An Acrobat Flate Compression PostScript Viewer Utility

Don Lancaster
Synergetics, Box 809, Thatcher, AZ 85552
copyright c2002 as GuruGram #08
http://www.tinaja.com
don@tinaja.com
(928) 428-4073

Most newer **Acrobat** PDF distribution files have their object streams individually compressed by using **Flate Compression**. Flate is an open source improved variant of LZW compression that is both more efficient and royalty free. Being able to read these obtuse objects in plaintext can be quite useful for program analysis, feature extraction, and new software design.

The Flate Compression operation is detailed in the **PDF Reference Manual** and the **PostScript Reference Manual** from **Adobe Systems**.

One obvious approach is to print your entire .PDF file to disk as **PostScript** and then redistill the new file with compression turned off. However, this can sometimes cause a reorganization or worse. One that might eliminate or change the very object you are attempting to view.

I've posted a simple **FLATVUE1.PSL** raw PS Flate Compression viewer utility to my **Guru's Lair** website as file **http://www.tinaja.com/psutils/flatvue1.psl**.

You normally would use this utility by bringing it up in Wordpad or another WP or editor, studying the tutorial comments, changing the requested data, and sending the modified file to Acrobat Distiller. Distiller in turn produces a log file that contains both the compressed and freetext forms of your target object.

For instance, a typical output might look like…

```
Flate compression...

  (H lo¸n´0ZE¿  0¸v  %q"!*^T'**¢¤@&˜"l ″rsrSP+ q ƒ,P~y˛ı  «D¯i  †–‘ v  $Z  !7“R
  l>
  ł ¢u§¥øyc a7CB–
  ¢• &
  @m  * z  9œT›œB¯ NB·2 ¢ u
  ¯A   R5VJk*T]#}K "˝Oœd‘KbB[P"  R˛~˚HG]Zo7 nL ·  ¨f+=4 9+ yr&YLsH T  Y¿‘¡P„…)
```

Reads as...

```
/DocumentSetup<</PageOrigin [13 13 ]/RulerOrigin [0 0 ]>> DP
/Layer<</Title (Layer 1)/Visible true /Preview true /Editable true /Printed true
/Dimmed false /Color [20224 32768 65535 ]/flatLayer false /Transparency
false >> BDC
0 0.4039 1 rg
0 0 0 RG
0 i 1 w 4 M 0 j 0 J []0 d
/RelativeColorimetric ri
/R2 gs 543.7773 557 m
80.4443 557 l
80.4443 601.4443 l
543.7773 601.4443 l
543.7773 557 l
h
B
0 0.8745 0.1961 rg
/R3 gs 368.2227 437 m
256 437 l
256 712.5557 l
368.2227 712.5557 l
368.2227 437 l
h
B
0.9216 0 0.0196 rg
/R4 gs 440.4443 497 m
190.4443 497 l
190.4443 634.7778 l
440.4443 634.7778 l
440.4443 497 l
h
B
EMC
```

Your usual starting point is to copy a PDF file to a location that your Acrobat Distiller can reach. Such as...

```
/sourcefilename
(C:\\WINDOWS\\Desktop\\gurugrams\\alphatrans\\threerects.pdf)
def
```

A read file object called **/workfile** can then be created. Note that **double** reverse slashes are needed inside a PostScript string every time you mean a single slash!

You then should pull your PDF file up into Wordpad or another editor and shop around for three crucial values:

> **/fileposloc** is the start of the object holding your target Flate code. Which you read from the cross reference at the end of the .PDF file. Object six start will be on the sixth line down etc...
>
> **/fileheaderoffset** is the byte count at which the Flate compression actually starts. Which you read by counting or guessing at the number of header characters. You want the **actual** file start **without** any preceding linefeeds or carriage returns.
>
> **/flatelengthstr** is the actual character count in the compressed Flate file. Which you read from the header of your target file. It may be an actual number or may refer to another object such as **7 0 R obj** If an object, go to object 7 or wherever and read the value.

Once you find these values, you reposition your file start using...

```
workfile fileposloc fileheaderoffset add setfileposition
```

You then read the actual Flate compressed data by using...

```
workfile flatelengthstr readstring pop
```

The sneaky line that does all the nasty decoding work is...

```
/FlateDecode filter reportstring readstring
```

Note that the logfile opening parenthesis **must** be on the **same** line as the first compressed Flate character! Error messages will result unless you have the **exact** Flate start and the **exact** length. Once properly decoded, you then pretty print both the compressed file and its expanded result to the Distiller log file.

Note that a "**WARNING: Empty Job. No PDF file produced**" message is normal and expected since your intended results appear in the log file. Your log file results can then be textfile viewed or exported by the usual cut-and-paste.

This example only works for Flate files and decompressed outputs less than 65K long. As set by the length limit of a **PostScript** string. More general code can be written to handle any length by repeated direct writing to disk files.

Consulting services available per **http://www.tinaja.com/info01.asp**.