

PATHFINDERS SOFTWARE

For Email Gateway version 3.6

PDF Template Programming Guide

PATHFINDERS SOFTWARE

PDF Template Programming Guide

Email Gateway version 3.6
© 2002 Pathfinders Software
300 Murchison Drive, Suite 307
Phone 650 692-9220 • Fax 650 692-9250
Updated June 14, 2002

Overview

Sending PDF documents through the email gateway is easy.

This manual will teach you how to create a text template that will be converted for you into a PDF file and attached to an outgoing email message “on the fly”.

Since the Adobe PDF format is a perfect way to send documents to customers (invoices, order confirmations, statements, etc.), many programmers have asked how to create them from inside the PICK environment. The challenge has always been that PICK is text only, it cannot handle binary file types such as PDF.

The solution is to create a text template in Pick and have the underlying AIX or Linux environment render it into a binary PDF document behind the scenes before attaching it to the email message.

System Requirements

You must have Email Gateway version 3.6 for AIX or Linux installed. Email gateway programs are installed on both sides (PICK and AIX/Linux) .

The only basic requirements for AIX or Linux is to have a working C compiler. During AIX setup, the Perl scripting language and the ghostscript engine will be installed. These come built into Linux already upon a full install.

Special Directory For Images

To have images inserted into PDF documents “on the fly”, we need to keep these image files in AIX or Linux. They also, must be converted to EPS format. Included in this Email Gateway is a gif to eps conversion tool called **gif2eps**.

Included programs:

- Ghostscript & Perl environments with MIME encoding module
- Gif2eps & ps2pdf conversion tools
- Enscript program (text to ps tool)
- Pick TCL commands: SENDREC and TEXT2PDF

Invoking The PDF Converter

To cause an attachment to be run through the PDF converter invoke SENDREC with the E option. E is for “encode to PDF”. Option A means “prompt for attachment name”.

```
SENDREC EMAIL.FILE MSG005 (AE
```

PDF Templates

Text based PDF templates are created in PICK, and converted into binary PDF files. Each template can mix regular displayed text and processing commands. These commands include font changes, color changes, background shading and image insertion. You may even include raw Postscript code if you wish. See Appendix A.

See Appendix A for the complete list of supported PDF commands.

Here is a sample:

```
^epsf[){//tmp/logo.eps}
^font{Helvetica-Bold12}
Customer: Harrison Incorporated
^color{.9 0 0}
Order ID: 44321
^color{0 0 0}

^font{default}
Product | Description | Price
-----|-----|-----
22A-7 | Left handed smoke detector | 7.95
-----|-----|-----
sub total |
tax | 0.25
total | 8.20

^bggray{.9}Account is 60 days past due^bggray{1}
```

Notice the following steps in the above template file:

- 1) Insert an image called logo.eps
- 2) Change font to Helvetica Bold 12 point
- 3) Display “Customer: Test Incorporated
- 4) Change color to red (90% red, zero green, zero blue)
- 5) Display text: “Order ID 44321”
- 6) Change color back to black and default font type and size

Drawing Lines On The Page

When creating documents like orders and invoices, drawing lines on the page is a common task. This requires issuing raw postscript commands to create vertical and horizontal lines. This is done with the `^ps{ }` command.

When To Draw

Since the page text must show up in their proper boxes, it is recommended that all lines be drawn at the beginning of the document. This is due to the way it keeps track of where it is on the page as it creates it.

Fixed text that is part of the template and never changes should also be drawn at the beginning also.

Example of Drawing Two Vertical Lines Downward

```
^ps{
  /inch {72 mul} def           % define term inch (72 pixels = 1 inch)
  .8 inch 7.65 inch moveto    % jump pen over to new position
  0 inch -1.0 inch rlineto    % place pen down and draw a line
  stroke                       % display vertical line we just made
  5.5 inch 7.65 inch moveto
  0 inch -1.5 inch rlineto
  stroke                       % display vertical line we just made
}
```

Notice that this raw postscript code allows optional comments that follow a percent sign “%”.

The command **rlineto** actually draws the line. It takes 2 parameters, a horizontal length and vertical length. If the first number is negative it draws to the left, if positive then it draws to the right. If the second number is negative it draws down the page, if positive it draws upward on the page. Diagonal lines may also be drawn if both numbers are non-zero.

The command **stroke** makes the drawn line “appear” on the page. It is required.

Example of Drawing A Horizontal Line

```
^ps{
  /inch {72 mul} def
  6.25 inch 0 inch rlineto
  stroke
}
```

Notice that this line will be drawn from the current position on the page 6.25 inches to the right. Also, the `/inch` line is not required if this came earlier in the document.

See Appendix C for more postscript commands.

Appendix A: PDF Processing Commands

Below is a list of available commands that can be used inside a text template file. Each command begins with a carat symbol “^”. Therefore this character cannot appear in a template file.

^bggray{value} Change the text background color shading.

value	a number from 0 to 1.0 where 1.0 is white and 0 (zero) is black. Default is 1.0 (white)
-------	---

Example:

```
^bggray{.5}
```

^color{red green blue} Change the text color.

red green blue	decimal from 0 to 1 indicating how much color component to include
----------------------	--

Example of a red font color change and back to black:

```
^color{.9 0 0}This is a red line.^color{0 0 0}
```

^escape{code} Change the escape character. Useful when the carat is needed as a regular character to be displayed.

code	a decimal number from 0 - 255. Default is 94, the carat “^”
------	---

Example changing the escape character to the @ sign and back to carat:

```
^escape{64}This is a carat ^ symbol.@escape{94}
```

^epsf[options]{filename} Insert EPS file (image) into the document.

options	c = centered r = right justified sn = scale by a factor of n units hn = scale height of n units sxn = scale width (x direction) a factor of n units syn = scale height (y direction) a factor of n units nx = do not update current x coordinate ny = do not update current y coordinate xna = move left corner to position n (x direction) yna = move left corner to position n (y direction) Units default to lines (characters) but can be specified as: i = inches c = centimeters p = postscript points default is “as is”.
Filename	Full path to eps image file

Example: Inserting a picture of a mailbox and scaling it to 2 inches in size followed by a line drawing unscaled.

```
^epsf[s2i]{/tmp/mailbox.eps}  
^epsf[]{/tmp/fancyline.eps}
```

Note: your system may include the program gif2eps for converting images into the proper eps format.

^font{fontname} Change the current font face and size.

Fontname	Font from Appendix B with numerical point size specified or the word default. The default is Courier10 for portrait and Courier7 for landscape.
----------	---

Example: Changing font and back to default

```
^font{Courier-Bold12} This is my bold writing!  
^font{default}
```

`^ps{code}`

Insert raw PostScript code into the document.

code	Raw postscript language codes. See Appendix C for common uses.
------	--

`^shade(gray)`

Change text background shading of current line.

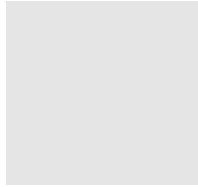
gray	Decimal value from 0 – 1.0 where 1.0 is the default (white).
------	--

Example: Changing the sentence to to a light gray shade.

```
^shade{.8} Payment is overdue! ^shade{1}
```

Appendix B: Available Fonts

These are all of the names of the fonts that can be use with this system:



Note: Do not forget to specify a point size when changing fonts

AvantGarde-Demi
AvantGarde-DemiOblique
AvantGarde-Book
AvantGarde-BookOblique
Bookman-Demi
Bookman-DemiItalic
Bookman-Light
Bookman-LightItalic
Courier-Bold
Courier-BoldOblique
Courier-Oblique
Helvetica
Helvetica-Bold
Helvetica-BoldOblique
Helvetica-Condensed
Helvetica-Condensed-Bold
Helvetica-Condensed-BoldObl
Helvetica-Condensed-Oblique
Helvetica-Narrow
Helvetica-Narrow-Bold
Helvetica-Narrow-BoldOblique
Helvetica-Narrow-Oblique
Helvetica-Oblique
Matrix
NewCenturySchlbk-Bold
NewCenturySchlbk-BoldItalic
NewCenturySchlbk-Italic
NewCenturySchlbk-Roman
Palatino-Bold
Palatino-BoldItalic
Palatino-Italic
Palatino-Roman
Symbol
Times-Bold
Times-BoldItalic
Times-Italic
Times-Roman
ZapfChancery-MediumItalic
ZapfDingbats

Fonts are specified like this:

```
^font{Helvetica-Bold10}INVOICE TOTAL
```

Appendix C: Inserting Raw Postscript

One common thing to do is to draw vertical and horizontal lines on the page. This must be done in raw postscript code.

The basic concept here is that you move an invisible pen on the page to x and y coordinates, then draw the line and then move it again.

Drawing a horizontal line from current position:

```
^ps{
  /inch {72 mul} def
  6.25 inch 0 inch rlineto
  stroke
}
```

The above example shows that we created a 6.25 inch line drawn horizontally from the current x,y coordinate on the page. This is useful when you wish to mix text and drawn lines across the page without worrying about what is the current “pen” coordinate.

Drawing of vertical lines can be done at the end of the page, when we will go back by moving the pen position first.

Moving The Pen And Drawing A Vertical Line Downward:

```
^ps{/inch {72 mul} def
  1 inch 8.5 inch moveto
  0 inch -1.5 inch rlineto
  stroke
}
```

The **moveto** command will go to a specific point on the page. The lower left corner is 0,0. So, in the above example, we are 1 inch from the left margin and 8.5 inches up from the bottom of the page. We then draw a 1.5 inch line downward by specifying -1.5 . If we gave it a positive 1.5 then it would draw upward.

Form Feed – Ending The Current Page:

```
^ps{ showpage }
```

Use this command to output the current page. It forces a form feed. This way, you can control when to end the current page and start the next page.

Setting Line Thickness:

```
^ps{2 setlinewidth}
```

This sets the current line drawing setting to 2 pixels wide. It might be somewhat device dependent, but overall it will work as expected.

Drawing A Black Box With White Writing:

```
^ps{
  /inch {72 mul} def
  newpath                                % start box
  0 inch 6 inch moveto                    % move pen to starting point
  7.5 inch 0 inch rlineto                 % draw horizontal top line
  0 inch -0.3 inch rlineto                % draw vertical right line
  -7.5 inch 0 inch rlineto               % draw horizontal bottom line
  closepath                               % draw 4th line automatically
  fill                                    % make solid box black
  1 1 1 setrgbcolor                       % change to white text color
  0.1 inch 5.88 inch moveto                % move pen back to start
  (QUANTITY) show
  0.1 inch 5.75 inch moveto                % move pen back under QUANTITY
  (ORDERED) show
  0 0 0 setrgbcolor                       % back to black text color
}
```

This shows drawing 4 lines and filling it with the current color (black) then changing the pen color to white. Finally, we move the pen back to just inside the upper left corner and draw QUANTITY then move the pen below that text and draw ORDERED underneath it.

Drawing A Simple Box With Black Writing:

```
^ps{
  newpath                                % start box
  2.1 inch .55 inch moveto                 % move pen to starting point
  2.3 inch 0 inch rlineto                  % draw horizontal top line
  0 inch -0.5 inch rlineto                 % draw vertical right line
  -2.3 inch 0 inch rlineto                 % draw horizontal bottom line
  closepath                               % draw 4th line automatically
  stroke                                  % show the box
  2.13 inch .44 inch moveto                 % move pen to starts
  (MERCHANDISE AMOUNT) show                % show text MERCHANDISE AMOUNT
}
```

This shows drawing 4 lines to make the box and then moving the pen inside the box to write the text MERCHANDISE AMOUNT. The box is black and the text is black.

Drawing A Pay Box With Triangle:

```
^ps{
  /inch {72 mul} def           % define variable "inch"
  newpath                     % ready to draw rectangle
  2.1 inch .55 inch moveto     % move pen to starting point
  5.35 inch 0 inch rlineto     % draw horizontal top line
  0 inch -0.5 inch rlineto     % draw vertical right line
  -5.35 inch 0 inch rlineto    % draw horizontal bottom line
  closepath                   % draw 4th line of rectangle
  stroke                       % show rectangle
  2.1 inch .4 inch moveto      % move pen inside rectangle
  4.0 inch 0 inch rlineto      % draw horizontal line
  stroke                       % show line
  6.1 inch .55 inch moveto     % move pen to PAY AMOUNT
  0 inch -0.5 inch rlineto     % draw vertical line
  stroke
}^ps{
  newpath                     % ready to draw triangle
  6.12 inch .4 inch moveto     % move pen to under text
  1.3 inch 0 inch rlineto      % top line of triangle
  -.65 inch -.1 inch rlineto   % right line of triangle
  closepath                   % close 3rd side of triangle
  fill                         % fill triangle with black
}^font{Helvetica-Bold7}^ps{
  6.11 inch .44 inch moveto    % move pen above triangle
  (PLEASE PAY THIS AMOUNT) show
}
```

The above example is the most complete one. First we draw a wide rectangle near the bottom of the page, then draw two lines. Then display the text PLEASE PAY THIS AMOUNT. Finally draw a triangle. Notice how we avoid extraneous Carriage returns and linefeeds by the use of }^ps{ sequences.

Appendix D: Pick TCL Commands

There are two TCL commands you may use for creating PDF documents. The first command, SENDREC will be used for sending a PDF attachment through email system. The second one, TEXT2PDF is used to create a PDF document in the underlying Linux or AIX environment for any use you may desire. One such use is the mapping of a drive letter from Windows and accessing a Linux folder specifically for holding PDF files generated by this system.

SENDREC *Filename Itemname* (AE

Use this command to use to send an email with a PDF attachment

Using the (AE options together will cause it to prompt you for the name of the PDF template. Then it will convert it and send it out as an attachment.

TEXT2PDF *Filename Itemname Unixpath*

Use this command to create pdf files for testing.

```
TEXT2PDF INVOICES INV001 /tmp/invoice1.pdf
```