that there is nothing special to break lines wider than 80 characters. They just go on into the right margin and fade away. This is probably the single greatest shortcoming of this approach, but it is not really obvious to me what the right thing to do in this case might be...

* This is sarcasm.

Now, just to prove how truly demented I am, I’m going to copy the code into the file here, just to show that it can process itself (tee hee!).

(dive, dive!)

```
%!-Adobe-1.0
%%Title: asciiprint.ps
```

```
%%Creator: Ben Cranston <zben@umd2.umd.edu>
%%CreationDate: Mon Jul 31 1989
%%Pages 0

/Courier findfont 10 scalefont setfont
/colwid (m) stringwidth pop def
36 756 moveto
256 string
{ %exec
  { %loop over file
    dup currentfile exch readstring exch
    { %loop FF
      (\f) search exch
      { %loop LF
        (\n) search exch
        { %loop CR
          (\r) search exch
          { %loop TAB
            (\t) search exch
            { %loop BS
              (\b) search exch
              show
              not { exit } if
              pop currentpoint exch colwid sub exch moveto
            } loop %BS
            not { exit } if
            pop currentpoint exch 36 sub colwid div .49 add cvi
            8 add 248 and colwid mul 36 add exch moveto
          } loop %TAB
          not { exit } if
          pop 36 currentpoint exch pop moveto
        } loop %CR
        not { exit } if
        pop 36 currentpoint exch pop
        11 sub dup 25 le { %if
          pop showpage 756
        } if
        moveto
      } loop %LF
      not { exit } if
      pop showpage 36 756 moveto
    } loop %FF
    not { exit } if
  } loop %over file
  pop showpage
} bind exec

(surface, surface!)
--
Sig     DS.L    ('ZBen')       ; Ben Cranston <zben@Trantor.UMD.EDU>
* Network Infrastructures Group, Computer Science Center
* University of Maryland at College Park
* of Ulm
```