Simple Text Setting
Calculations for PostScript
Language Drivers

Adobe Developer Support

Technical Note #5041

31 March 1992

Adobe Systems Incorporated
Adobe Developer Technologies
345 Park Avenue
San Jose, CA 95110
http://partners.adobe.com/
Contents

Simple Text Setting Calculations for PostScript Language Drivers 5
1 Introduction 5
2 Line Breaking 5
3 Line Layout Options 6
   Ragged Right Text 6
   Justified Text 7
   Centered Text 7
   Flush-Right Text 7

Appendix: Changes Since Earlier Versions 9

Index 11
1 Introduction

Setting text with the PostScript™ language involves two basic operations: determining where to break a line and “distributing” the text appropriately once the line breaks have been established.

2 Line Breaking

Line breaking involves determining the point at which no more text will fit on the current line. A text-processing system should maintain this information on the host system for two reasons: 1) a WYSIWYG application needs to have this information to properly show line breaks on the screen and 2) even if the application is not WYSIWYG, because the PostScript language is interpreted, any calculations that can be done on the host, should be, to enhance the performance of the print job. Line breaking is based on the widths of the individual characters.

When using PostScript fonts, these decisions are made based on font metrics. These metrics provide the width of each character, which can be used to keep track of the overall string width of a line of text set in that font.

Font metrics are distributed by Adobe Systems in the form of Adobe™ Font Metric (AFM) files. These files contain the width of each character in the font in units of 1/1000 of an “em.” The numbers are independent of point size—they actually represent an idealized 1-point size. See the Adobe Font Metrics File Format Specification for more information.

Font metrics can be maintained in other forms on any given host system. For example, on the Macintosh® the information is available in the FOND resource for each font. The information is typically parsed from the AFM files and written into the more host-specific format.
The following is a calculation that will yield the correct string width of a piece of text, based on the width of the individual characters in the line (as found in the AFM files)

\[
\text{String Width} = \frac{\sum (\text{AFM widths all characters} \times \text{Point Size})}{1000}
\]

The string width can be maintained as the sum of the (integer) AFM width of the characters. When the precise width is needed at the PostScript language level, the string width value can be multiplied by the point size of the text and divided by 1000.

*Note* Floating-point calculations can be eliminated completely if the final division by 1000 is handled by relocating the decimal point in the resulting number.

The width computed by this method is as accurate as computing (in the PostScript interpreter) the *stringwidth* of 1000-point characters, and, in fact, the widths stored in the PostScript language character descriptions themselves are exactly the same numbers as appear in the AFM files. Remember that the character descriptions are executed in a “character” coordinate space scaled by the *FontMatrix* entry in the font dictionary (the scale factor is 1000 in all Adobe fonts).

### 3 Line Layout Options

There are several kinds of text setting that can be easily implemented once the width of the line of text is known.

#### 3.1 Ragged Right Text

Once the line of text has been constructed and line breaks are determined, it is only necessary at the PostScript language level to set the current font, the current position, and use the *show* operator to image the text. This will provide ragged right text

```latex
LeftMargin Y moveto
(Line of text) show
```
3.2 Justified Text

If you want the text justified, it is necessary to determine how much the line must be adjusted and to use the `widthshow` operator to modify the word spacing accordingly. The amount to pass to `widthshow` is computed as follows:

\[
\text{SpaceMod} = \frac{(\text{LineWidth} - \text{StringWidth})}{\text{Spaces}}
\]

In this equation, LineWidth is the desired width of the column (the fully justified width of the line of text). StringWidth is the width of the line of text as computed above, based on the AFM widths of the characters. Spaces is a count of the total number of space characters in the line of text.

The value SpaceMod is used with the byte code for the space character (32) in the call to `widthshow`:

```
LeftMargin Y moveto
SpaceMod 0 32 (Line of text here) widthshow
```

This will modify all occurrences of the space character by the amount given by SpaceMod. This accomplishes word spacing to achieve full justification.

3.3 Centered Text

Text can be centered even more easily than justifying. The StringWidth of the text can be simply divided in half, and the current point established from the center point of the column of text. The X value passed to `moveto` in the PostScript language is:

\[
X = \text{CenterMargin} - \frac{\text{StringWidth}}{2}
\]

In the PostScript language, the text is set simply:

```
X Y moveto (Line of text) show
```

3.4 Flush-Right Text

Flush-right text can be obtained in much the same way as centered text:

\[
X = \text{RightMargin} - \text{StringWidth}
\]

At the PostScript language level, the text is set simply:

```
X Y moveto (Line of text) show
```
Appendix: Changes Since Earlier Versions

Changes since the May 4, 1991 version

• Document was reformatted in the new document layout and minor editorial changes were made.

Changes since the January 16, 1989 version

• A few layout styles were changed.

• The cover addresses and phone numbers were updated.
Index

C
centered text 7

F
flush-right text 7
font 5

J
justified text 7

L
line breaking 5
LineWidth 7

P
PostScript language drivers
   text setting calculations 5–7
      floating-point calculations 6
   font metrics 5
   line layout options 6–7

R
ragged right text 6

S
SpaceMod 7
StringWidth 7