

Print Wizard

Version 2.7w (as in AnzioWin 15.0s) and later

Operating Instructions



10240 SW Nimbus, Suite L9, Portland, OR 97223 USA
e-mail: rsi@anzio.com

Phone: 503/624-0360 Fax: 503-624-0760
<http://www.anzio.com>

Table of Contents

1	What Print Wizard Is.....	5
2	Print Wizard's Text Processing.....	7
2.1	Output Options.....	7
2.2	Beginning Assumptions.....	8
2.3	Print Level Switching.....	8
2.4	Basic Text Handling.....	8
2.5	Auto-fit Logic.....	9
2.6	Mini-page Logic.....	10
2.7	Auto-rotation Logic.....	10
2.8	Character Set Issues.....	10
2.9	Automatic Font Switching and Diacritics.....	11
2.10	AutoTab and Tab Handling.....	11
2.10.1	Classic tab handling.....	11
2.10.2	Autotab tab handling.....	11
3	Programming for Print Wizard.....	13
3.1	Column Alignment with FS Characters.....	13
3.2	Print Wizard's Markup Language.....	13
3.3	Using Initialization Files.....	14
4	Print Wizard's Photo Printing.....	16
5	Metafiles.....	17
5.1	Capturing an SPL File.....	17
5.1.1	Locating the SPL Directory.....	17
5.2	The Capture Process.....	18
6	The Scanner as Input File.....	19
7	Using Print Wizard.....	20
7.1	Inside AnzioWin.....	20
7.2	The PRINTWIZ.EXE Program.....	21
7.2.1	Starting Print Wizard.....	21
7.2.2	Parameters.....	22
7.2.3	Examples: Active mode.....	24
7.2.4	Examples: Despooler mode.....	25
7.2.5	Examples: JetDirect printing.....	26
7.2.6	Examples: Default print processor.....	26
7.2.7	Debugging Mode.....	27
7.2.8	The Unix <i>netprint</i> Program.....	27
7.3	The PWLPD.EXE Program.....	28
7.4	The Print Wizard DLL.....	28
7.4.1	Using the DLL from Visual Basic or VBA.....	28
7.4.2	Using the DLL from AcuCobol.....	29
7.5	Web Print Object (WēPO).....	30
7.5.1	The Web Page.....	30
7.5.2	The Print Job File.....	32
7.5.3	Samples.....	32
8	Faxing and Emailing with Print Wizard.....	33
8.1	Faxing and Emailing Components.....	33

8.1.1	The MAPISEND Program.....	33
8.1.2	WinFax Pro Support.....	34
8.1.3	Acrobat Support.....	34
8.2	Using the Faxing and Emailing Components.....	34
8.2.1	Internal “!” Commands.....	35
8.2.2	Command-line Faxing.....	35
9	PDF Generation.....	36
9.1	The Built-in PDF Generator.....	36
9.2	Things You Can Do with PDF.....	36
10	PCL Translation.....	37
10.1	The Built-in PCL Translation.....	37
10.2	Where Do PCL Files Come From?.....	37
10.3	How You Can Use PCL Translation.....	37
10.4	When TranslatePCL is OFF.....	38
11	Form Overlays.....	39
11.1	Bitmap Overlays.....	39
11.2	Scanned Overlays.....	40
11.3	EMF and WMF Overlays.....	40
11.4	SPL Overlays.....	40
11.5	PCL Overlays.....	40
11.6	PWML Overlays.....	40
11.7	Comparison of Methods.....	41
12	The Print Preview Feature.....	42
13	Print Wizard’s Markup Language.....	43
13.1	Recognizing a Marked-up Document.....	43
13.2	Tag Basics.....	43
13.3	End Tags.....	43
13.4	The Bare Minimum.....	44
13.5	Units of Measure.....	44
13.6	Positioning.....	45
13.7	Global Filtering with Regular Expressions.....	45
13.8	Default Margins and Alignment.....	47
13.9	Setting Page Dimensions and Margins.....	47
13.9.1	Paper Names, Bin Names, etc.....	48
13.9.2	Changing Pagesize in the Middle of a Job.....	48
13.9.3	Usage Patterns in Pagesize Tags.....	48
13.10	Multi-column Printing.....	49
13.11	Colors.....	49
13.12	Fonts.....	50
13.13	Automatic Font Selection.....	50
13.14	Font Size, Pitch, and Linespacing.....	51
13.15	Character Entities.....	52
13.16	Variable Width Spaces.....	53
13.17	Line-drawing Characters.....	54
13.18	Rectangles.....	54
13.19	Lines.....	54
13.20	Graphics.....	54
13.21	Form Overlays.....	55
13.21.1	Storing and Using PCL Overlays in PCL 5 Printers.....	55
13.22	Barcodes.....	55
13.23	PWML Reference.....	56

14	The Samples Directory.....	64
14.1	Files to be Printed.....	64
14.2	Other Files.....	65
15	Application Notes.....	66
15.1	Print Wizard and filePro.....	66
15.1.1	Installation and Use – Windows.....	66
15.1.2	Installation and Use – Unix/Linux.....	66
15.1.3	Usage Notes.....	67
15.2	Print Wizard and III Millennium and Innopac.....	68
15.2.1	Connecting to Innopac via Attached Print.....	68
15.2.2	Connecting to Millennium or Innopac via Workstation JetDirect.....	68
15.2.3	Character Sets.....	68
15.2.4	What Print Wizard Will Do.....	69
16	Using MAPISEND.....	70
16.1	What MAPI Is.....	70
16.2	Configuring MAPI.....	70
16.2.1	Choosing Your MAPI Program.....	70
16.2.2	Basic MAPI.....	71
16.2.3	Extended MAPI.....	71
16.3	The MapiSend Program.....	71
16.4	How it Works – Notes on Various Configurations.....	72
16.4.1	Microsoft Outlook e-mailing.....	72
16.4.2	Outlook Express e-mailing.....	72
16.4.3	Netscape Messenger e-mailing.....	73
16.4.4	Microsoft Outlook faxing.....	73
16.4.5	Outlook Express and Netscape Messenger faxing.....	73
16.4.6	Others.....	73
16.5	Examples.....	73
16.6	Errors.....	74

1 What Print Wizard Is

Print Wizard is a print processing engine that runs on Windows (95 and later). It takes as input streams of plain text, and outputs to any printer (or fax driver) that Windows supports. This print engine has been implemented as a program, (PRINTWIZ.EXE), as a callable module (PWDLL.DLL), as an ActiveX object (PRINTWIZ.OCX), and as an integral part of AnzioWin, a telnet (terminal emulation) program also from Rasmussen Software. In addition, the Print Wizard product includes the MapiSend program (MAPISEND.EXE) which can programmatically send faxes using Print Wizard. The Print Wizard product also includes a program (PWLDP.EXE) that acts as an LPD/LPR server, allowing Unix and other host systems to print on the PC's printer.

-
- Note: This document covers several *programs*, which may be included in several *products*. The description here does not imply that a particular program is part of a particular product.
-

Print Wizard's first task is, given a wide variety of print job formats, to print each job in a reasonable way, without user intervention. Beyond that, it lets the designer of the print job specify precisely how the job should print.

Print Wizard is targeted primarily at "line printer" or "greenbar" style print jobs; that is, one line in the data file should be printed as one line on the paper. It is not aimed at producing filled, justified, or centered text. Yet.

Print Wizard can process text that comes to it in a variety of ways:

- From a host system, via passthrough print
- From a PC disk file
- Generated by various programming languages in Windows
- Fetched from an FTP or HTTP (web) server
- Despoiled from a PC directory
- From Windows Explorer, as the designated print agent for a particular file type
- Via Windows drag-and-drop
- From a host system, via the LPD/LPR protocol
- From a host system, using the "JetDirect" protocol
- From a web page

Print Wizard can take as input:

- Plain text files
- Text files with formfeeds, tabs, backspace-bolding, and backspace-underlining
- Text files with embedded control sequences
- PostScript files
- Text files with markup tags
- Bitmap files
- PCL files
- EMF, WMF, or SPL files
- Tab-separated values files

Print Wizard can output:

- Plain text reports that always fit the page
- With specified orientation, paper size, bin, and duplexing
- Bolding, italics, and underlining
- Font changes
- Multilingual characters (Unicode), including the Euro
- Rectangles, gray bars, and color bars
- Lines

- Color text, including white-on-black for security
- Bitmaps (inline or precisely placed)
- Bitmap form overlays
- Windows metafile form overlays
- PCL form overlays
- Barcodes
- With precise placement of print elements
- Multi-column labels
- One-at-a-time labels on continuous forms
- Variable-length output, such as with receipt printers
- Full-page photos
- Through fax software to a remote fax machine
- Through email, as plain text
- Through Acrobat PDF Writer, to a PDF file
- Through Acrobat PDF Writer, to email
- Natively to a PDF file

Print Wizard solves a wide variety of printing problems:

- Data processing reports that won't print right on a laser printer
- Line wrap and page break problems
- Multilingual printing requirements
- The Euro symbol
- Driving fancier printout from legacy programming environments
- Printing from scripting languages
- Printing from DOS programs in a Windows environment
- Printing to "Windows only" printers from legacy environments
- Adding form overlays to existing print environments
- Programmatic faxing
- Programmatic emailing
- Printing "print-to-file" files from PC disk
- Printing to serial, parallel, and network-connected printers
- Printing HTML pages that browsers don't print well
- It takes too many steps to print a photo
- The need to preview a print job and adjust output placement before printing
- The need to print a PCL file on a non-PCL printer

Print Wizard is designed to work with any printer that Windows supports. However, its input is completely device independent. With Print Wizard you can program your print output without locking in to a particular type of printer.

Where most Windows programs let you *interactively* design and print your output, Print Wizard lets you *program* your output.

2 Print Wizard's Text Processing

Print Wizard takes as its input a "text stream". This may be a file, or it may be arriving via some sort of streaming protocol. However, its operation is basically the same.

Print Wizard's logic is designed to handle these general classes of input:

- 1) Raw Text - text with embedded control sequences, or PostScript data.
- 2) PCL files – files with PCL escape codes designed for a Laserjet or compatible printer.
- 3) Plain Text - text with linefeeds (LF), and optionally carriage returns (CRs), tabs and formfeeds (FFs).
- 4) Marked-up Text - text with Print Wizard's Markup Language (PWML), which is based on HTML, or HTML itself.
- 5) Bitmap files - to be printed at full page
- 6) A scanned image
- 7) Meta files - Windows WMF, EMF, and SPL files.

Print Wizard will first determine what class of input it has. If it is raw text, it will be sent to the Windows spooler at a very low level, bypassing the Windows printer driver.

If Print Wizard is configured to translate PCL, and it sees escape codes in the data, it will translate the PCL internally and print the data on any Windows supported printer.

For plain text, Print Wizard will analyze the report to determine the number of characters per line and, if possible, the number of lines per page. Based on this information, and the printable size of the page, it will set the font size, spacing, and margins, in each direction, and then print out the report using the Windows printer interface. In some cases, it will decide what length the page needs to be, and will create a custom page length.

If Print Wizard detects its markup language, it will proceed as with plain text, but the various markup tags can control virtually all aspects of its behavior. The markup language can also direct Print Wizard to print certain other non-text items, including barcodes, bitmaps, and rectangles.

If the filename extension indicates that the file is a bitmap, Print Wizard will print it as large as possible.

A scanned image can be printed as a full page bitmap.

If the filename extension indicates that the file is a metafile, Print Wizard will print it as one or more full pages.

The following sections describe these features in detail.

2.1 **Output Options**

Print Wizard can "print" its output on:

- 1) The Windows default printer
- 2) Any other printer visible to Windows
- 3) One or more EMF files
- 4) A PDF file, through its native PDF support
- 5) A PDF file, via Acrobat PDF Writer
- 6) A fax, via Windows' native fax support
- 7) A fax, via WinFax Pro
- 8) A fax, via MAPI
- 9) An email, via MAPI

2.2 Beginning Assumptions

Print Wizard starts off assuming several things about how you wish to print. Any of these assumptions can be overridden, in several ways, as described later.

Print Wizard's beginning assumptions are:

- 1) that you wish to print on the printer identified by Windows as your default printer.
- 2) that it will use the Windows printer driver; that is, do "high level" printing.
- 3) that paper size, orientation, bin, duplexing (2-sided printing), etc. are as selected in the "properties" of the chosen printer, at the Windows level.
- 4) that printing should be done in the "Courier New" font. This font is found on every Windows system, is scalable in each direction, has good character set coverage, and is mono-spaced. Furthermore, auto-font switching is enabled; see below.
- 5) that printing should begin as high and as far to the left on the page as the printer is capable of printing.

Again, any and all of these assumptions can be overridden in a number of ways.

2.3 Print Level Switching

Print Wizard starts off by assuming that it will print at a high level; that is, it will use the Windows printer interface to draw characters (and other things) on each page. This high-level print is Windows' preferred (and assumed) way of printing, and provides access to the Windows spooler functions, as well as support for printers connected via serial, parallel, network, USB, and other means.

This approach is a problem if the data stream contains escape sequences, such as a sequence that puts an Epson printer into compressed mode. This could be true if the data stream is being generated by an application program that knows (or thinks it knows) what kind of printer you have, and is trying to control that printer. If printed at high level, these escape sequences would be "drawn" on the page, and they would not be obeyed; that is, they would not affect the printer's printing. So print jobs of this type must be printed at "spooler" level (however, if Print Wizard is configured to translate PCL, and it sees escape codes, it will go into its PCL translation mode).

Likewise, if the print job contains PostScript code, printing it at high level would result in a listing of the PostScript instructions, instead of the intended printout. Print Wizard detects PostScript jobs by seeing if they start with "%!". These print jobs must also be output at spooler level.

So if Print Wizard detects either PostScript or escape sequences (and it is not translating PCL), it will switch automatically to spooler level. In spooler level, Print Wizard writes directly to Windows' spooler, bypassing the printer driver. The printer can still be connected via parallel, serial, network, JetDirect, etc.

(Earlier versions of Print Wizard used "raw" level; this has been superseded by the spooler level).

The remaining sections describing Print Wizard's text handling do not apply to spooler mode.

2.4 Basic Text Handling

➤ This section assumes that Print Wizard is writing text at a high level (see above).

A font has been chosen by some means, and character size, spacing, and margins in each direction have been determined (more on that later). What follows is a description of Print Wizard's basic text handling.

Print Wizard assumes line-oriented text, with a linefeed (hex 0A) and optionally a carriage return (hex 0D) at the end of each line. It does not currently have the capability of doing freeform text. Each line of input is assumed to be intended as

one line of output. If lines are too long to fit within the margins, the line will be broken at the character (not the word); no characters are ever lost (unless margins have been set outside the area that that printer can print).

If a tab (hex 09) is encountered, Print Wizard will treat it in one of two ways. In the classic approach, Print Wizard assumes that tab stops are every 8 columns, and will advance to the next tab stop. However, if *autotab* is on, Print Wizard will treat the job as a tab-separated-values file – see below.

If a backspace (hex 08) is found, Print Wizard checks to determine whether it is being used to cause overstriking of the same character (backspace bolding) or underlining. It will convert these cases to actual bolding and underlining, respectively. For other cases of backspacing, Print Wizard will overprint the characters as indicated.

If a Field Separator (hex 1C) is found, Print Wizard will adjust the horizontal position to align fields; more on that later.

If a carriage return without a linefeed is encountered, Print Wizard will print the next line over the top of the current line. Some legacy reports use this technique for bolding and/or underlining. Print Wizard will print the characters correctly, but will currently not convert the data to a bold or underlined font.

When a formfeed (hex 0C) is found, or when no more lines will fit on the page, Print Wizard will advance to the next page.

Characters above hex 7F are printed according to the current *character set*; more on that later.

Print Wizard will also attempt to avoid printing blank pages at the beginning of the print job.

2.5 Auto-fit Logic

Print Wizard attempts to determine the best way to fit the text to the page. It does this by analyzing the first several hundred lines of data. In a plain text print job, this begins at the beginning of the file. In a marked-up print job, it begins when a `<pre>`, `<listing>`, `<xmp>`, `<legacy>`, or `<plaintext>` tag is found.

Print Wizard makes its decisions based on convention and readability. For instance, it is conventional to have 10 character per inch horizontal spacing, and it is conventional to have 66 lines per page. But it would not be readable to have more than 20 characters per inch.

Print Wizard analyzes lines to determine the maximum number of characters per line. In doing this, it knows that certain Asian characters are double wide, and that certain characters are combining marks. It then checks to see the page size selected, or more precisely, the printable area. This is based on information returned by the Windows printer driver. Based on these factors, it determines character width, pitch, and left and right margins.

It also checks to see whether the data is paginated; that is, if it contains formfeeds. If so, it determines maximum lines per page. If the data does **not** contain formfeeds, Print Wizard attempts to do "linage guessing". It will look through the print data, looking for patterns of blank or similar lines, as well as for occurrences of the word "Page", and do a best guess as to what is the implied page length. If it can not make a good guess, it will assume that 66 lines equals a page.

Once linage is determined, Print Wizard will test whether that number of lines can be printed reasonably on the page. If not, it assumes 6 lines per inch. With a lines-per-page measure determined, it calculates character height, line spacing, and top and bottom margin.

By default, Print Wizard will assign the minimum possible margins for the top and left; that is, printing will begin as high and as far to the left as the printer is capable of printing. This is generally the same point at which the printer will print if no printer driver is involved, so text positioning should be correct for preprinted forms. However, there is also an option to have Print Wizard assign "nice margins"; that is, margins up to ½ inch at the top and left. This may be preferable if the printer can print to the extreme edges of the page. Setting options is described elsewhere.

Any of these calculated values can be overridden with markup tags. Some of them can also be adjusted in the *print preview* screen.

-
- You can get an inside look at Print Wizard's calculations by turning on the *debug printing* option. This will cause Print Wizard to display information about its calculations, as it goes along.
-

2.6 Mini-page Logic

Print Wizard has an additional feature that makes it easy to print a small number of lines, such as might occur when printing one label at a time, or when printing variable-length receipts.

If a) the selected paper feed is 8 (tractor), 7 (auto), or 15 (automatically select), and b) you haven't specified paper length, etc., and c) the printer can print to the very top of the page, and d) the number of lines in the print job would constitute less than 80% of a page, if printed at 6 lines per inch (or specified linespacing), then Print Wizard will automatically adjust the page size to exactly fit the text, at 6 lines per inch (or specified linespacing).

So suppose Print Wizard gets a print job containing exactly 12 lines, and it is configured to use a tractor-feed printer with letter-size paper. After applying the tests above, it configures the print job for a custom paper length of 2 inches, then prints the 12 lines. This can be repeated indefinitely, and each print job will use 2 inches of paper. Thus, if continuous-feed label stock is used, with 2-inch vertical spacing, printing will be correct and no labels will be wasted.

If the printer driver doesn't identify the paper source as Windows standard "tractor", you may need to specify your BIN as "8", "7", or "15" using the markup language as described below.

If you specify linespacing, that linespacing will be used in the calculation above. You can specify linespacing directly, or you can do so implicitly by specifying a *font size*.

If you specify a *topmargin*, that amount will be added into the calculation of page length.

2.7 Auto-rotation Logic

One of the most basic formatting questions is orientation: which way on the paper should the job be printed. The usual options are portrait and landscape.

Print Wizard has an additional option called AUTO. If ORIENTATION has been set to AUTO (as explained later), Print Wizard will start off assuming an orientation of portrait. However, if one of two thresholds is crossed, it will automatically switch to landscape.

The first threshold is characters per line. The default is 100; that is, if Print Wizard detects at least 100 characters of data in the widest line, it will switch to landscape. You can override this default with a parameter "ROTATEWIDTH=n", where n is a number of columns.

The second threshold is the width of the character. After Print Wizard analyzes the print job, and determines the length of the longest line, it calculates how wide each character needs to be (the pitch). If the needed width would be less than or equal to a certain value, it will switch to landscape. The default is 1/16 of an inch. You can override this with the "ROTATEPITCH=n" parameter, where n is a value in DOTs or specified units.

2.8 Character Set Issues

When Print Wizard receives characters beyond the ASCII range (i.e., greater than hex 7F), it has to know what character set they are in, in order to print them correctly. The character set is specified in different ways, depending on the form of Print Wizard that is being used. Print Wizard is able to handle characters in the following sets:

- ISO: The "Windows" character set (varies by national Windows version)

- OEM: The "DOS" character set (varies by national Windows version), usually containing line-drawing characters
- UTF-8: Unicode characters, encoded in the UTF-8 scheme

In AnzioWin (which contains Print Wizard logic), there is also the capability to translate passthrough print data according to several additional character sets.

When Print Wizard is analyzing a print job to determine maximum line width, it takes into account the fact that most Far East characters are rendered twice as wide as Latin characters.

When Print Wizard prints the line-drawing characters found in the OEM (DOS) character set (also in UTF-8), it will use internal bitmap characters, to ensure that lines print as connected properly.

Print Wizard has some support for bidirectional languages (Arabic, Hebrew), on Windows platforms that include such support. This includes localized versions of Windows 95 through ME, localized NT, and all versions of 2000, XP, and 2003 (if that feature has been installed and enabled).

2.9 Automatic Font Switching and Diacritics

Print Wizard has sophisticated handling of "font coverage". That is, it will detect when the font it is using does not contain the character it needs to print. When that happens, it will search through other fonts available on the Windows system to find one that contains the needed character. If the data being printed is mono-spaced, it will try first to find a mono-spaced font that will work; failing that, it will check variable-spaced fonts also.

Print Wizard also takes special care with combining diacritics. If it needs to print a character/diacritic combination that does not exist in any font, it will print the base character, and then try a variety of techniques to find the needed diacritic: a non-combining form, an alternate form, a replacement character, or an internal bitmap.

All this happens automatically, so that Print Wizard can render successfully an extremely wide range of characters. As with other aspects of Print Wizard's behavior, you can control font usage with markup tags, as explained below.

2.10 AutoTab and Tab Handling

Print Wizard has two approaches to interpreting tab characters: *classic* and *autotab*.

2.10.1 Classic tab handling

In the classic approach, it is assumed that tabs are set at every 8th column, starting at column 1 (so 1, 9, 17, 25, etc.). Print Wizard keeps track of a "column counter" as it prints (even with a variable-pitch font). When it encounters a tab character (control-I, hex-09), it advances to the next tabstop.

2.10.2 Autotab tab handling

When *autotab* is active, however, an entirely different approach is used. This approach is provided so that you can easily print tab-separated-values (TSV) files. These files can be exported by various applications, including Microsoft Excel, for instance. The intent of Print Wizard is to print these files in a logical, readable manner, without user intervention.

The TSV file must follow this format:

- 1) The file is a line-oriented text file, with linefeeds or return/linefeeds between lines.
- 2) Each "record" in the file is in one line.
- 3) Fields (columns) within a line are separated by tab characters. Fields may or may not be padded with spaces.
- 4) Each line that contains tabs should have the same number of tabs, and therefore the same number of fields.
- 5) The **first** line that has tabs may or may not be a header line, with text describing each column.

- 6) There may be lines in the file that have no tabs. These are treated as normal text.

When Print Wizard analyzes the file, before printing, it will determine the maximum width needed for each column, adding one space for separation between columns. In total, this determines the maximum line size needed for the report; this will affect the auto-fit mechanism in Print Wizard. If the maximum line size is too big to be legible (would print at more than 20 character per inch), Print Wizard will adjust the tabs. This results in a multi-line output for each input line.

Print Wizard will also analyze the data in each column to see if the data appears to be numeric in all fields within the column (but the first tabbed line is ignored, because it might be a header line). This determines whether the column is a numeric column.

As it prints the data, Print Wizard will follow these rules:

- 1) When a tab character is encountered, it will advance to the next tab stop.
- 2) If the data hits the right margin, it will wrap to the next line (on a character boundary, not a word). This will not disrupt the tab handling.
- 3) If a column is numeric, it will be printed right-justified. The header text for a numeric column will also print right-justified.
- 4) If the file is generating multi-line output, Print Wizard will draw a thin line below each record (input line), and will not do a page break within a record.
- 5) Lines with NO tabs will be printed as normal text.
- 6) This logic will work in either a fixed-space or a variable-space font.

3 Programming for Print Wizard

This section describes how you can “speak Print Wizard’s language”. If you have access that lets you change the content of the file that you feed to Print Wizard, you can control Print Wizard’s printing, a little or a lot. Even if you can’t alter the original file, you can tell Print Wizard to use another file as an initialization file.

3.1 *Column Alignment with FS Characters*

Print Wizard provides a fairly simple way to create reports with aligned columns, even when printing with variable-spaced fonts.

Reports generated by many legacy programs achieve alignment by simply using spacing. For instance, the "customer name" field in every line begins at column 30. This works fine, as long as it is printed in a mono-spaced font – one in which every character has the same width (such as the Courier New font). However, if you instead print in a variable-spaced (proportional-spaced) font, such as Arial, the alignment is destroyed. Customers have asked for an easy way to have both – column alignment and variable-spaced fonts.

You can achieve this using the "Field Separator" character, or FS. This control character has a value of hex 1C. Here is how it works.

When Print Wizard first analyzes the print job, it determines an intrinsic *character width*, based on the number of characters per line. This value is retained for the entire report.

When Print Wizard prints each line, it keeps a *column count* – the number of characters that have been printed.

Whenever it encounters an FS character, Print Wizard does a realignment. It sets the horizontal printing position to the product of the current column count times the intrinsic character width. So variations that have happened because of differing character widths are discarded.

Furthermore, if there are one or more space characters immediately after the FS, Print Wizard advances the width of the zero character, which is usually also the width of each numeral. That way columns of numbers will align properly in most cases.

3.2 *Print Wizard’s Markup Language*

Print Wizard has a markup language to allow you to control most aspects of the printing. This markup language is contained within the file being printed. So to change the printout, you may need to be able to alter the program that generates the print job. However, it is also possible to put some initial markup into a *print initialization* file, which Print Wizard will process before the primary print job. This way you can alter the printout *without* changing the original program.

Print Wizard’s markup language (PWML) is based on HTML. This has several advantages. It is known by many programmers and web designers. It is based entirely on printable ASCII characters. It can be created with any programming language. And it adapts easily to dealing with text-only data.

But HTML is lacking in two key areas: precision and pagination. It does not allow precise placement of print elements. And because it is designed primarily for the screen, not the printer; it has no concept of a page break. (Some of these issues are being addressed, with CSS for instance). So Print Wizard’s markup language extends HTML in these and some other areas.

Conversely, Print Wizard is targeted at printing “line printer” type legacy printouts. Thus it does not (yet) handle word wrapping. However, it can print some kinds of HTML documents, such as those that use <listing> or <pre> tags. In this document, we will use “PWML” and “HTML” as virtually synonymous. In actual use, the initial tag (see examples below) can be either <PWML> or <HTML>.

Print Wizard allows you to move incrementally into the markup language. Suppose you have a print file that contains 132 columns of data. If you feed this to Print Wizard when it is set to portrait orientation, Print Wizard will compress the font horizontally in order to fit 132 columns onto the page. But now suppose you require that report to print out in landscape orientation. You can simply add to the beginning of the file:

```
<HTML><pagesize orientation="landscape"><BODY><PLAINTEXT>
```

and you will accomplish your objective. Similarly, you could specify other “job-level” parameters such as paper size, bin selection, font, line spacing, and lines per page. You could even specify a form overlay (or watermark). But the original text would not need to be changed at all.

In fact, you can place these initial sequences in a separate file, and instruct Print Wizard to prepend (insert at the beginning) that file. This is called a “print-init file”.

At the next level, to use bolding, italics, or underlines, you could insert simple tags such as “” to start bolding and “” to end bolding, in the body of your report. You could also indicate special characters, such as “€,” for the Euro character. You would need to change the <PLAINTEXT> above to <PRE>, so that tags would be recognized.

But in <PRE> mode, your existing data might contain things that looked like tags (“<”) or like character entities (“&”), so you’d have to change “<” to “<,” and change “&” to “&,”. To avoid that need, there is <LEGACY> mode. In <LEGACY> mode, tags (starting with “<”) and character entities (starting with “&”) must be preceded by a “trigger” control character, which is normally control-Z (hex 1A); otherwise, they are not recognized, but are printed as plain text.

Finally, you could advance to including bitmaps, barcodes, rectangles, etc.

A later section explains the details of Print Wizard’s markup language.

3.3 Using Initialization Files

Regardless of how Print Wizard is invoked, it is possible to specify a print initialization file. This file should be a plain text file, created with an editor such as NOTEPAD or EDIT. Whatever is in the initialization file will be inserted onto the beginning of the data file. The initialization file can contain escape sequences, or, more commonly, PWML initialization tags. It can also contain actual text to be printed at the beginning of the print job.

The file can be located on a local or networked disk drive, or it can be on an HTTP, HTTPS, or FTP server.

If the file contains escape sequences, that will force Print Wizard to switch to spooler mode, so that the initialization file and then the data file will be sent at a very low level to the printer, bypassing the Windows printer driver. The escape code sequences must be appropriate for the kind of printer involved. This is a less-common use of printer initialization files. (It would also be possible to do PCL translation of these codes.)

More often, the file will contain PWML tags. These can initialize the Print Wizard process, establishing job-level parameters such as paper selection, orientation, font, overlay, etc. The initialization file will generally leave the print process in a mode such as PLAINTEXT or LEGACY.

For instance, the following initialization file will cause reports to print in landscape mode, with an assumed length of 60 lines per page (if there are no formfeeds), with a top margin of ¾ inch:

```
<PWML><Pagesize orientation=landscape length=60>  
<BODY topmargin=.75in><legacy>
```

The following file would cause reports to print with a bitmap form overlay, and text margins set to 1 inch at top and left:

```
<PWML><pagesize overlay=myform.gif><body topmargin=1in leftmargin=1in><pre>
```

Note that line breaks before <legacy>, <pre>, <listing>, or <xmp> are ignored (both in the initialization file and in general usage). Furthermore, a single line break at the end of the initialization file is ignored.

The SAMPLES directory contains some sample print-init files. The file "LETTER.INIT" initializes Print Wizard to letter-size paper (8/5" by 11"), auto orientation, with convenient margins. The file "LABELS.INIT" initializes Print Wizard for 2-column labels. Files starting with "Gaylord" are for laying out labels on standard labels as sold by Gaylord, a library supply company. You can use these files as starting points for your own experimentation.

4 Print Wizard's Photo Printing

The freestanding Print Wizard program can also assist with what *should* be the simple task of printing a photo; that is, a file in bitmap format with a file extension .BMP, .GIF, .JPG, or .JPEG.

When Print Wizard is told to print a file with one of these extensions, it will print it as large as possible, given the printer's printable area. That is, the bitmap image will be stretched to fit the page. If the picture is wider than it is high, Print Wizard will print it in landscape orientation. Other options, such as paper type and graphics resolution, can be set in printer setup.

It is possible, as explained below, to tell Windows to use Print Wizard as the designated printing program for various file types (extensions). When you do this for one of the bitmap types, such as “.JPG”, it provides a very easy way to print your photos. While browsing a particular directory (in Windows Explorer), you can right-click on a JPG file. The popup menu that appears will include an option “print”. Just select that option, and Windows will use Print Wizard to print the file.

Also, you can create a shortcut icon for a particular printer on your desktop. Then you can drag-and-drop a file of the appropriate type onto that printer, and Print Wizard will print it on the indicated printer.

See “Examples: Default print processor”, page 26.

5 Metafiles

Windows has long supported metafiles. A metafile is in essence a recording of print operations, that can be “played back” at a later time. The first implementation was of Windows Meta Files, which carried a “.WMF” file extension. Later, the structure was enhanced to carry more information, and these became Enhance Meta Files (EMFs). It is important to note that a WMF and an EMF can contain only one page of output.

-
- Many graphics programs can produce WMF or EMF files. If you have a choice between WMF and EMF, use EMF.
-

Later versions of Windows (NT and later) introduced a variation of EMF that is a “spool file”, and carries an extension “.SPL”. Windows uses these as a temporary storage format for print jobs on their way to the printer. An SPL file can contain more than one page, where each page is embedded as an EMF. It is possible to capture the SPL file that is generated by virtually any Windows program printing to any Windows printer, as described below.

Print Wizard can read WMF, EMF, and SPL files, and use them in various ways, including:

- As an overlay. An SPL file can be a multi-page overlay.
- As an included picture, similar to a bitmap. A metafile may be a smaller and more efficient file than the corresponding bitmap file.
- To be printed directly.
- To be sent as a fax.
- To be converted to a PDF file.

Print Wizard can also **produce** EMF files from any of its usual input sources.

5.1 *Capturing an SPL File*

As mentioned, Windows NT and later uses the SPL format to store a print job on its way to the printer; that is, while it is “spooled”. With a few easy steps, it is possible to capture the SPL file. For instance, you might have a form that you have defined in Microsoft Word. If you print this from Word, and capture the SPL file, Print Wizard can use this as a form overlay.

5.1.1 **Locating the SPL Directory**

To capture a SPL file, you have to know where it will be created. The directory where Windows holds the SPL files is usually “\windows\system32\spool\printers”. Under Windows NT, it may instead be “\winnt\system32\spool\printers” or \winnt\SYSTEM\CurrentControlSet\Control\Print\Printers.

It is possible, however, to configure Windows to use a different directory for all printers, or for a specific printer. To find your PC’s general spool directory:

- 1) Go to the Control Panel
- 2) Click on “Printers” or “Printers and Faxes”
- 3) In the File menu, click on “Server Properties”
- 4) Click on “Advanced”
- 5) Note what is in “Spool folder”.

If in the steps below you fail to find a particular spool file, you may want to check the Windows Registry, under one of the following areas:

- HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Print\Printers\DefaultSpoolDirectory

- HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Print\Printers*specific printer name*\SpoolDirectory

Once you have identified the spooler directory, proceed as follows.

5.2 The Capture Process

The trick to capturing a SPL file is to get it to stay in place long enough for you to copy it. You can do this by either pausing the printer (at the Windows level), or telling Windows to retain the print job after it is printed. The following steps detail the first method:

- 1) Open the Control Panel's "Printers" or "Printers and Faxes" applet.
- 2) Double-click the icon for the printer in question. Ideally, there will be no print jobs showing.
- 3) From the Printer menu, choose "Pause printing".
- 4) Print the document you want to capture, on the chosen printer, using whatever Windows software is appropriate. You should see the print job show up in the printer window on your screen.
- 5) Look in the spooler directory as identified previously. You should find a file with a ".SPL" extension (hopefully only one). Copy this file to some other location on your computer.
- 6) In the printer's window, cancel the print job.

You now have your SPL file as copied in step 5.

6 The Scanner as Input File

In any place Print Wizard might otherwise use a bitmap file, it can instead be told to pull an image from a scanner attached to your PC; in fact, from any TWAIN source. To specify this, you refer to the scanner by a special file name: "SCAN://". The name is NOT case sensitive. Whenever Print Wizard goes to open a file, it looks for this special name and, if found, it uses the TWAIN interface to read an image. Then the image is treated the same as another bitmap file.

Print Wizard will present a dialog with two buttons: "Select" and "Acquire". Click on "Select" to choose which TWAIN source to use (if you have more than one). Click "Acquire" to activate the scanner's user interface. What you see from this point on will vary, because this is provided by the scanner's driver. When you actually scan an item, control will return to Print Wizard.

In the scanner's user interface, you can generally select the color depth and scan density (dots per inch) to be scanned. It is best to match these to your printer (or other output source) for maximum clarity and minimum processing.

Following are some suggested uses and notes:

- SCAN:// as the main file: The scanned image will be printed as large as possible on the page.
- As an overlay: The scanned image is used as a form overlay for the print job. For best results, scan at the same page size as you are printing (edge-to-edge).
- As a PREVIEWOVERLAY: The scanned image will be shown in the Print Preview window, but will not be printed. You might want to scan the preprinted form or label stock that the job will be printed on, and use the Print Preview dialog to adjust the spacing of the text to properly fit.
- As an included graphic: A PWML print job could call for an tag where "SRC" was identified as "SCAN://". As it prints, it will prompt you to scan something. You could include a picture or a signature, for instance. If multiple tags are specified in a document, you will be prompted for a scan for each one.
- Scan to Fax: When the output is one of Print Wizard's fax mechanisms, you can achieve programmatic faxing from a scan.
- Scan to PDF: You can scan in image and convert it to a PDF file, using either Print Wizard's native PDF generation or Adobe PDF Writer.

7 Using Print Wizard

Because Print Wizard exists in several forms, there are several ways to use it:

- Inside AnzioWin, via passthrough print, file transfer, etc.
- The freestanding Print Wizard program (PRINTWIZ.EXE) can print files delivered by various ways.
- The PWLPD program functions as an LPD server, and passes print jobs off to PRINTWIZ.EXE.
- The Print Wizard DLL (PWDLL.DLL) can be called from other programming languages.
- The Web Print Object (WēPO) is an OCX (ActiveX module) that can print jobs from web pages.

The following sections explain how to use Print Wizard.

7.1 *Inside AnzioWin*

AnzioWin is our telnet (terminal emulation) program for Windows. More information can be found at:

<http://www.anzio.com>

We also have Anzio Lite, but it does not include the Print Wizard logic.

When Print Wizard is turned on in AnzioWin, it affects a) print screens, b) the PRINTFILE command, c) CAPTURE to printer, d) certain COPY commands, e) certain file transfers, and f) passthrough print, sometimes called aux print, transparent print, or attached print. If you are unfamiliar with passthrough print, see our documents at the web site above.

If the program generating the printout on the host system has some capability for specifying printer types, it may insert escape sequences, especially at the beginning of the print job, to format the printing. If Print Wizard sees these (and it is not translating PCL), it will switch to “spooler” level, and will do no further processing of the job. In this mode, the host program is “in charge”. That is, its escape sequences are responsible for configuring the printer. If they are wrong, there is nothing Print Wizard can do about that.

However, if you can tell the host system that you have a “generic” printer, it should send out plain text. This will allow Print Wizard’s logic to control the format of the printout.

With AnzioWin’s menu system, the user can select several things that influence Print Wizard’s operation, including:

- File:Printer Setup: Choice of printer, orientation, bin, paper size, duplexing (2-sided printing)
- File:Printer Font: The initial font *name* that will be used
- File:Print Level:Spooler Setup: Which printer will be used for spooler mode
- File:Flush Timer: Determines end of print job
- Communicate:Character Set: In what character set is the data coming from the host?
- Edit:Advanced Options:General:Debug printing: Turn this on to get information on Print Wizard’s decision making
- Edit:Advanced Options:General:Printing initialization file: Specify a file to be inserted at the beginning of the print job.
- Edit:Advanced Options:Print Wizard: Options described below

➤ The font *size* chosen (in Printer Font) does NOT affect Print Wizard output, since font size is automatic.

Following are some specific notes.

In AnzioWin you can specify a different printer to be used for jobs printed at a spooler level, using File:Print level:Spooler setup. If you do so, then when Print Wizard detects escape codes (or PostScript code) and switches to spooler level, it will also switch to a using a different printer.

AnzioWin also lets you control (turn on or off) certain features of Print Wizard. This is done by going to the Edit menu, then to “Advanced options”, and finally to the “Print Wizard” tab. There you will see a checkmark for each of the following features:

- Print Preview Should the Print Preview window be displayed before each print job?
- Auto Orientation If on, orientation is assumed to be *Portrait*, unless there are more than 100 characters across, or text would be narrower than 16 characters per inch.
- Create Mini-pages Controls Print Wizard’s mini-page feature, as described in this document.
- Guess Linage Controls the *linage guessing* feature, as described in this document.
- Nice Margins If on, Print Wizard will create balanced margins, up to ½” top and left margins. If off, printing will start as far left and as high as possible.
- Translate PCL If on, Print Wizard will translate PCL print jobs and overlay files for use on any Windows printer.

7.2 The PRINTWIZ.EXE Program

The freestanding Print Wizard program is completely contained in the file PRINTWIZ.EXE. This is a console-mode program that ordinarily has NO user interface. It is driven completely by command-line parameters. Thus it can be initiated by another program, a desktop icon, a batch string, etc.

Print Wizard can be initiated to print a particular file, a group of files as specified by a wildcard, or any files that get dropped into a particular directory (which might be a networked drive). It can receive files sent to a particular TCP/IP port, using the “Jet Direct” protocol. It can even process and print files that exist on web or FTP servers. It can log its progress to the screen or to a disk file.

Printwiz.exe starts with an assumption that it should print in the Courier New font, on the Windows default printer. Properties such as paper size and orientation are taken from that printer’s properties as set in the Windows control panel. Print Wizard also assumes that incoming text data is in the DOS (OEM) character set.

7.2.1 Starting Print Wizard

As a console mode program, Print Wizard is started with a textual command, which may include parameters. There are many ways to do this in the Windows environment.

- If you have a console window (command window, DOS window) open, you can “CD” to the directory that contains Print Wizard, and type in the command as below. Print Wizard’s output, if any, will show in that window.
- You can click the “Start” button at the bottom of the screen, then “Run”, then type the command. You may need to include the program’s directory, such as `c:\printwiz27\printwiz somefile`.
- You can create a desktop shortcut. Each shortcut has associated with it a *target*, which is the command to execute.
- You can create (or modify) a start menu item, to run Print Wizard in the way you like. Print Wizard’s installation program creates several menu items for you.
- You can create a menu item in the Start menu’s “Startup” group, which will cause Print Wizard to execute automatically when you reboot.

Each of these methods has a command line associated with it. The following section details this command line.

7.2.2 Parameters

Print Wizard is started with a command in the form:

```
Printwiz [parameters] [filespec]
```

The *[parameters]* are optional, and are case **insensitive**. The *[filespec]* is also optional. Don't enter the “[” and “]”. Parameters are processed *before* any PWML tags in a print file, so PWML tags can override parameters. Possible parameters are:

/s	means do printer setup before printing each job. This allows you to select which printer to use, and also to choose printer-setup items such as paper size or “print to file”.
/l[logfile]	indicates logging. If no <i>logfile</i> is specified, logging will be to the screen.
/p"printer string"	sets which printer, by its Windows name, such as /p"HP LaserJet III". A space after the “p” is optional. If you have trouble determining how to specify a particular printer, run Print Wizard with the printer-setup (/s) option and the debug (/DEBUG) option, choose the printer in question, and note its name in the debug output.
/f[filename]	means print to file (or device, such as "COM3"). If a <i>filename</i> is not given, you will be prompted for it.
/fPDF://[filename]	creates a PDF file instead of printing. If <i>filename</i> is not given, the file will be "Printwiz_output.PDF".
/fPDF://[filename]?view	creates a PDF file instead of printing. If <i>filename</i> is not given, the file will be "Printwiz_output.PDF". After creation, the file is opened as configured by Windows, typically by Adobe Acrobat Viewer.
/fFAX32://phonenum	sends the output to the FAX32 feature, faxing it to the indicated <i>phonenum</i> .
/fEMF://[filename]	creates an EMF file instead of printing.
/oL	sets orientation to landscape.
/oP	sets orientation to portrait.
/oA	sets orientation to auto. Report will print in portrait mode, unless it's a wide report, in which case it will switch to landscape.
/bH	means duplex with horizontal binding, that is, along the short edge of the paper (requires duplex printer).
/bV	means duplex with vertical binding, that is, along the long edge of the paper (requires duplex printer).
/bL	means duplex with binding along the left edge of the paper, after orientation is determined (requires a duplex printer).

<code>/bT</code>	means duplex with binding along the top edge of the paper, after orientation is determined (requires a duplex printer).
<code>/bN</code>	turns off duplexing.
<code>/I</code>	means file is in ISO character set (otherwise assume OEM set).
<code>/U</code>	means file is in Unicode UTF-8 encoding.
<code>/vFONT="xxx"</code>	sets the initial font(s) to be used. To set the primary font and a list of alternate fonts, separate them with commas. See below.
<code>/OVERLAY=filename</code>	<p>specifies the name of a file to be used as a form overlay. The file can have an extension .BMP, .GIF, .JPG, .JPEG, .WMF, or .EMF; these will print on any printer. Or it can have an extension .PCL, in which case handling will be determined by the <code>"/TRANSLATEPCL"</code> option.</p> <p>The filename can be a URL, starting with <code>"http://"</code> or <code>"ftp://"</code>, and Print Wizard will fetch it from the indicated location. Or it can be the special name <code>"SCAN://"</code>, which will fetch a scanned image via TWAIN. (The older style <code>"/vOVERLAY=filename"</code> will still work.)</p>
<code>/PREVIEWOVERLAY=filename</code>	specifies the name of a file to be displayed as an overlay, only in the preview window. Otherwise this is treated the same as <code>/OVERLAY</code> above.
<code>/w</code>	means wait for despooling to occur.
<code>/d</code>	means to delete the file after printing. (Combine with <code>/w</code> for wait-then-delete).
<code>/r</code>	means to rename the file (to a unique name) while it is being printed.
<code>/k</code>	means keepalive (see below).
<code>/DESPOOL</code>	means to run in despooler mode (equivalent to <code>"/w /d /r /k"</code>).
<code>/e</code>	means eject paper at end of job.
<code>/cnnn</code>	sets baud rate to <i>nnn</i> if printing "raw" to a serial printer.
<code>/vLEVEL=xxx</code>	forces print level to <i>xxx</i> , which can be "HIGH" or "LOW" or "RAW" or "SPOOLER".
<code>/vTAGS=OFF</code>	turns off tag processing, in order to print HTML or PWML source.
<code>/DEBUG</code>	turns on debugging mode. This is very helpful for analyzing printing problems. (The older style <code>"/vDEBUG"</code> will also still work.)
<code>/vDOCNAME=name</code>	will use <i>"name"</i> as the document's name, which can be seen and tracked in the Windows spooler.
<code>/vUSER=name</code>	specifies a username (not currently used for anything).
<code>/LISTEN</code>	causes Print Wizard to "listen" on port 9100 for a "JetDirect" connection. This setting forces keepalive mode on, also.

<code>/vPORT=num</code>	causes Print Wizard to “listen” on port “ <i>num</i> ” for a “JetDirect” connection. This setting forces keepalive mode on, also.
<code>/INIT=filename</code>	tells Print Wizard to insert the text in file “ <i>filename</i> ” at the front of each print job, as a print initialization file. (The older style “ <code>/vINIT=filename</code> ” will also still work.)
<code>/vFORMAT=IBMFBA</code>	tells Print Wizard the input file is in IBMFBA format.
<code>/vEXT=xxx</code>	tells Print Wizard to register the extension “ <i>xxx</i> ” to be printed by Print Wizard. This parameter can be repeated, so that more than one file extension is registered in Windows. A dot before the “ <i>xxx</i> ” is optional (“ <code>/vEXT=.JPG</code> ”, for instance).
<code>/PREVIEW</code>	turns on the Print Preview feature.
<code>/VIEW</code>	turns on the Print Preview feature.
<code>/TRANSLATEPCL</code>	turns on PCL translation for main file and/or overlay; applied if they contain any escape codes. Allows printing PCL documents on non-PCL printers, or translating PCL to PDF.

The following parameters can be used when faxing with Print Wizard via WinFax Pro:

<code>/xphonenumber</code>	means to fax this print job to this phone number.
<code>/t“text”</code>	when used with <code>/x</code> , specifies the text of the “To:” line of the fax.
<code>/attachmentfile</code>	attaches the named file.
<code>/jsubjecttext</code>	includes the text literally as the subject of the fax.
<code>/nxxx</code>	if <i>xxx</i> is a filename, use the contents of that text file as the cover page; if not, use <i>xxx</i> as literal text.

➤ See Faxing and Emailing with Print Wizard

The *filespec* can be

- 1) a single file,
- 2) a dash (“-”), which means to use standard input (stdin),
- 3) a wildcard,
- 4) a URL,
- 5) SCAN://.

A filespec is NOT used if `/LISTEN` or `/vPORT` is used.

Examples are provided below.

7.2.3 Examples: Active mode

This is the mode where you simply tell Print Wizard to print a particular file. For instance, to print a file on the default printer:

```
printwiz myreport.txt
```

To print a group of files on a specified printer:

 **Rasmussen Software, Inc.**

10240 SW Nimbus, Suite L9, Portland, OR 97223 USA
e-mail: rsi@anzio.com

Phone: 503/624-0360 Fax: 503-624-0760
<http://www.anzio.com>

```
printwiz /p"My Deskjet" \temp\*.txt
```

Printers are identified by name. The name given must match what is shown in the Printers control panel.

To print a file from an FTP server:

```
printwiz ftp://somehost.com/pub/myfile.txt
```

Note that with this format, Print Wizard will do an anonymous connection to the FTP server. It is also possible to include a username and password for the FTP request:

```
printwiz ftp://username:password@somehost.com/directory/myfile.txt
```

➤ This will generally fetch a file relative to **root**, not to the user's home directory.

To print a file from a web server:

```
printwiz http://somehost.com/myfile.txt
```

7.2.4 Examples: Despooler mode

Print Wizard can act as a despooler, in which it runs in the background, watching a particular directory (or other wildcard spec) for files which have been created there and released (closed). For each file, it will print it as indicated, and delete it. It can optionally wait until the Windows spooler reports that the file is fully printed before deleting it. A typical use would be:

```
printwiz /k /w /d c:\spool\*.*
```

The filespec could also be a single file name. It can NOT be a URL, though; Print Wizard has no way to delete a file on an FTP or HTTP server, and so it would print it over and over again.

If the PC where Print Wizard is running can "see" a directory on a Unix/Linux host, such as with Samba or VisionFS, you can configure Print Wizard to act as a despooler for the host. Set up your Unix/Linux programs to print-to-file into that directory, and set up Print Wizard to print and delete them.

Note that if you're running an old DOS program under Windows, you may be able to configure the DOS program to "print to file" to a specified file name in the "c:\spool" directory, and thereby use Print Wizard to process your DOS program's output. If the program has some ability to drive different kinds of printers (inserting escape sequences), try configuring it to print to a "generic" printer, so Print Wizard can function fully. Or, configure it for an HP Laserjet III (or similar), and have Print Wizard translate the PCL:

```
printwiz /k /w /d /translatepcl c:\spool\*.*
```

If you add a "/r" parameter to the line above, Print Wizard will rename the file to a temporary, unique file name before printing (and then deleting) it. That way, if you start another print job during the time Print Wizard is printing the first print job, there will be no name collision.

A shortcut command is:

```
Printwiz /despool c:\spool\*.*
```

The "/despool" parameter is equivalent to "/k /w /d /r".

7.2.5 Examples: JetDirect printing

Print Wizard can mimic one of the protocols used by many printer network interfaces, such as the JetDirect devices from HP. We call this protocol “JetDirect” or “listen mode”. In this mode, a host computer writes print data to a particular IP address and port number. Print Wizard can monitor a port, process any data that is received there, and send the output to a printer. For instance, the command

```
Printwiz /vPORT=9100
```

or

```
printwiz /listen
```

will cause Print Wizard to “listen” on port 9100 (the default for this protocol), and route any print jobs received there to the Windows default printer.

See below for a description of the "netprint" program for Unix (-like) systems.

-
- The machine sending the data must be able to reach the PC running Print Wizard. That usually means that the PC must have a static IP address, unless the netprint program (described below) is being used.
-

7.2.6 Examples: Default print processor

You can tell Windows that you want to use Print Wizard to print files of certain extensions (such as “.TXT”, “.PTW”, “.PWML”, etc.). Then, in Windows explorer, you can point to one of these files, right-click, and choose “print”. Also, this technique is necessary when you want to use Print Wizard to fax certain files (such as with MAPISEND, explained below).

To tell Print Wizard to register itself as the “print” and “printto” handler for a particular file extension, such as “.PTW”, do

```
Printwiz /vEXT=PTW
```

A period before the extension is optional.

The command line can contain multiple parameters of this type, to cause Print Wizard to register itself as the print handler for multiple file types, i.e.:

```
Printwiz /vEXT=PTW /vEXT=.txt /vEXT=PWML
```

To **manually** create a new file type:

- 1) Start up Windows Explorer (not Internet Explorer)
- 2) Go to “View:Folder Options”
- 3) Click on “File Types”
- 4) Click on “New Type”
- 5) In “Description of type”, enter “Print Wizard document”
- 6) In “Associated extension”, enter a list of file extensions, such as “txt pwl”
- 7) Click on “New”
- 8) In “Action”, enter “print”
- 9) In “Application used to perform action”, enter the full pathname of the PRINTWIZ.EXE, followed by “%1” which represents the file to be printed. For example:

```
C:\printwiz\printwiz.exe "%1"
```

- 10) Click on “New”
- 11) In “Action”, enter “printto”

- 12) In "Application used to perform action", enter the full pathname of the PRINTWIZ.EXE, followed by appropriate parameters:

```
C:\printwiz\printwiz.exe /p"%2" "%1"
```

Or, you can add a "printto" action using Print Wizard to an existing file type, or change a "printto" action to use Print Wizard.

The "printto" action is what Windows uses to print a particular file type on a specified printer. The printer name will be inserted in place of the "%2", and the file name to be printed will be inserted in place of the "%1". For diagnostic purposes, you might want to add parameters "/DEBUG" and/or a log file designation.

You can also add various other parameters as listed above, to affect printing in various ways. For instance, you might register Print Wizard to be the designated print program for files with an extension ".utf8", and in that registration include the parameter "/u" to tell Print Wizard that the file is encoded as Unicode UTF8 characters.

7.2.7 Debugging Mode

When you turn on debugging mode (with "/DEBUG"), Print Wizard will provide some very useful debugging information, such as what printer it is writing to, whether it detects escape sequences, etc. It will also tell you all the analysis it does in its attempts to "best-fit" the document to the page. And it will tell you if it finds any markup tags it does not recognize. The debug output will go to the screen, unless a log file is specified (with "/Lfilename"), in which case it will be added to the log file.

7.2.8 The Unix *netprint* Program

The Print Wizard product (but not AnzioWin) includes a program for various Unix (and similar) platforms, called "netprint". This program is designed to work with Print Wizard's "JetDirect" protocol. It will also work with HP JetDirect and similar devices. It takes piped input and sends it to a particular machine name (or IP address) and port (default 9100).

You will need to identify which netprint object is correct for your host system, by noting the file extension. For instance, the file "netprint.sco" in the "netprint.obj" directory is for SCO Open Server 5. Check the README.TXT file for other versions. Contact Rasmussen Software if you need netprint compiled for another system. Once you've identified your object, move it to the host system, such as with FTP. Be sure to transfer it as binary. Rename it, make it executable, and put it in your PATH; for instance:

```
mv netprint.sco netprint
chmod +x netprint
cp netprint /usr/bin
```

The print data must always be piped into netprint. For instance,

```
ls -l | netprint
```

will send a directory listing to netprint, with a default port number of 9100.

But what machine will netprint send the data to? The netprint program has a unique feature in that it assumes you want to send the data to the same PC you are logged in from in a telnet session. (To do this, netprint looks at the utmp or utmpx file in Unix.) This results in a form of "follow-me printing". Regardless of where I log in from, even with dynamic IP assignments, netprint will send the print job to my IP address, where Print Wizard can process it and send it to the correct printer. Note, however, that this will not work if the Unix spooler is trying to process the output, because it will have lost track of the login name. So generally, to use netprint, you will want to pipe your print out to netprint INSTEAD of to the Unix spooler.

Of course, you can also specify a particular destination machine. The complete syntax of netprint is:

```
netprint [options]
```

where options include

```
-p nnn use port number nnn
-h xxx send to hostname (or IP number) xxx
-d nnn set debug level to nnn.
```

-
- Note that if the destination machine is behind a firewall (or other device) that is doing Network Address Translation (NAT), you will need to configure the firewall to direct these print jobs to the correct PC, based on the port number.
-

7.3 The PWLPD.EXE Program

The LPD/LPR protocol is a standard network protocol by which one computer on a network can print on another computer's printer. Most Unix, Linux, and similar systems, and Windows NT and later, can send print jobs out using this protocol. The PWLPD program, part of the Print Wizard package (but not part of AnzioWin), can serve as a print server for this protocol.

Configuring LPD/LPR on your host system will vary by platform. The end result is that the host system will send the print job out to a particular machine name (or IP number) and port (normally 515). The machine name/IP number has to match the PC that PWLPD is running on. The print job sent to the PC will also identify a "queue name", which tells PWLPD which printer to print this job on. More on that later.

When PWLPD is running, it will accept jobs sent to it with the LPD/LPR protocol, store those jobs on disk, and then invoke PRINTWIZ.EXE to print them. PWLPD can also track their progress via the Windows spooler, and feed status back to the sending system when requested.

-
- A separate file, PWLPD.DOC, explains the details of using PWLPD.
-

7.4 The Print Wizard DLL

Print Wizard is also provided as a callable module, PWDLL.DLL, that can be called from many programming languages, including Visual Basic (VB) and Visual Basic for Applications (VBA). This makes a very easy way to get basic or advanced printouts from these languages without delving into the Windows printer API. In fact, you can create printout with as little as one call.

-
- The DLL is included in the freestanding Print Wizard product. It is NOT included with AnzioWin or WẽPO.
-

7.4.1 Using the DLL from Visual Basic or VBA

The following VB subroutine shows the interfaces, and demonstrates the use of some:

```
Private Declare Function pw_configure_printer Lib "pwdll" (ByVal window As Integer) _
    As Long
    'returns 1 if successful, else 0
Private Declare Sub pw_set_printer Lib "pwdll" (ByVal lpchar As String)
    'set printer by name
Private Declare Function pw_print_text Lib "pwdll" (ByVal lpchar As String, _
    ByVal count As Long, ByVal cs As Long) As Long
```

```

        'returns 1 if successful, else 0
        'cs is 0 for ISO
        '    1 for OEM
        '    2 for other (no translation)
        '    3 for Unicode UTF-8'
Private Declare Sub pw_flush Lib "pwdll" ()
    'terminates print job
Private Declare Sub pw_set_html_base Lib "pwdll" (ByVal lpchar As String)
    'sets the html_base for relative filenames
Private Declare Sub pw_set_preview Lib "pwdll" (ByVal b As Long)
    'b is 1 for preview on
    '    0 for preview off
Private Declare Sub pw_set_orientation Lib "pwdll" (ByVal b As Long)
    'b is 0 for portrait
    '    1 for landscape
    '    2 for auto
Private Declare Sub pw_set_font Lib "pwdll" (ByVal lpchar As String)
    'specify font name(s)
Private Declare Sub pw_set_overlay Lib "pwdll" (ByVal lpchar As String)
    'specify name of overlay
Private Declare Sub pw_set_copies Lib "pwdll" (ByVal n As Long)
    'set number of copies
Private Declare Sub pw_eject_page Lib "pwdll" ()
    'equivalent to printing a formfeed
Private Declare Function pw_set_output Lib "pwdll" (ByVal lpchar As String, _
    ByVal count As Long) As Integer
    'returns 1 if successful, else 0
    'sets type of output
    '    empty string for printing
    '    PDF://          Generate PDF file with default name
    '    PDF://<filename>  Generate PDF file with given name
    '    PDF://<filename>?view  Generate PDF file with given name
    '    FAX32://<phonenumber>  Send fax
Private Declare Function pw_print_file Lib "pwdll" (ByVal lpchar As String, _
    ByVal count As Long, _
    ByVal cs As Long) As Integer
    'prints an entire file
    'returns 1 if successful
    'cs is charset as above

Private Sub Command1_Click()
    res = pw_print_text("This is a Print Wizard test", 27, 0)
    pw_flush
End Sub

```

Note that all integers are 32-bit.

In this example, only plain text is sent to Print Wizard. However, you can also send PWML markup tags, and so control all the print formatting that Print Wizard can do.

The Samples directory contains a Visual Basic source program "Testdll.vbp" that demonstrates calling the DLL.

7.4.2 Using the DLL from AcuCobol

Most of the features of Print Wizard can be used from an AcuCobol program running on a Windows PC. This includes auto-fitting text to the page, as well as generating fancier printing with font changes, pictures, form overlays, etc.

The SAMPLES directory (under the directory that contains Print Wizard) includes a sample source program for AcuCobol, named "TEST-PW.CBL". This program demonstrates using the PWDLL.DLL to print from COBOL. The program demonstrates several techniques, and is heavily commented.

7.5 Web Print Object (WēPO)

➤ Note: Web Print Object is a separately licensed product.

Another form of Print Wizard is as an ActiveX object, known as Web Print Object, or WēPO. The advantage of WēPO is that it can be run from a web page. With WēPO in place on a client PC, the web designer can precisely control printing on the client's printer. This can be thought of as "push printing", because the web site "pushes" the print job out to the client's machine, at the user's request.

The print job can include any of the features described here, including barcodes, form overlays, page size selection, etc. Also, this approach can be an alternative to having the web browser print certain reports that the browser may not handle well.

The process of printing from a web page always involves two files. One file is the web page that includes a "print" button; the other file is the data to print – the print job. The two files can be on the same or different servers.

7.5.1 The Web Page

The web page is a standard HTML file that includes an <OBJECT> tag that references WēPO. The object tag indicates how WēPO appears and how it behaves. Typically, it will appear as a pushbutton that says "Print".

The particular OCX is identified by its *classid*, a globally unique character string (see below). If the client's PC does not have the Print Wizard OCX installed, the client will be advised of that by the browser, and given the chance to install it. This may require altering, at least temporarily, certain security settings. Once it is installed, the pushbutton will appear.

The web page also specifies certain parameters to the OCX that govern its behavior. These are in <param> tags. Most critical is the "mainurl", which specifies the URL of the file that contains the print job.

A sample <OBJECT> tag is as follows:

```
<OBJECT
    classid="clsid:659D3554-54CD-46BC-B0F1-D88C4CCFD10C"
    codebase="http://www.anzio.com/controls/PWButtonXControl1.cab#version=2,7,0,0"
    width=100
    height=100
    align=center
    hspace=0
    vspace=0
>
<param name="Caption"           value="Click to print">
<param name="mainurl"          value="filetoprint">
<param name="debug"            value="1">
<param name="fontname"         value="courier new">
<param name="initfile"         value="c:\printwiz\_prtinit.txt">
<param name="LICENSEcode"      value="DFD5-NK45-MUKD-K7">
<param name="orientation"      value="auto">
<param name="printersetup"     value="0">
<param name="charset"          value="utf-8">
<param name="duplex"           value="left">
</OBJECT>
```

Following is an explanation of each part of the <OBJECT> tag:

Classid A globally unique character string by which WēPO is registered. It is required. This string may change with changing versions of WēPO. Contact us for the correct code.

 **Rasmussen Software, Inc.**

10240 SW Nimbus, Suite L9, Portland, OR 97223 USA
e-mail: rsi@anzio.com

Phone: 503/624-0360 Fax: 503-624-0760
<http://www.anzio.com>

Codebase	Tells the browser where the OCX can be installed from, if it is not already on the client's PC.
Width	The width in pixels of the button to appear on the screen.
Height	The height in pixels of the button to appear on the screen.
Align	Where to display the button, relative to the current printing position. Options are "texttop", "middle", "left", "textmiddle", "right", "baseline", and "center". Consult a book on HTML for specifics.
Hspace	The amount of space to the left and right of the object, in pixels.
Vspace	The amount of space above and below the object, in pixels.

Following is an explanation of the various <PARAM> tags. Some of these tags are optional; the MAINURL is required. In each case, you can change what is in the "value" part.

Caption	The text to appear in the middle of the button.
Enabled	A zero value marks the button as disabled (grayed), so the user can not click it.
Visible	A zero value makes the button invisible.
MainURL	The file to be printed when the user clicks the button. Can specify a file on the local disk, or a URL starting with ftp://, http://, or https://.
Debug	A non-zero value tells the OCX to run in debug mode, displaying a lot of useful information in a popup window.
Fontname	Specifies an initial font for printing.
Initfile	Specifies a print initialization file. Can be on a local disk, or a URL.
LicenseCode	If a licensecode is specified, and it correlates to the location of the MainURL, then the OCX will NOT print out a line indicating it was printed with a demo version. Licensecodes are provided by Rasmussen Software.
Orientation	Sets the printing orientation. Possible values are "portrait", "landscape", or "auto".
Duplex	Tells the printer to print on both sides of the paper, if possible. Possible values are "off", "vertical", "horizontal", "left", "right", or blank. If blank, the duplexing will follow the printer driver's configuration.
PrinterSetup	A non-zero value will cause WēPO to prompt the user for Printer Setup, allowing them to choose a printer and set its properties.
PrinterName	Indicates which printer to use, by its Windows printer name. After printing, this parameter will contain the name of the printer that was used, such as the one selected via Printer Setup.
Charset	Specifies the character set of the print data. Options are "ISO", "OEM", and "UTF-8". Default is ISO.
Overlay	The name of a file to be used as a full-page form overlay.
TranslatePCL	A non-zero value causes WēPO to translate any PCL codes in the main print job and/or the overlay file(s), allowing them to be printed on any printer.

CaptionFont	The name of the font used to display the caption in the button. The default is set by the font of the preceding HTML.
CaptionHeight	The size (in points) of the font used for the caption. The default is set by the font of the preceding HTML.
CaptionStyle	A string representing the style to be used for the caption. Include, in any order, "B" for bold, "I" for italics, and/or "U" for underline. The default is set by the font of the preceding HTML.
Printed	After the button is clicked, this parameter will be non-zero to indicate that the printing succeeded. This parameter would not likely be set by the web page.

7.5.2 The Print Job File

The print job, that is the content, comes from a text file on an HTTP or FTP server. Or, it can be generated using CGI techniques. The file can be plain text, text with escape sequences (that would be unusual, though), PWML, or HTML (as long as it was within the features handled by Print Wizard).

7.5.3 Samples

To see the Web Print Object in action, visit the web site <http://www.anzio.com/products/pwactivex.htm> .

8 Faxing and Emailing with Print Wizard

Print Wizard has several features providing several approaches to faxing and emailing. All are designed to allow *programmatic* sending; that is, sending without user intervention. Supported approaches include:

- The command line to PRINTWIZ.EXE can cause:
 - faxing via WinFax Pro
 - creating PDF files via Acrobat, which can then be emailed via MAPI
- Special commands *inside* the print data can cause:
 - emailing via MAPI*
 - faxing via MAPI
 - faxing via WinFax Pro
 - creating PDF files via Acrobat, then emailing via MAPI
- A command line to start MAPISEND.EXE command can cause:
 - emailing via MAPI
 - faxing via MAPI
- Printing to various fax printer drivers can initiate faxing, but you will be prompted for details
- PWLPD can route print jobs to various 3rd party fax and email programs

*MAPI is Windows' messaging protocol (Messaging Application Programming Interface), which can send emails and (in some cases) faxes.

8.1 Faxing and Emailing Components

This section provides some background on the components that Print Wizard has and uses to do faxing and emailing. Another section below will explain how to use these components.

8.1.1 The MAPISEND Program

This section will give an overview of the MAPISEND program which is supplied with Print Wizard. The details of configuring your Windows messaging, and specific command line parameters, are covered in section 16 of this manual.

➤ Microsoft also has a program of this same name. What we are describing here is *our* MAPISEND program.

The MAPISEND program is a command-line driven program to interface with the Windows MAPI calls. Programs of this type can be initiated by the user typing the command into a Windows "DOS box" or "command window". They can also be entered at the Start:Run prompt, included in a BAT string, initiated by various scripting programs, etc.

In its most basic form, MAPISEND can be used to email a text file to someone:

```
mapisend /emyfriend@somehost.com sometextfile
```

Here the "/e" parameter indicates the email address that MAPI is supposed to send *sometextfile* to. The file is sent as plain text. Other parameters can have various effects on how the file is sent.

Or, this command:

```
mapisend /f1-800-555-5555 somefile.doc
```

will fax the file *somefile.doc* to the indicated fax number. But this is more complicated. The MAPI system will do the following:

- Look in the *file types* information of Windows to see what program can print a file of this extension.
- Invoke that program to print the file, to the appropriate fax printer driver.
- Direct the output to be faxed to the indicated fax number.

So in this example, MAPI would invoke Microsoft Word, telling it to print the file *somefile.doc* to the printer driver *Windows fax*, and send the result out the fax/modem board. Again, other command-line parameters can affect the fax job.

If MAPISEND is going to use Print Wizard to fax a particular file type, that Print Wizard must be *registered* as the default print processor for that file type. Doing that is described in section 7.2.6.

So let's assume you have registered Print Wizard as the default print processor for files of type “.PTW”. If you execute the command

```
mapisend /f1-800-555-5555 somefile.ptw
```

then MAPI will invoke Print Wizard to render the file *somefile.ptw* to the Windows fax driver, and fax it out.

There are various ways, explained below, to use MAPISEND with Print Wizard.

➤ This technique can be used only with the Print Wizard *program*, not with other forms of Print Wizard.

8.1.2 WinFax Pro Support

Print Wizard has special support for WinFax Pro, version 10 and higher.

Any Windows program that prints, including Print Wizard, can print to the WinFax Pro printer driver. Ordinarily this would cause WinFax to pop up and ask the user for faxing details: fax number, recipient, etc. With the added support in Print Wizard, however, this information can be supplied programmatically, so no operator intervention is required.

See below for information on how to use this feature.

8.1.3 Acrobat Support

Print Wizard has special support for Adobe Acrobat PDFWriter.

First, Print Wizard can control the name of the file that PDFWriter creates.

Second, Print Wizard can automatically create a PDF file (using PDFWriter), email that file using MAPISEND, and then delete the file. This way you can email a PDF file to someone, with precise control over the content and format of that file.

8.2 Using the Faxing and Emailing Components

There are two ways you can tell Print Wizard (the program) to fax or email output.

The first method uses command lines inside the print file. These command lines all start with an exclamation mark (“!”), sometimes called “bang”).

The second method uses command-line parameters on the command that launches PRINTWIZ.EXE.

8.2.1 Internal “!” Commands

If the first character in a print job is “!”, Print Wizard assumes that the first line is a bang command. Specifically, the first line can start with “!MAPISEND”, “!WINFAX”, or “!ACROBAT”, each with additional parameters as described below. The “print job” then consists of all following lines until a) another bang command or b) the end of the job.

If the command is “!MAPISEND”, Print Wizard will copy the following print job into a new file, and then run the MAPISEND program with the given parameters. If those parameters tell MAPI to email the file, then the plain text will be emailed. If those parameters tell MAPI to fax the file, MAPI will use the file type’s default print processor (normally PRINTWIZ.EXE) to render that file to the fax driver, and send it off.

If the command is “!WINFAX”, Print Wizard will use the fax number, recipient name, etc. provided as parameters to the command, and will use WinFax to fax the print job to that fax machine.

If the command is “!ACROBAT”, Print Wizard will use PDFWriter to create a PDF file with a specific name, and then use MAPISEND to email that file to the indicated recipient. Then Print Wizard will delete the file.

Details of each command are provided in section 13.23.

8.2.2 Command-line Faxing

The command line that starts PRINTWIZ.EXE, as explained above, can contain command-line parameters. Some of these parameters are provided specifically for faxing (with WinFax Pro):

<i>/xphonenumber</i>	means to fax this print job to this phone number.
<i>/t“text”</i>	when used with /x, specifies the text of the “To:” line of the fax.
<i>/attachmentfile</i>	attaches the named file.
<i>/jsubjecttext</i>	includes the text literally as the subject of the fax.
<i>/nxxx</i>	if <i>xxx</i> is a filename, use the contents of that text file as the cover page; if not, use <i>xxx</i> as literal text.

➤ To fax using MAPI, use the MAPISEND command.

9 PDF Generation

Adobe Systems Incorporated has defined and published a specification called “Adobe Portable Document Format”. Files that are created in this format are commonly called “PDF files”, and are a popular way of storing, transferring, viewing, and printing data in a device-independent (portable) way.

Print Wizard is able to generate PDF files in the following ways:

- 1) Via Adobe PDF Writer, as a pseudo printer driver
- 2) Via other pseudo printer drivers
- 3) With its own built-in PDF generator

For the first two methods, Print Wizard prints to a printer driver just as though it were printing to a printer. However, Print Wizard can specify the output filename to be created.

For the third option, the built-in PDF generator, Print Wizard has complete control, and no other software is needed.

9.1 *The Built-in PDF Generator*

Print Wizard’s built-in PDF generator allows it to produce industry compatible PDF files with no additional software. This generator is developed entirely by Rasmussen Software. Following are some notes on how PDF is generated:

- Files are generated as PDF-1.4.
- Files are generated with a resolution of 600 dots per inch.
- Where reasonable, fonts are mapped to PDF’s built-in fonts. So Courier New and Lineprinter become Courier, Arial becomes Helvetica, and Times New Roman becomes Times (with variations for bold and italics). Also, System becomes Helvetica.
- Characters outside of the ANSI range will be output as bitmaps.
- There is support for ZapfDingbats and Symbol.
- Characters in other fonts will be output as bitmaps. There is no font embedding or use of non-standard fonts.
- Line-drawing characters are output as bitmaps.
- Output files are not in “linearized PDF” format.
- Graphics are not compressed.
- There are numerous other things that PDF defines that Print Wizard does not yet support.
- When generating a PDF, Print Preview is disabled.

9.2 *Things You Can Do with PDF*

Nearly everything that Print Wizard can do when printing can also be done when generating a PDF. Here are some things you might not have thought of:

- Print with form overlays
- Include barcodes
- Convert a bitmap file to PDF
- Translate a PCL file to PDF
- Convert a TSV file to PDF
- Print purchase order data, with a form overlay, adding a dynamically-scanned signature, and email it to a supplier

10 PCL Translation

PCL is the printer command language developed by Hewlett Packard, and supported by most, if not all, HP laser printers, some HP inkjet printers, and many printers from other manufacturers. It is a language, based on escape codes, that controls such things as paper size, font selection, character size, rectangle drawing and more. PCL sometimes includes HPGL, a vector-based language originally developed for plotters. HPGL is able to draw straight lines, arcs, polygons, etc. Many printers that support PCL also support HPGL as an alternate language. Finally, PCL is sometimes wrapped in PJI (Printer Job Language), a language that initiates and controls whole print jobs.

Print Wizard has built-in support for PCL, HPGL, and PJI, collectively referred to as “PCL translation”. This means that Print Wizard can read, understand, and process these languages, independent of a printer.

10.1 The Built-in PCL Translation

Print Wizard understands PCL 5e. This feature can be turned on and off as “TranslatePCL”. The mechanism to do this varies by which implementation of Print Wizard you are using. The same switch controls PJI and HPGL translation.

The built-in PCL emulation also includes emulation of the barcode fonts provided in HP’s barcode font cartridge.

We will not claim that support for these languages is totally complete. However, it should cover the majority of situations. This translation is constantly being enhanced. If you encounter a PCL file that is not translated correctly by Print Wizard, please let us know and, if possible, email us the file.

10.2 Where Do PCL Files Come From?

Many application systems, particularly on Unix/Linux, can insert PCL sequences into print jobs in order to control printing on laser printers. This can range from the minimal, such as a code to switch into compressed print, to control of the entire job.

It is also possible to create PCL files from most Windows applications. To do so, install (if necessary) a PCL printer driver on your Windows PC.

-
- Be sure you use a PCL 5 driver. PCL 6 is something entirely different. We recommend an HP Laser 3, 4, or 5 driver.
-

From your Windows application, select Printer Setup, and choose this driver. Checkmark the “Print to file” box. Now print your document. You will be prompted for a file name. Enter a file name or, if allowed, an entire pathname so you’ll know where the file ends up. Once you have located the generated file, we recommend you change its file extension to “.PCL”.

In this way you can make a PCL file from a Word document, an Excel spreadsheet, or a forms design program. This file can then be used as a form overlay, or as the primary document to be printed, faxed, or converted to PDF, as described later.

10.3 How You Can Use PCL Translation

With TranslatePCL turned on, you can do a variety of things:

- Use a PCL file as an overlay, even on a non-PCL printer (even a Windows-only printer).

- Use a PCL file as an overlay for a fax.
- Use a PCL file as an overlay in a PDF.
- Render a PCL file on any printer.
- Convert a PCL file to a fax.
- Convert a PCL file to a PDF file.
- Print a PCL main file on a PCL overlay, on printer, fax, or PDF.

➤ When a PCL file is specified as an OVERLAY or PREVIEWOVERLAY, and PREVIEW is on, Print Wizard will translate the PCL file to use as an overlay in the Preview window, **regardless** of the setting of TranslatePCL.

10.4 When TranslatePCL is OFF

Then TranslatePCL is turned OFF, Print Wizard assumes that you are using a PCL5-compatible printer. Specifically:

- If the main print file contains any escape codes, Print Wizard will switch to Spooler mode, passing the print job unchanged to the printer. Of course, if the escape codes don't correspond to what the printer can do, you won't get the results you want.
- If a file with a ".PCL" extension is referenced as an OVERLAY, Print Wizard will do some processing on it, to convert it into one or more overlay macros. These will then be loaded into the printer, and dynamically combined with the contents of the main file. It is also possible to have Print Wizard save these macros in the printer for later use.

➤ If you try this with a non-PCL printer, it won't work.

➤ For overlays, the *printer driver* must also be a PCL 5 driver. Some laser printers support PostScript, PCL-5, and PCL-6. Install and use the PCL-5 driver.

11 Form Overlays

A powerful and useful feature of Print Wizard is its ability to print forms as overlays. An overlay is any printing that is placed in every page of output, in combination with the text (or other content) that is coming from the main file.

-
- The term “overlay” is actually a misnomer. It should be “underlay”, because it is generally drawn *first* on the page, with the variable text over it.
-

When overlays are printed as part of the output, it creates a way to print on plain paper what would otherwise require a preprinted form. This can be a significant cost savings.

The classic illustration of an overlay is in printing invoices. You have a file of text that in the past has been printed on preprinted invoice forms. If you can create an electronic image of that invoice form (as described below), Print Wizard can print the invoice *plus* the text on each page. Another example would be a *watermark*; that is, an image or large text that prints lightly in the background of each page.

Historically, an overlay has consisted of one image, that was printed on every page of output. However, Print Wizard also supports *multi-page overlays*, in which the overlay images are cycled as text is printed.

Note also that Print Wizard can apply overlays to non-printed output, namely faxes and PDFs.

Finally, note that an overlay is always considered to be a full-page image. There is no mechanism for specifying the *placement* of text on the page.

An OVERLAY is always contained in a separate file. This file, like the main print file, can be on a local disk, a network drive, or an HTTP, HTTPS, or FTP server. You can specify that a particular overlay file should be used a) externally, such as in the command line for PRINTWIZ.EXE; b) in a print-initialization file; or c) in the PWML header of the main print file.

A special case of overlays is the PREVIEWOVERLAY. This is an overlay that is shown only in the Print Preview window. It is not actually printed. This can be useful as an image of the preprinted form (or label stock) that the data will be printed on. You can use the adjustment capabilities of the Print Preview window to make the data line up correctly on the overlay.

The way that Print Wizard handles an overlay file depends on its file extension. Make sure that's right.

Following are descriptions of the various kinds of overlays Print Wizard can support.

11.1 **Bitmap Overlays**

A bitmap overlay is simply a file that contains a picture that you want in the background. Some notes:

- Works on any non-generic printer
- Supports BMP, GIF, JPG, JPEG
- Image is stretched to the full size of the paper, including covering unprintable areas. If you are prescanning a form, scan it edge-to-edge so it will print at the same size.
- One-page overlay only.

11.2 Scanned Overlays

In this case, you scan something to use as an overlay at the start of the print job. You specify this by using the reserved filename "SCAN://". Some notes:

- Works on any non-generic printer, fax, PDF.
- Supports any TWAIN-compliant scanner
- Image is stretched to the full size of the paper, including covering unprintable areas. Scan it edge-to-edge so it will print at the same size.
- For best results, scan at the same dot density as the printer.
- One-page overlay only.

11.3 EMF and WMF Overlays

Any EMF (Enhanced Meta File) or WMF (Windows Meta File) can be used as an overlay. These files can be created by many graphics programs.

- Works on any non-generic printer, fax, PDF.
- Image is stretched to the full size of the paper, including covering unprintable areas.
- File extension "EMF" or "WMF".
- One-page overlay only.

11.4 SPL Overlays

An SPL (spool) file is even more enhanced than an Enhanced Meta File (EMF), because it can contain multiple pages. SPL files are created by capturing print output from other programs, as described elsewhere. Notes:

- Works on any non-generic printer, fax, PDF.
- Image is stretched to the full size of the paper, including covering unprintable areas.
- File extension "SPL".
- Can be multi-page. Pages are cycled as data is printed.

11.5 PCL Overlays

As previously described, a PCL file can be used as an overlay. Described elsewhere is how to create a PCL file using a Windows printer driver. Notes:

- With TranslatePCL on, works on any non-generic printer, fax, PDF.
- With TranslatePCL off, requires a PCL-5 printer and driver. This is more efficient.
- File extension "PCL".
- Can be multi-page. Pages are cycled as data is printed.

11.6 PWML Overlays

You can use a file containing PWML markup as an overlay. This would be appropriate if you wanted to *describe* the overlay using PWML tags, such as <LINE> and <HR>. The entire overlay file is processed first, and saved as one or more pages in memory (as EMFs). These are then added to the data from the main file. There is no interaction between the tags in the overlay and the tags in the main file. Notes:

- Works on any non-generic printer, fax, PDF.
- File extension "PWML".
- Can be multi-page. Pages are cycled as data is printed.

11.7 Comparison of Methods

The primary tradeoffs between methods are file size, speed, and portability.

A full-page bitmap is a large file, and it must be sent to the printer for each page of output. It is inherently inefficient.

A WMF, EMF, or SPL file *can* be small, depending on how it is produced. It must be sent to the printer for each page. Also, although these meta files are designed to be portable between versions of Windows and types of printers, there could be problems. Fonts specified in a meta file might not be present on a different PC.

PCL files can be quite small and efficient. Later drivers (Laserjet 5 as compared to Laserjet 3, for instance), tend to be smaller. They are very portable.

PWML files are small, efficient, and extremely portable. However, they must be created by hand.

12 The Print Preview Feature

Print Wizard has the option to open a Print Preview window, before printing each print job. In the freestanding Print Preview, this option is turned on with the “/view” or “/preview” command line parameter. In AnzioWin, it is controlled in Edit:Advanced Options:Print Preview.

The Print Preview window will show the first page (only) of the job to be printed. It will also show many of the calculated (or specified) printing parameters, such as “*Lines per inch*”, “*Characters per inch*”, “*Lines per page*”, margins, etc. As is usual in a print preview, you can choose to print or to cancel.

You can also change many of the printing parameters. If you increase the *left margin*, for instance, Print Wizard will compress the text horizontally, changing the *Characters per inch*. If the print job contains no formfeeds, and you change the *Lines per page*, Print Wizard will break the page in different places.

Changes are made immediately to the sample shown on the screen. If you click *Print*, the document will be printed with the changes you have indicated.

Changes are temporary; they will not be stored in any way.

The Print Preview window can be resized or zoomed to give you a better look at the page.

If you are running on a Tablet PC, or on a version of Windows with Tablet features installed, you can annotate the preview image by drawing on it with the Tablet’s stylus or the mouse. This drawing will be printed on the first page (only) of your printout.

13 Print Wizard's Markup Language

As stated earlier, Print Wizard's markup language (PWML) is based on HTML. A book about HTML may be useful to those unfamiliar with this coding concept. This document assumes some basic familiarity with HTML.

Because of Print Wizard's focus on legacy data, it is not yet fully compatible with generic HTML. Many capabilities of HTML have not yet been implemented, including freeform text, tables, and frames. Cascading Style Sheets (CSS) are not recognized.

13.1 Recognizing a Marked-up Document

Print Wizard detects marked-up data as a file starting with any one of (case insensitive):

```
<!DOCTYPE
<HTML
<PWML
```

Any white space (tabs, spaces, returns, or linefeeds) at the beginning of the file will be ignored when looking for these indicators.

13.2 Tag Basics

Print Wizard's tag structure, like HTML, consists of tags between less-than and greater-than, such as:

```
<PRE>
```

If you are in LEGACY mode (after a <LEGACY> tag), the less-than must be preceded with a "trigger" character, normally control-Z.

Most tags have additional parameters, such as:

```
<FONT face="Courier New">
```

If the parameter value contains embedded spaces, the parameter must be in quotes as shown. If there are NO embedded spaces, the quotes are optional.

The parameter value for some parameters must be numeric, implying there is a need for units of measure. These are described later.

Tags are case insensitive, and can be continued from one line to another.

13.3 End Tags

As with HTML, Print Wizard recognizes certain *end-tags*. So for instance, will start **bold** printing, and will end it. Similar tags that have an on/off behavior are <I> *italics* </I> and <U> underlined </U>.

Other end-tags cause a return to settings that were in effect before the start-tag. For instance, if you are in a certain font such as Arial, then do a FONT tag specifying a FACE of "Courier New", then do , you will return to printing in Arial. Tags that behave in this way include FONT, P, DIV, SUP, and SUB.

13.4 The Bare Minimum

Print Wizard will not actually start printing anything until it gets into the plain text part of the file. That is, it must encounter a <PRE>, <XMP>, <PLAINTEXT>, <LEGACY>, or <LISTING> tag. Since any of these should be inside the <BODY> section, a minimal set of tags (that does not alter Print Wizard's default behavior) is:

```
<HTML><BODY><PLAINTEXT>
```

The differences between the five tags above involve a) what kind of tags are allowed after each, b) assumed width, c) whether ampersand character entities (explained below) are allowed, and d) whether a trigger character is needed.

- After the PLAINTEXT tag, no other tag or character entity processing or is done.
- LISTING implies 132 characters per line, and XMP implies 80 characters per line. No character entities are processed. The only tag that is processed is the ending tag, “/LISTING” and “/XMP” respectively.
- After PRE, any tag is recognized, and character entities are processed. There is no assumed number of characters per line, but that can be specified with the WIDTH attribute.
- After LEGACY, any tag or character entity is allowed, but only if preceded by the trigger character, normally control-Z.

➤ For new code, we recommend either PRE or LEGACY.

13.5 Units of Measure

Standard HTML is ambiguous about placement of text, images, etc. For instance, font size is indicated from “1” to “7”, and bitmap widths use units of pixels. Print Wizard, however, is designed to allow precision, and pixel size varies from printer to printer.

For every numeric value entered in a tag, Print Wizard will assume a certain unit of measure. For instance, the tag

```
<FONT pointsize=12>
```

(which doesn't exist in standard HTML) assumes a unit of “point” (1/72 inch). Units can also be relative to the current setting. To make the pointsize 2 points larger, do

```
<FONT pointsize=+2>
```

A minus sign (“-”) also works. Changes can also be specified in percentages:

```
<FONT pointsize=-25%>
```

Finally, many measurements allow you to specify units exactly, with a two-character code immediately following the number, such as:

```
<PAGESIZE x=8.5in>
```

Allowable units are:

pt = points (72 points = 1 inch)

pc = pica (1 pc = 12 pt)

in = inches

mm = millimeters

cm = centimeters

li = lines (based on current linespacing)

px = pixel

For many measurements, we have tried to make the assumed units compatible with standard HTML, which deals in dots or pixels of the screen. When translating this to the printer, we had to make some assumptions. We have tried to maintain a certain amount of compatibility with existing browsers, as well as with earlier versions of Print Wizard.

Print Wizard takes the following approach. Each document starts off with an assumption of a virtual pixel density of 120 dots per inch (DPI). This is similar to the density used by various browsers when they print. Thus a horizontal rule (HR) that has a WIDTH of 300, will be printed as 300 DOTS wide, or 2.5 inches. Similarly an image (IMG) with a WIDTH of 480, will be printed as 480 IMG-DOTs wide, or 4 inches. Metric equivalent is 304.8 dots per centimeter.

If Print Wizard encounters a <PAGESIZE>; tag, which is non-standard HTML, it then assumes a DPI of 720 (or 1828.8 dots per cm), and a density for images (IMG-DPI) of 300 (or 762 dots per cm). Thus bitmaps that are scanned at 300 DPI will print in true size, regardless of the printer's capabilities. Finally, within either the <PAGESIZE> tag or the <BODY> tag, DPI can be set with a DOTSPERINCH attribute, and IMG-DPI can be set with an IMGDOTSPERINCH attribute.

Thus if DPI is 720, then a DOT unit is 1/720 of an inch. Likewise, IMG-DPI determines the size of an IMG-DOT. If a tag places a rectangle at X=360, and DPI is 720, then the rectangle is placed 1/2 inch in from the edge of the page.

➤ You will find it easier to always specify units (such as "in" for inches), and avoid this DOT confusion.

13.6 Positioning

Items that Print Wizard will print, including text, bitmaps, rectangles, etc., can all be placed precisely where you want them. The horizontal position is always relative to the left edge of the page. The vertical position is always relative to the top edge of the page. Positions can be specified in various units (such as inches) as described above.

Many printers have an unprintable area on one or more edges of the page. This means that you will not be able to print something to the very top edge, for instance. However, Print Wizard takes into account the unprintable areas to ensure that all positioning is done relative to the edges, so that printout will be consistent between printers.

13.7 Global Filtering with Regular Expressions

It is possible to specify one or more search-and-replace operations that will be performed on every line of the data. These operations are specified using *regular expressions*, an approach that is widely used in the computer world.

To use this feature, place one or more REPLACE tags at the beginning of the document, *before* the PRE or LEGACY tag. Replacement operations are applied to all lines *following* the PRE or LEGACY tag, including acting on any tags contained there. It is also possible for a REPLACE action to replace plain text with a tag (with the normal requirement that in LEGACY mode a TRIGGER character must be used).

The syntax is:

```
<REPLACE IN="search string" OUT="replace string">
```

This will search every line for the string listed after IN, and replace it with the string listed after OUT. The general format of the *search string* and *replace string* follows conventions used elsewhere. A book on regular expressions may be useful. But while the concepts are widely used, details and syntax vary, so they are listed here.

13.7.1 The IN String

The characters that can be searched for in the IN string are as follows:

<u>Entry</u>	<u>Meaning</u>
.	(period) Any character except Newline
*	Matches previous character 0 or more times
+	Matches previous character 1 or more times
?	Matches previous character 0 or 1 times
^	Beginning of line
\$	End of line
[abc]	Matches a or b or c
[a-c]	Match a to c
[^a-c]	Matches NOT a to c
\s	space
\t	tab
\b	backspace
\r	return
\l	linefeed
\n	newline (return + linefeed)
\p	(pipe symbol)
\w	word delimiter ; matches \t\s!"&()*+,-./:;<=>?@[\\]^_{}~
\xHH	character represented by hex value HH
(...)	Change sequence of operation, or concatenation
{...}	Include contents as 1 replacement field
	OR

13.7.2 The OUT String

The characters that can be in the OUT string are:

<u>Entry</u>	<u>Meaning</u>
--------------	----------------

\z	Replace with nothing (delete)
\s	space
\t	tab
\b	backspace
\r	return
\l	linefeed
\n	newline (return + linefeed)
\1 to \9	Tagged fields found by search and enclosed in {}
&	Entire found text

13.8 Default Margins and Alignment

Print Wizard by default will begin printing text as far up and to the left as possible (if *nice margins* are not turned on). Stated another way, the initial margins are the same as the edges of the printable area (described above). So when you print with Print Wizard, text should align with the same place it would if you sent that text directly to the printer, with no printer driver involved.

Also, Print Wizard's auto-fit logic will, if possible, print the data at 10 characters per inch on the horizontal, and 6 lines per inch on the vertical. Again, this corresponds to common printing standards.

Taken together, this means that if you move from printing certain text directly from DOS or Unix, to printing it with Print Wizard, the alignment should generally be the same. If the printout fit nicely onto a preprinted form before, on the same printer, it should continue to do so.

On the other hand, if *nice margins* is ON, Print Wizard will attempt to provide balanced margins, up to ½ inch.

13.9 Setting Page Dimensions and Margins

The <PAGESIZE> and <BODY> tags, taken together, allow you to specify many aspects of the paper selection and page layout, including paper size, orientation, input bin, orientation, duplexing, and margins. Some of these settings will be dependent on your printer's capabilities.

Paper size can be specified as one of certain standard sizes (identified by name or number), by printer-specific size names or numbers, or by actual dimensions ("X = 8.5in Y=11in"). Note that in Windows NT only, there must be a "form" defined to fit arbitrary dimensions. However, on all other versions of Windows, arbitrary sizes can be specified freely.

Orientation can be specified as "portrait", "landscape", or "auto". In "auto" orientation, Print Wizard will print in portrait, unless the width (in columns) or computed character pitch crosses a certain threshold, as described in "Auto-rotation Logic", page 10.

If the printer can print both sides of the paper, that can be controlled with the "DUPLEX" parameter. Note that "VERTICAL" and "HORIZONTAL" refer to binding along the long and the short edge of the paper, respectively, without regard to the printing orientation. However, "LEFT" and "TOP" do take into account the orientation.

The printing margins can be specified in the <BODY> tag. Note that margins are specified relative to the edge of the paper.



-
- Note also that the “BOTTOMMARGIN” is specified relative to the TOP of the paper, and the “RIGHTMARGIN” is specified relative to the LEFT edge of the paper.
-

The PAGESIZE tag can also specify what BIN the printer will pull paper from (by name or number), and the number of COPIES to be printed.

The PAGESIZE tag can also specify the MEDIA; that is, the kind of paper (by name or number). On printers that support this feature, the printer will prompt on its front panel for an operator to insert the specified kind of paper, before the printer will print the job. Or, the printer may decide which bin to draw paper from, based on what it has been told about what kind of paper is in what bin.

13.9.1 Paper Names, Bin Names, etc.

When you specify things such as PAPER and BIN, you have several options. You can:

- 1) Use a Windows-standard number, as listed in the reference section.
- 2) Use a Windows-standard name, as listed in the reference section.
- 3) Use a printer-specific number.
- 4) Use a printer-specific name.

Printer-specific names and numbers are defined by the printer drivers. Once you find out what names/numbers are available, you can use those for a particular printer.

-
- Using printer-specific settings makes your PWML files no longer device independent.
-

To find out what options are available for a particular printer, run Print Wizard with DEBUG and Printer Setup (“/s”). When you are presented with the Printer Setup box, go to the appropriate area, such as “Input bin”. Note the names available – these are what you can use in PWML. Now select the one you want, and print the job. In the DEBUG output, you will find a line that says, for instance, “Paper bin code = 257”. This is the numeric equivalent.

13.9.2 Changing Pagesize in the Middle of a Job

It is possible to issue a new PAGESIZE tag after printing has started. In this case, the parameters specified will take place at the beginning of the next page. You might, for instance, specify an initial PAGESIZE with a BIN of “1”, start into the first page, and then issue another PAGESIZE tag specifying a BIN of “2”. This would cause the second page (when it occurs) and all subsequent pages to print from bin 2 of the printer.

Parameters that can be changed in this manner include PAPER, X, Y, COPIES, BIN, MEDIA, ORIENTATION, OVERLAY, and USEOVERLAY.

13.9.3 Usage Patterns in Pagesize Tags

For certain parameters in a PAGESIZE tag, you can specify a *pattern* instead of a single value. For instance, if you specify

```
<PAGESIZE BIN=1, 2>
```

then Print Wizard will print the first page from bin 1, the second page from bin 2, then go back to bin 1. That is, by default, the pattern will *cycle*. Or, you could enter

```
<PAGESIZE BIN=1, 2, continue>
```

In this case, the reserved word “continue” means to stay on the last value, and NOT to cycle. This would print the first page from bin 1 and all later pages from bin 2.

Parameters that can be set in this way include PAPER, BIN, MEDIA, COPIES, and USEOVERLAY.

13.10 Multi-column Printing

Print Wizard has the ability to convert plain text input into multiple columns, such as for printing on label stock. That means you can adapt a print job that was designed to print on one-across labels on a dot matrix printer into multi-column labels on a laser printer. This is done by specifying all the relevant dimensions of the label.

Specify the TOPMARGIN and BOTTOMMARGIN so as to avoid printing outside any labels. If necessary, specify the LEFTMARGIN so as to position the beginning horizontal print position of the left-most label. You may want to specify the RIGHTMARGIN, just to make sure Print Wizard doesn't try to print too many columns.

The most critical measurement, and the one that switches on the multi-column feature, is COLUMNOFFSET. Specify the COLUMNOFFSET as the horizontal distance from one label to the next (left edge to left edge). To avoid printing in the gap between labels (that is, between the columns), you can specify COLUMNGUTTER; this is subtracted from the COLUMNOFFSET to establish the right edge of the printable space on a label.

You will also need to ensure that the print data coming to Print Wizard is properly formatted in the vertical dimension. The number of lines per label, combined with Print Wizard's LINESPACING (computed or specified), must cause printing to advance to the correct point at the top of each label. Most often, you will want a LINESPACING of 1/6 inch (which is the default), so if your label stock is 1 inch high (top-of-label to top-of-label) your print data must contain 6 lines per label.

Print Wizard's logic then works like this. It starts printing at the TOPMARGIN, working its way down the page, until it reaches the BOTTOMMARGIN. Then, when it sees that a COLUMNOFFSET is in place, it will add the COLUMNOFFSET to the working left margin, reset to the TOPMARGIN, and begin printing again. When it can no longer add COLUMNOFFSET to the working left margin and still stay within the RIGHTMARGIN, it will advance to the next page.

Following is the beginning of a multicolumn label job:

```
<pwml><pagesize paper=letter orientation=portrait linespacing=12pt>
<body topmargin=.5in bottommargin=10.5in
leftmargin=.5in rightmargin=8in columnoffset=4in><legacy>
Name 1
Address 1a
Address 1b
City, state, zip
USA

Name 2
...
```

Often, the dimensions of the job will be placed in a print initialization file. Included with Print Wizard are many sample initialization files. The file "labels.init" is a generic sample. The files whose names start with "Gaylord" are designed for stock labels from Gaylord, a library supply company.

13.11 Colors

Print Wizard recognizes the following colors for fonts, font backgrounds, and horizontal rules:

#rrggbb	hex RGB value
GRAYxx	gray scale from GRAY00 (black) to GRAY99 (white)
GREYxx	likewise
aqua	
black	

blue
fuchsia
green
lime
maroon
navy
olive
purple
red
silver
teal
yellow
cyan
magenta
white

13.12 Fonts

Print Wizard is generally designed to work with mono-spaced fonts, such as Courier New. Mono-spaced fonts are consistent with the focus on legacy print data, which generally relies on character spacing to align columns. Also, Courier New can contain many international characters, if those options have been installed on your Windows system.

Print Wizard also works best with scalable fonts, i.e., TrueType fonts. These fonts can be scaled precisely to get the necessary height and width.

It is possible, with the FONT tag, to specify ANY available font, but your results may not be acceptable. If you are using a dot matrix printer, you may want to specify a font that is built in to the printer, for faster printing. These fonts can be identified by doing a font selection in a Windows program, and looking for a printer icon beside the font. Again, though, if they're not scalable then that limits Print Wizard's flexibility.

If you specify a font that is built into the printer, such as "Courier 10cpi" in an Okidata printer, and then Print Wizard determines that it needs to print at a tighter pitch (more character per inch), Print Wizard will attempt to find a similarly named font that will work, such as "Courier 17cpi". The *debug* output will show you that this has happened.

Text can also be printed in any rotation, with where "*n*" is in degrees, counterclockwise from horizontal. When ROTATION is not zero, Print Wizard does not advance the printing position ("cursor"), so you will always need to specify location afterward.

Some printer drivers may not handle international characters well. If you have trouble, open the printer's properties in Windows' control panel, and try different settings for font substitution and type of download. You might also try a different but similar printer driver, such as a LaserJet 4 driver for a LaserJet 5 printer.

13.13 Automatic Font Selection

Print Wizard has sophisticated logic to ensure that the characters you want to print will in fact get printed. Part of this logic includes testing the font to make sure that it contains the character. If it does not, Print Wizard will attempt to find and use one that does.

Print Wizard can be told to print non-ASCII text by a) using an HTML ampersand character entity (see below), or b) feeding it characters whose numeric value is above hex 80 (decimal 128). In the latter case, with the freestanding Print Wizard program, those characters are assumed to be in the OEM (DOS) character set, unless a command-line parameter specifies them to be in the ISO (Windows) set ("/i") or in Unicode UTF-8 ("/u"). Within AnzioWin, the character set of passthrough print data can be specified in the Communicate:Character set menu item.



Print Wizard works with a "font list", a list of font names separated by commas. The first font in the list is the primary font; it is used if possible. The second and following fonts are alternate fonts. If a character is not in the primary font, Print Wizard will search through the second and following fonts until it finds one that contains the character. If the last font in the font list is "auto", Print Wizard will check Windows for all installed fonts (generally with a preference for mono-spaced fonts) to find one with the necessary character.

On startup, Print Wizard has a font list of "Courier New, auto". That is, Courier New will be used wherever possible; when it doesn't contain a character Print Wizard will automatically search for an appropriate font.

You can specify a different font list in two ways: a) through the command line that invokes Print Wizard, and b) using Print Wizard Markup Language (PWML).

For the command line, use a parameter:

```
/vFont="name1, name2, name3"
```

You can list as many names as you like. Names can contain embedded spaces. Names must be separated by commas. Use a name "auto" as the LAST item in the list. For instance:

```
printwiz /vFont="Courier new, ms song, auto" somefile.txt
```

Within PWML (or HTML), use the syntax

```
<font face="name1, name2, name3">
```

You can also specify one or more fonts that should NOT be used during "auto" font selection, by including those font names with a minus sign in front. For instance:

```
<font face="Courier new, Mingliu, -badfont, auto">
```

Note that if you specify the font list by either means, it will override the default startup values, including secondary font(s), including "auto".

13.14 Font Size, Pitch, and Linespacing

LINESPACING is the vertical distance from one line of text to another. Your PWML code can specify LINESPACING directly. If you do not specify LINESPACING, but you specify a FONT size (or pointsize), LINESPACING will be the same as the FONT size. If you specify neither, Print Wizard calculates the LINESPACING necessary to fit the required lines into the page, and then bases the font size on that.

The standard linespacing, at least in the US, is 6 lines per inch. Because there are 72 points in an inch, a font size of "12pt" (12 point) would yield 6 lines per inch.

If the font size is changed within the body of a print job, with a tag, the linespacing will be changed accordingly, for all printing from then on.

➤ LINESPACING affects every *linefeed*; that is the end of every line in your input file. We recommend changing FONT (and therefore linespacing) *before* an end-of-line.

In a similar manner, the spacing in the horizontal direction is related to the width of the characters. Both are controlled by the PITCH parameter within the FONT tag. It is quite common to print at 10 character per inch; that is, one character is 1/10 inch wide. This would be specified:

```
<font pitch=.1in>
```

For 132-column reports in 8 inches, it is common to print at 17 CPI. Because $1/17 = .0588$, this would be specified:

```
<font pitch=.0588in>
```

Again, these settings will affect both the spacing between characters and the printed widths of the characters themselves (at least for TrueType fonts).

-
- For variable-pitch fonts, such as Arial or Times New Roman, specify a PITCH of 0 (zero).
-

13.15 Character Entities

In <PRE> mode, Print Wizard recognizes HTML standard named values for character entities, such as “<” for less-than (“<”), “&” for ampersand (“&”), and also “€” for the Euro character (“€” -- might not show on your PC).

It also recognizes characters coded numerically, such as “ò” and “&#D242” (both decimal), and “&#H20AC” and “€” (both hex). Numbers below 128 (decimal) resolve to ASCII values. Numbers between 128 and 255 are in the ISO 8859-1 character set, which is essentially the same as Windows Latin-1. As in true HTML, numbers above 255 resolve to 16-bit Unicode characters.

-
- Character specified in this way are **not** affected by the character-set setting (/I, /O) of Print Wizard.
 - To look up Unicode values, you can browse through the tables at <http://www.unicode.org>. Or, on later versions of Windows, you can use the Charnap utility. Do Start:Run:Charnap. Set the font to Courier New, for instance. Now browse through the available characters. As you move your mouse over a character, a hint will appear showing something like “U+221E: Infinity” for the infinity symbol. The “221E” is the hex Unicode value.
-

Following is a complete list of the named character entities that Print Wizard recognizes. Note that names are case sensitive. Some characters may not show below, due to font limitations.

quot	"	plum	±	Egrave	È	szlig	ß
amp	&	sup2	²	Eacute	É	agrave	à
lt	<	sup3	³	Ecirc	Ê	aacute	á
gt	>	acute	´	Euml	Ë	acirc	â
euro, Euro	€	micro	µ	Igrave	Ì	atilde	ã
nbsp	non-break	para	¶	iacute	Í	auml	ä
	space	middot	·	Icirc	Î	aring	å
ixcl	¡	cedil	¸	Iuml	Ï	aelig	æ
cent	¢	sup1	¹	ETH	Ð	ccedil	ç
pound	£	ordm	º	Ntilde	Ñ	egrave	è
curren	¤	raquo	»	Ograve	Ò	eacute	é
yen	¥	frac14	¼	Oacute	Ó	ecirc	ê
brvbar	¦	frac12	½	Ocirc	Ô	euml	ë
sect	§	frac34	¾	Otilde	Õ	igrave	ì
uml	¨	iquest	¿	Ouml	Ö	iacute	í
copy	©	Agrave	À	times	×	acirc	î
ordf	ª	Aacute	Á	Oslash	Ø	iuml	ï
laquo	«	Acirc	Â	Ugrave	Ù	eth	ð
not	¬	Atilde	Ã	Uacute	Ú	ntilde	ñ
shy	-	Auml	Ä	Ucirc	Û	ograve	ò
reg	®	Aring	Å	Uuml	Ü	oacute	ó
macr	¯	Aelig	Æ	Yacute	Ý	ocirc	ô
deg	°	Ccedil	Ç	THORN	Þ	otilde	õ

ouml	ö	nu	ν	empty	∅	rfloor	⌋
divide	÷	xi	ξ	nabla	∇	lang	□
oslash	ø	omicron	ο	isin	∈	rang	□
ugrave	ù	pi	π	notin	∉	loz	◇
uacute	ú	rho	ρ	ni	∋	spades	♠
ucirc	û	sigmaf	ς	prod	∏	clubs	♣
uuml	ü	sigma	σ	sum	∑	hearts	♥
yacute	ý	tau	τ	minus	-	diams	♦
thorn	þ	upsilon	υ	lowast	*	quot	"
yuml	ÿ	phi	φ	radic	√	amp	&
fnof	ƒ	chi	χ	prop	∝	lt	<
Alpha	A	psi	ψ	infin	∞	gt	>
Beta	B	omega	ω	ang	∠	OElig	Œ
Gamma	Γ	thetasym	ϑ	and	∧	oelig	œ
Delta	Δ	upsih	Υ	or	∨	Scaron	Š
Epsilon	E	piv	ϖ	cap	∩	scaron	š
Zeta	Z	bull	•	cup	∪	Yuml	ÿ
Eta	H	hellip	...	int	∫	circ	ˆ
Theta	Θ	prime	'	there4	∴	tilde	˜
Iota	I	Prime	"	sim	≈	ensp	en space
Kappa	K	oline	-	cong	≅	emsp	em space
Lambda	Λ	frasl	/	asymp	≈	thinsp	thin space
Mu	M	weierp	℘	ne	≠	zwnj	zero-width non-joiner
Nu	N	image	ℑ	equiv	≡	zwj	zero-width joiner
Xi	Ξ	real	ℜ	le	≤	lrm	
Omicron	Ο	trade	™	ge	≥	rlm	
Pi	Π	alefsym	ℵ	sub	⊂	ndash	-
Pi	Π	larr	←	sup	⊃	mdash	—
Rho	Ρ	uarr	↑	nsup	⊄	lsquo	‘
Sigma	Σ	rarr	→	sube	⊆	rsquo	’
Tau	Τ	darr	↓	supe	⊇	sbquo	‚
Upsilon	Υ	harr	↔	otimes	⊗	ldquo	“
Phi	Φ	crarr	↵	perp	⊥	rdquo	”
Chi	Χ	lArr	⇐	sdot	⋅	bdquo	„
Psi	Ψ	uArr	⇑	lceil	⌈	dagger	†
Omega	Ω	rArr	⇒	rceil	⌋	Dagger	‡
alpha	α	dArr	⇓	lfloor	⌊	permil	‰
beta	β	hArr	⇔			lsaquo	‹
gamma	γ	forall	∀			rsaquo	›
delta	δ	part	∂				
epsilon	ε	exist	∃				
zeta	ζ						
eta	η						
theta	θ						
iota	ι						
kappa	κ						
lambda	λ						
mu	μ						

13.16 Variable Width Spaces

In certain situations, especially when printing in a variable-spaced font, it is convenient to be able to use spaces of different widths. Print Wizard recognizes several space characters as standardized in Unicode. These can be specified by their numeric values, as described above, or as encoded in UTF-8 in the print file. Note that many sizes are based on an "em" unit, which is the same width as the current point size (height).

Character value (hex) Width



0020	standard
2000	½ em height
2001	em height
2002	½ em height
2003	em height
2004	1/3 em
2005	1/4 em
2006	1/6 em
2007	same width as zero
2008	same width as decimal point
2009	1/5 em
200A	1/12 em
200B	zero width space

13.17 Line-drawing Characters

Line-drawing characters are commonly used in DOS and Unix programs to create boxes. Print Wizard can be told to print these in two ways: a) as 8-bit characters in an OEM encoding, or b) by their Unicode values (in the hex 2500 range).

Print Wizard does NOT rely on fonts to print these characters, because fonts are inconsistent in placing and connecting the lines. Instead, Print Wizard will draw the lines using internal bitmaps. Thus they should always print properly.

13.18 Rectangles

HTML has a feature called “horizontal rule”, as <HR>. By specifying SIZE (height), WIDTH, COLOR, and OUTLINE, you can create a solid, hollow, or color-filler rectangle of any size. The OUTLINE is always black.

Print Wizard extends this with absolute positioning (X and Y), so you can place the HR any place on the page. With clever use of these, you can create grids and other forms.

Placing an HR will change the current printing position, so you may want to use a GOTO tag afterward to begin printing text in a particular spot.

It is possible to print rectangles over each other, text over rectangles, etc.

13.19 Lines

You can also tell Print Wizard to draw arbitrary lines, from one x-y location to another (including angles). The WIDTH and COLOR parameters will dictate how the line is drawn. Defaults are one pixel wide, in black.

The endpoints of the line are identified as X and Y for one end, and X2 and Y2 for the other end. The default for both ends is the current printing position; if you don't specify *any* you will not get a visible line.

13.20 Graphics

Print Wizard is able to print bitmaps (pictures) from files that are in BMP, GIF, or JPG/JPEG format, as well as Windows metafiles (WMF) or enhanced metafiles (EMF). Files can be on the PC, a local area network, or a web or FTP server. Pictures are coded using the tag. These can be sized with the HEIGHT and WIDTH attributes. A BORDER can be added (always in black).

If the filename (in the SRC parameter) does not contain a complete path, Print Wizard will look for it in a particular directory, generally the directory containing the data file. You can override this by using the BASE tag with the HREF parameter. You can also specify an assumed file extension, with the EXT parameter of the BASE tag. So, if early in the print job you specify

```
<base href=c:\mypix ext=".bmp">
```

then you can generate a picture with the tag

```
<img src=wally>
```

and Print Wizard will read the file "c:\mypix\wally.bmp".

With Print Wizard's additional attributes (X and Y), graphics can be placed anywhere on the page.

Printing a bitmap will change the current printing position, so you may want to use a GOTO tag afterward to begin printing text in a particular spot.

It is possible to print text over graphics. However, graphics over text will likely hide the text.

13.21 Form Overlays

The general concept of form overlays is described elsewhere. To specify within your PWML document that an overlay should be used, include an OVERLAY attribute in the <PAGESIZE> tag. To specify an overlay that will be used for Print Preview but will not be printed, use PREVIEWOVERLAY instead.

The overlay does NOT affect the position at which Print Wizard will begin printing text on the page.

It is also possible to specify an overlay using the "/OVERLAY" command line parameter for Print Wizard.

Note that Print Wizard relies on the file extension to know how to process the file.

13.21.1 Storing and Using PCL Overlays in PCL 5 Printers

If you are using a PCL 5 printer (with macro capabilities), Print Wizard can load a PCL overlay file into the printer's memory and leave it there. It can also invoke (activate) overlay macros that are already stored there, either by Print Wizard or by some other program. More to come on this.

13.22 Barcodes

Print Wizard can internally generate a variety of barcodes, which can then be printed on nearly any printer, with a tag as simple as

```
<barcode src=1234>
```

Of course there are many kinds of barcode, and you'll want to specify style, size, position, etc. The Reference section below lists all the parameters. But here are some special notes.

The kind of barcode is set with the STYLE parameter; default is CODE128.

The SIZE of the barcode is its height. If not specified, it will default to the current font height.

Print Wizard starts off assuming that BARWIDTH, the width of the narrowest bar or space, is .012 inches. However, this width will be rounded up, based on your printer's graphics resolution, so the actual width of the printed barcode will vary some between printers. It is advisable to set your printer's properties (through Windows) at its maximum resolution. On a low resolution printer, the default BARWIDTH may produce unreadable barcodes. Therefore, you may need to specify a larger value.

Some barcode styles have particular requirements, such as numeric data only or a particular length. If you don't meet these requirements, Print Wizard may detect that and print a message on the page, in place of the barcode.

Some barcode styles can have a check character, which is used by the barcode reader to mathematically validate the barcode. You can tell Print Wizard to calculate and print this character with the ADDCHECKCHAR parameter.

Barcodes usually have one line of human-readable text also. This can be turned on or off with the CAPTION parameter. The font used for the caption can be specified with the FACE parameter; default is the same face being used for text.

Certain styles of barcode have optional features, such as bearer bars or tall guard bars. These can be turned on or off.

Bar codes can be rotated to 90, 180, or 270 degrees, with the ROTATION parameter.

Barcodes are printed in line with the text, unless placement is specified with X and/or Y. The horizontal (if not rotated) size of the barcode is calculated automatically from the BARWIDTH, the STYLE, and all the necessary elements of the barcode. The current printing position will advance based on the total width. However, if the barcode is rotated, final printing position is undetermined.

-
- Print Wizard does not enforce the need for white space around a barcode. If you have other printing too close, the barcode will not be scanned reliably. Be sure to test your results.
-

Once the format of the barcode (STYLE, SIZE, etc.) has been established, it will remain in effect for the duration of the print job. It is sometimes easier to establish this format once, at the beginning of the print job, by including a BARCODE tag that has no SRC parameter (and will therefore not print anything).

Print Wizard still retains an earlier approach to printing barcodes, using special barcode font files. If you need to use these, please contact Rasmussen Software.

13.23 PWML Reference

The following table indicates all the tags and parameters that Print Wizard recognizes. Items marked with "+" are extensions to the HTML standard. Where units of measure are involved, the table shows what unit is assumed; see above for explanation. The table also indicates when relative or percentage changes are allowed, and where exact units can be specified.

<HTML> . . . </HTML>

<PWML> . . . </PWML>

<HEAD> . . . </HEAD>
(ignored)

<META>
CHARSET=UTF-8

<PAGESIZE> +
DOTSPERINCH
establishes DPI and DOT

IMGDOTSPERINCH
establishes IMG-DPI and IMG-DOT

X=n
in DOTS
relative, %, units

Y=n
in DOTS
relative, %, units

PAPER=n
1 or "letter"
2 or "lettersmall"
3 or "tabloid"
4 or "ledger"
5 or "legal"
6 or "statement"
7 or "executive"
8 or "A3"
9 or "A4"
10 or "A4small"
11 or "A5"
12 or "B4"
13 or "B5"
14 or "folio"
15 or "quarto"
20=#10 envelope
or other Windows standard sized, by number
or printer-specific paper name
<pattern>

BIN=n
1=upper
2=lower
3=middle
4>manual
5=envelope
6=envelope manual
7=auto
8=tractor feed
<pattern>

MEDIA=n
1 or "Standard"
2 or "Transparency"
3 or "Glossy"
or printer-specific media name
<pattern>

COPIES=n
number
<pattern>

ORIENTATION=s
1 or "landscape"
0 or "portrait"
2 or "auto"

ROTATEWIDTH=n
Number of characters

ROTATEPITCH=n
DOTS
Relative, %, units

LENGTH=n

```

    lines per page
LINESPACING=n
    vertical, DOTS
    relative, %, units
OVERLAY=filename
    always printed at full page size
    currently supports BMP, JPEG, JPG, GIF, WMF, EMF, PWML, and PCL formats
PRELOAD=n
    Stores overlay into PCL printer
PREVIEWOVERLAY=filename
    displayed in preview window, but not printed
USEOVERLAY=n
    number
    pattern
    use PCL macro(s) already in printer
DUPLEX=xxx
    VERTICAL
        Binding on long edge
    HORIZONTAL
        Binding on short edge
    LEFT
        Binding on left edge, respecting orientation
    TOP
        Binding on top edge, respecting orientation
    OFF

<BASE>
    HREF=s
        base location for images
    EXT=s +
        assumed extension for IMG files

<BODY> ... </BODY>
    DOTSPERINCH +
        establishes DPI and DOT
    IMGDOTSPERINCH +
        establishes IMG-DPI and IMG-DOT
    TOPMARGIN=n
        in DOTS
        relative, %, units
    BOTTOMMARGIN=n
        in DOTS
        relative, %, units
        measured from top edge of paper
    LEFTMARGIN=n
        in DOTS
        relative, %, units
    GUTTER=n
        in DOTS
        relative, %, units
        adds space on left side (or top, if DUPLEX setting indicates binding
        there) of odd-numbered pages, right side (or bottom) of even-numbered
    RIGHTMARGIN=n
        in DOTS
        relative, %, units
        measured from left edge of paper
    COLUMNOFFSET=n +

```

in DOTs
units
COLUMNGUTTER=n +
In DOTs
units

<LISTING> ... </LISTING>
implies 132 characters per line
no embedded tags
no character entities

<XMP> ... </XMP>
implies 80 characters per line
no embedded tags
no character entities

<PLAINTEXT>
no tags can follow, not even </PLAINTEXT>
no character entities

<PRE> ... </PRE>
can contain tags
can contain character entities
WIDTH=n
characters per line

<LEGACY> ... </LEGACY>
can contain tags prefixed by trigger
can contain character entities prefixed by trigger
TRIGGER=n
describes the trigger character in decimal
default trigger is control-Z

<P> ... </P>
paragraph break
ALIGN=s
LEFT
CENTER
RIGHT
JUSTIFY
LEFTMARGIN=n
in DOTs
RIGHTMARGIN=n
in DOTs

<DIV> ... </DIV>
division break, treated as paragraph break
ALIGN=s
LEFT
CENTER
RIGHT
JUSTIFY
LEFTMARGIN=n
in DOTs
RIGHTMARGIN=n
in DOTs

```
<SMALL> ... </SMALL>
    makes text 20% smaller
<BIG> ... </BIG>
    makes text 25% larger
<H1> ... </H1> through <H6> ... </H6>
    headers
    sets text size to 24pt, 18pt, 14pt, 12pt, 10pt, 7pt respectively

<FONT> ... </FONT>
    FACE=s
        Windows font name
    FACE="face1, face2, face3"
        Windows font names for primary, additional fonts
        use "-face4" to prohibit use of face4
        end with "auto" for continued auto-selection
    SIZE=n
        1 through 7, arbitrarily assign to point sizes
        relative, %, units
        affects linespacing
    POINTSIZE=n +
        exact height in points
        relative, %, units
        affects linespacing
    PITCH=n +
        in DOTS
        relative, %, units
        forces horizontal spacing of characters
    LINESPACING=n +
        in DOTS
        relative, %, units
    ROTATION=n +
        Any value, in degrees
    COLOR=xxx +
        Colors listed elsewhere
    BACKGROUND=xxx +
        Colors listed elsewhere

<SUP> ... </SUP>
    Superscript

<SUB> ... </SUB>
    Subscript

<GOTO> +
    X=n
        in DOTS
        relative, %, units
        accounts for unprintable area
        specifies left side of character cell
    Y=n
        in DOTS
        relative, %, units
        accounts for unprintable area
        specifies top of character cell by default (see ALIGN)
    ALIGN=xxx
        TOP
```

The Y argument applies to the top of the character cell

BASE
The Y argument applies to the baseline of the character

BOTTOM
The Y argument applies to the bottom of the character cell

SAVE=X
save position for later RESTORE

SAVE=Y
save position for later RESTORE

RESTORE=X
RESTORE=Y

 ...
bold on/off

<U> ... </U>
underline on/off

<I> ... </I>
italics on/off

Break line

LINE +
X=n
in DOTS
relative, %, units
default is current position

Y=n
in DOTS
relative, %, units
default is current position

X2=n
in DOTS
relative, %, units

Y2=n
in DOTS
relative, %, units

COLOR=s
color names described elsewhere

WIDTH=
in DOTS
relative, %, units
default is 1 pixel

<HR>
horizontal rule or rectangle

SIZE=n
height in DOTS
relative, %, units

WIDTH=n
in DOTS
relative, %, units

COLOR=xxx
Colors listed elsewhere

OUTLINE=n
width of outline, in DOTS
relative, %, units
always black

X=n +

in DOTS
 relative, %, units
 accounts for unprintable area
 defaults to centered on page

Y=n +
 in DOTS
 relative, %, units
 accounts for unprintable area
 defaults to next line

SRC=s
 file name, with forward slashes
 can be URL
 affected by BASE .. HREF, EXT
 can be "SCAN://"

currently supports BMP, JPEG, JPG, GIF, WMF, and EMF formats

ALIGN=s
 top
 top of image to top of text
 middle
 middle of image to baseline of text
 bottom
 bottom of image to baseline of text
 default

BORDER=n
 DOTS
 relative, %, units

HEIGHT=n
 in IMG-DOTS
 defaults to bitmap's height

WIDTH=n
 in IMG-DOTS
 defaults to bitmap's width

HSPACE=n
 horizontal space around image
 in DOTS
 relative, %, units

X=n +
 in DOTS
 relative, %, units
 accounts for unprintable area
 defaults to right after last text

Y=n +
 in DOTS
 relative, %, units
 accounts for unprintable area
 defaults to next line

<BARCODE> +

STYLE=s
 UPCA
 UPCE
 EAN13
 EAN8
 2OF5
 CODABAR

CODE11
CODE39
CODE93
CODE128
CODE128B
CODE128C
POSTNET
PDF417
MAXICODE

FACE=s
Windows font name for caption text
does not affect normal text

POINTSIZE=n
in points
relative, %, units

SIZE=n
in points
relative, %, units

X=n
DOTs
relative, %, units
specifies left edge

Y=n
DOTs
relative, %, units
specifies top edge

SRC=s
String to be barcoded

SUPPLEMENT=s
String for barcode supplement

BARWIDTH=n
inches
relative, %, units
defaults to .012in

WIDTHRATIO=n
wide-to-narrow ratio
no units
default is 2

BEARERBARS=b
OFF or ON

TALLGUARDBARS=b
OFF or ON

CAPTION=b
OFF or ON

SHOWGUARDCHARS=b
OFF or ON

ADDCHECKCHAR=b
OFF or ON

ROTATION=n
0, 90, 180, 270

REPLACE +
IN=s
regular expression
OUT=s
regular expression



14 The Samples Directory

The Print Wizard product includes several sample files, to demonstrate the product's capabilities. These are generally installed in a SAMPLES or PRINTWIZ-SAMPLES directory, under the main install directory. You can examine the contents of each file by opening it in NOTEPAD, for instance, or any other text editor.

There is a program included called "Printwiz Sampler", accessible from the Windows Start menu, which makes it easy to print and view these sample files.

14.1 Files to be Printed

The files in the SAMPLES directory that have a ".TXT" extension are to be printed. You can also print each sample, with Print Wizard, to see the results. For instance, if you open a DOS (command) Window, then CD to the Print Wizard directory, you could type in:

```
printwiz samples\char-chart.txt<enter>
```

Following is a description of each file, and notes on printing it.

features.txt	A 2-page document demonstrating an overview of Print Wizard's features.
manpage.txt	A capture of a Unix "man" page, that includes backspace-bolding and backspace-underlining, which Print Wizard converts to true bolding and underlining.
80-col.txt	A plain text 80-column report.
plain.txt	An HTML report demonstrating <plaintext> mode.
pre.txt	An HTML report demonstrating <pre> mode.
132-col.txt	A plain text 132-column report. Try printing this with option "/oP", then with "/oL", then with "/oA".
135-co.txt	A plain text 135-column report.
135-col-landscape.txt	A PWML report with 135 columns, forced to landscape orientation.
fonts.txt	Demonstrates font size and faces.
rfc1416.txt	Demonstrates a plain text documentation file.
fontcolr.txt	Demonstrates many foreground and background colors and grays (prints as grays on a monochrome printer).
barcode.txt	Various kinds of barcodes. Does NOT require special printer features.
barcoder.txt	Various rotated barcodes.
mult-clm.txt	Demonstrates multiple-column address labels.

rx-label.txt	Pharmacy labels, demonstrating multi-column labels, font size changes, rotated barcodes, and rotated text.
minipage.txt	Demonstrates Print Wizard's mini-page logic. Try printing this on a tractor-feed printer. Note that the page size created is just as long as is needed for the number of lines in the report, at 6 lines per inch.
hudform.txt	Demonstrates text-over-form, where the form is in PCL format.
taxform.txt	Demonstrates text-over-form, where the form is in PCL format.
insform.txt	Text-over form, where the form overlay is a GIF bitmap. Should print on any printer, or fax board.
dos-report.txt	A file generated by a DOS program, with line-drawing characters. Must be printed with character set equals OEM (the default). Line-drawing should all connect properly, regardless of character size.
char-chart.txt	A plain text file containing a character chart, with characters from hex-20 to hex-FF. Try printing as ISO and then as OEM.
utfsampler.txt	A PWML file, preset to UTF-8 encoding, that prints a sampler of Unicode characters. Also demonstrates Print Wizard's automatic font selection. Your results will vary depending on what fonts you have installed on your PC.
usmarc.txt	A PWML demonstrating special treatment of diacritics, while printing character/diacritic combinations used in the library industry.
postnet.txt	A PWML file demonstrating the Postnet barcode in four rotations.

14.2 Other Files

Also in the SAMPLES directory are the following:

test-pw.cbl	A sample COBOL source program that uses the Print Wizard DLL.
letter.init	A sample print-init file, that initializes Print Wizard to use letter-size paper, print with auto orientation, etc.
label.init	A sample print-init file, that initializes Print Wizard to print 2-column labels.
Gaylord*.init	Print-init files for Gaylord label stock.
sampledialog.exe	A program that lets you print the various samples, or other files, with option settings.
testdll.vbp	A sample Visual Basic program that demonstrates calling PWDLL.DLL (also testdll1.frm).

15 Application Notes

15.1 *Print Wizard and filePro*

Print Wizard includes two printer definition files for use with filePro (a database system). They are:

- PRINTWIZ.PRT For use with monochrome printers
- PRINTWZC.PRT For use with color printers

In filePro a PRT file defines a “printer type”. It contains some general information about the printer, some initialization and termination codes that are used in general, and some “printcode entries”. Printcode entries are identified by number. Each printcode entry consists of a control sequence to send to the printer, and a description. A programmer designing a report can include printcodes in the report. At the time the report is printed, the user will by some means have chosen a printer type, which corresponds to a PRT file. The filePro system uses the PRT file to translate each printcode into the control sequence that is appropriate for that printer. This achieves a certain amount of device independence, assuming there is consistency in what function is assigned to what numbered printcode.

The Print Wizard PRT files translate these printcodes to PWML sequences. For instance, printcode 5, labeled “Underline on”, sends out the sequence “<U>”, which Print Wizard understands to mean “underline on”. The printcode numbers used in the Print Wizard PRT files are synchronized where possible with laser printer printcode files delivered as part of the filePro product.

When the user selects Print Wizard as their printer type, filePro will render each report in PWML. Now as long as that print job goes through Print Wizard in its way to the printer (either standalone Print Wizard or the Print Wizard logic built into AnzioWin), it will be printed correctly on virtually any printer that Windows supports.

15.1.1 Installation and Use – Windows

To make the Print Wizard definition files available in Unix/Linux-based filePro, copy one or both PRT files into the “lib” directory of your “fp” directory tree. For instance, into “c:\fp50\fp\lib”.

To use, select “printwiz” or “printwzc” as your printer type in “Printer Maintenance”.

To route the printer output through Print Wizard, use one of these methods”

- Configure filePro to print to disk. Set the PFPOSTPRINT environment variable to execute the appropriate command after closing the file; that command would be to run PRINTWIZ.EXE with the necessary parameters.
- Configure filePro to print to disk, in a particular directory. Have a copy of Print Wizard running in the background, in despooler mode, watching that directory, and printing and deleting any files that are placed there.

15.1.2 Installation and Use – Unix/Linux

To make the Print Wizard definition files available in Unix/Linux-based filePro, copy one or both PRT files into the “lib” directory of your “fp” directory tree. For instance, into “/appl/fp/lib”.

➤ Make sure the file name is in all lowercase.

To use, select “printwiz” or “printwzc” as your printer type in “Printer Maintenance”.

There are many ways, in this environment, to route the printer output through Print Wizard. Following are some brief descriptions:

- Use AnzioWin as your terminal emulator, and tell filePro to do passthrough print by setting the PFPT environment variable.
- Define a “remote printer” to Unix/Linux, which tells it to use the LPD protocol. Then run PWLPD to receive and process the data, on the same or a different PC.
- Route the data through the netprint program, to Print Wizard (freestanding) running on a PC.
- Have filePro print to file, in a particular directory that is visible to the PC via Samba, Vision FS, etc. Run Print Wizard (freestanding) on a PC, in despooler mode, watching that directory.

15.1.3 Usage Notes

Once everything is installed and working, you can start making creative use of the Print Wizard definitions. Most printcodes are self explanatory. Following are some general notes on using the printcodes defined for Print Wizard:

- The general printer initialization code is #3. This leaves Print Wizard in LEGACY mode, with a ½” left margin.
- Codes 55 to 58 can be used to initialize particular print jobs to certain orientation and linespacing.
- To include a bitmap inline, do #75 (Start bitmap), then the name of the bitmap file, then #76 (End bitmap). The bitmap file can be in any of the formats Print Wizard supports (.BMP, .GIF, etc.). It is wise to include an entire pathname.
- If you don’t want to include the extension (“.GIF”) when specifying a bitmap, use #68 (Initialize bitmaps) once at the start of the print job. Alter this code as needed.
- You could have a report that specified a bitmap inclusion as “logo.gif”. Then, before printing, your processing could copy a specific file into “logo.gif”. This would have the effect of changing which bitmap got printed, without changing the report specification.
- To print barcodes, do #72 once at the beginning of the print job, to establish the style and size of the barcode. You can alter this code as needed. Then, for each barcode, do #73 (Start barcode), then the data to be barcoded, then #74 (End barcode).
- Box drawing will work very well.
- Several special characters, such as the Euro, are provided.

Of course you are free to modify the PRT files as you like, with these tips and warnings:

- The editor built into filePro (at least in some versions) will mess up files with lines that are too long, such as these. Use a freestanding editor such as vi or notepad.
- Any embedded space must have a backslash before it.
- Initialization codes leave Print Wizard in LEGACY mode. Any following Print Wizard tag must be preceded by a trigger character control-Z, represented as “^Z” in this file.
- After the first PAGESIZE tag (such as in #3), you are in LEGACY mode. To have another PAGESIZE tag, it must immediately follow (in printing sequence); it must take you OUT of LEGACY mode, then do the PAGESIZE tag, then go back into LEGACY mode. This is demonstrated in #55.
- It would be possible to add or modify a code with a PAGESIZE tag to include an OVERLAY parameter. Then, if the named file existed, it would be used as a form overlay. If the file did not exist, no overlay would be used. Other coding could be used to create or copy, and delete, the named file.
- The BASE tag in #68 could be changed to include an HREF, which would be the assumed directory location for bitmap files.

Finally, if you are not getting the output you expect, you can do these things:

- Turn on the debug option in Print Wizard.
- Have filePro print to a file, and then open that file in an editor to see what is there. Study the PWML code in relation to this manual. Try editing the file and running it through Print Wizard. In general, see a) what needs to be generated in this file to get the printout you want, and b) what you need to do in filePro in order to put the right thing in the file.

15.2 Print Wizard and III Millennium and Innopac

Millennium and Innopac are products from Innovative Interfaces Inc., for the library industry. Print Wizard can be used to enhance, correct, adapt, and otherwise manipulate the print jobs coming out of these packages.

15.2.1 Connecting to Innopac via Attached Print

Innopac is a character-based application. Library staff can use AnzioWin as their telnet client to run all the functions of Innopac. To print in this environment, simply tell Innopac to use “attached print” (or something similarly named), that prints through the terminal session to the attached printer. In AnzioWin, turn on Print Wizard, and select the printer.

15.2.2 Connecting to Millennium or Innopac via Workstation JetDirect

Millennium is a Java-based system, so a telnet client is not needed. For this system, and also for Innopac, III has designed the “Workstation JetDirect” (WJD) printer type. When you tell Millennium or Innopac to print to a WJD printer, it sends the data using the JetDirect protocol, to either a) a specific IP address, or b) the same IP address as the client connection is coming in from. The latter is more popular, resulting in print data coming back to the user’s PC, for processing by Print Wizard.

What III calls the Workstation JetDirect protocol, Print Wizard calls “listen mode”. For this mode to work, both players must use the same “socket”; both default to socket 9100.

To get started, on your PC run the command:

```
printwiz /listen /L /debug /u /preview
```

This command can be entered in a “DOS window”.

This runs Print Wizard in listen mode, with logging and debugging information going to the screen. The “/u” specifies the UTF-8 character set (see below). Print preview is enabled. You will now be able to see what Print Wizard receives.

Now in Millennium or Innopac, start a print job that goes to a WJD printer. It should be received by Print Wizard on your PC, resulting in some information appearing in the Print Wizard window, and the Print Preview window appearing.

In Millennium, you can configure each login to have four printer types: standard, receipt, label, and form. Each of these can be tied to a system-attached printer, an email address, or a WJD printer. Note that you could go so far as to configure each printer type to go to a different WJD printer, each configured to use a different port (9100, 9101, 9102, 9103). Then you could run four instances of Print Wizard on your PC, each configured for a different port, and each configured to use a specific printer.

Once everything is working, you could cause Print Wizard to be started up automatically by Windows, by putting a shortcut in the Startup group.

15.2.3 Character Sets

Innopac/Millennium is able to deal with international character sets, including diacritics and CJK. When it goes to print some output, it checks the printer’s “diac map” setting to determine if or how to output non-ASCII characters. You can change this setting to change how international characters are output. Then you would want to match Print Wizard’s (or AnzioWin’s) character set configuration to match.

The most inclusive setting is Unicode UTF-8. This encoding can contain all the MARC diacritics, Chinese, Japanese, Korean, Cyrillic, etc. So we advise setting the “diac map” to UTF-8, and configuring Print Wizard or AnzioWin the same.

-
- If the Innopac/Millennium diac table is set to “HP” or some other sets, the data from the host could contain escape codes. This might throw Print Wizard into Spooler mode, which is not good. Your debug information will tell you if this is occurring.
-

15.2.4 What Print Wizard Will Do

Once you have Print Wizard processing the print jobs from the III software, Print Wizard will do some things for you automatically:

- Although III does not output formfeeds between pages, Print Wizard will figure out how many lines of text go on each page. (If Print Wizard has trouble doing so, configure III to use 66 lines per page.)
- Full-page reports will be auto-fitted to the page.
- Diacritics, CJK, and other international characters will be printed properly, with automatic font selection.
- Short output, such as labels and receipts, will trigger the mini-page logic. Print Wizard will tell Windows to make a custom page size, which will be just as long as is needed, so there will be no excess paper feeding at the end of the job.

Beyond this automatic handling, it is possible for you to modify the output format, by use of a print initialization file (as explained elsewhere). For this environment, you might want to:

- Adapt III’s single-column spine label format into multi-column full sheet label stock, for use in laser printers.
- Add text and/or logos to checkout receipts.
- Adjust margins for reports that will be bound.

16 Using MAPISEND

MapiSend is a freestanding program (MAPISEND.EXE) provided with the freestanding Print Wizard product. It provides a command-line interface to the Windows Messaging Application Programming Interface (MAPI).

16.1 What MAPI Is

MAPI is a set of messaging tools available from Microsoft for Windows 95 and later. Two levels of MAPI can be loaded. *Basic MAPI* is loaded when you install Outlook Express (or possibly Netscape Messenger), and does e-mail only. *Extended MAPI* is loaded when you install Outlook (part of Microsoft Office) or Exchange; it does e-mail and it *may* do faxing.

For faxing with Extended MAPI, you must have installed a MAPI-compatible fax program, either from Microsoft or from a 3rd party. Microsoft has been inconsistent in their support for faxing. One program, Microsoft Fax, was included in Windows 98 and available for Windows 95. Microsoft did not provide a fax application for Windows ME, NT, or 2000. Starting with XP, there is support at the operating system level, but configuring MAPI to support it is difficult.

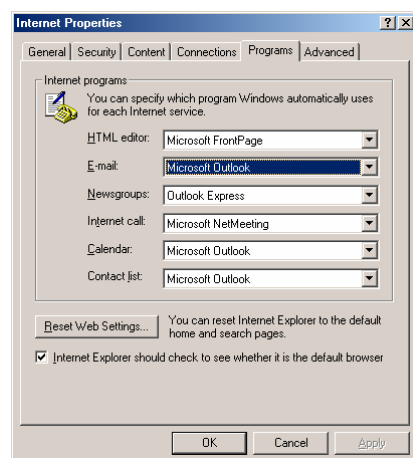
➤ Options and configuration vary by version of Windows.

16.2 Configuring MAPI

In general, you will want to configure your email and faxing (if supported) to the point that you can use them directly (without MAPISEND), as an initial step.

16.2.1 Choosing Your MAPI Program

The Internet Options dialog is used to specify which program will be used for sending e-mails (and faxes). From your Control Panel, select Internet Options, and go to the Programs tab. You should see this:



In the E-mail slot, make sure the correct program is selected.

➤ This slot affects faxing, also.

16.2.2 Basic MAPI

If you are using Outlook Express or Netscape Messenger, you will have access to Basic MAPI, for e-mail only. Configure the selected program to be able to send e-mails, and test that. This completes your setup.

16.2.3 Extended MAPI

If you are using Outlook (not Express) or Exchange, you have Extended MAPI. Extended MAPI has a concept of *profiles*. A profile is a set of services and options, used by a particular user at a particular time. You must configure your profile(s), and you must tell MAPISEND which profile to use. To do so, go to the Control Panel and select the Mail item.

-
- A *profile* contains one or more *services*, such as e-mail and faxing. A profile also has a *default* service. If you have trouble getting MAPI to use the service you want, try creating a profile with just one service, such as e-mail.
-

Because mail configuration varies between Windows versions, we will avoid the specifics here.

16.3 The MapiSend Program

MapiSend is designed to provide a command-line interface between the various MAPI functions and the user, without having to delve into writing your own MAPI application. Primarily, MapiSend provides Print Wizard with a method of e-mailing and faxing through the default installed mail and fax services.

For each “message” that you send with MAPISEND, you should specify either a faxnumber or an e-mail address. You have the option of specifying the recipient’s name. If you specify a recipient but **not** a faxnumber or e-mail, MAPI will try to use your address book to determine how to send it.

For an email, the file(s) named in the command will each be an attachment. Text given with “/n” or “/i” will be sent as the main part of the message.

For a fax, the file(s) named will be sent at the end of the fax. Text given with “/n” or “/i” becomes cover sheet text.

The general format for a MapiSend command is:

```
MapiSend [options] filename [filename2...]
```

where options can be:

/f<faxnumber>	To fax, include the fax number here, just as you would dial it.
/e<email>	The email address to send to.
/r<recipient>	The recipient name to include.
/s<subject>	The subject text to include.
/p<profile>	The Outlook/Exchange profile to use.
/n<note text>	The text to include as the body or cover sheet.
/i<include note file>	The full path to a file that contains the body or cover sheet text (replaces the /n note).
/d	Turn on debug messages to aid in debugging errors, including bringing up the send dialogs to manually send messages.

-
- If any parameter contains embedded spaces, put double quotes around that parameter.
-

16.4 How it Works – Notes on Various Configurations

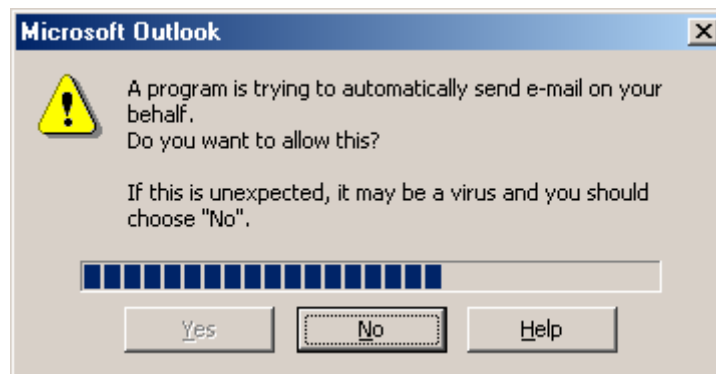
The following will go through several configuration notes. These fall in to the categories of:

- 1) Microsoft Outlook e-mailing
- 2) Outlook Express e-mailing
- 3) Netscape Messenger (version 4.0 or later) e-mailing
- 4) Microsoft Outlook faxing
- 5) Outlook Express and Netscape Messenger faxing
- 6) Others

Note that Microsoft Outlook and Exchange behave pretty much the same way.

16.4.1 Microsoft Outlook e-mailing

- You must specify a profile name with the /p<profile> option
- The “From” address and name will be taken from the profile setting
- You may get the following Security dialog in later versions of Outlook, depending on your configuration



- You may experience delays, depending on a) whether you are set to send e-mails automatically, and b) if your server experiences any delays. If you are not set for automatic e-mailing for some reason, you may need to bring up Outlook to begin the send.
- Attachments come in as regular attachments, even if they are standard text (.txt) files.

16.4.2 Outlook Express e-mailing

- If your PC has *only* Basic MAPI installed, you should not need to specify a profile. However, if Extended MAPI has been installed, you will probably need to specify a profile.

- The “From” address is determined from the default e-mail settings, not from a profile, so this could appear different than that for a profile
- If any bad or unsend e-mails are in the OutBox, you may get a dialog box up even if there is no “/d” option. You will have to close this to continue. If everything in your message is formatted correctly, your message should still go out (read through the messages carefully).
- Attachments come in as regular attachments, even if they are standard text (.txt) files.

16.4.3 Netscape Messenger e-mailing

- No Netscape profile is necessary.
- “From” comes from the default e-mail configuration.
- Any attachments with the .txt extension will usually become part of the body text (no way around this that we know of).

16.4.4 Microsoft Outlook faxing

- Use the “/f<phone number>” tag with complete phone number (no dialing properties apply).
- Include a profile that has the Microsoft Fax or the 3rd party fax service available.
- “From” comes from the profile name.
- This does not apply to Windows ME, 2000 or NT. Windows 98 comes with Microsoft Fax installed, and it depends on which service pack you use for Windows 95. It can also be downloaded from Microsoft for Windows 95, but will not work under ME, 2000 or NT.

16.4.5 Outlook Express and Netscape Messenger faxing

If you try to send a fax when either program is set as the default e-mail client (in Internet Options), they send the request through as an e-mail service with the phone number becoming the e-mail address to send to. Most e-mail servers will then reject this as “User Unknown” and cause an error dialog box to appear.

16.4.6 Others

There are other 3rd party e-mail applications available. The question is whether they support MAPI the same way. Many applications claim to be MAPI compliant, but may not support inter-program access.

There are also other 3rd party faxing applications available. Again while these may support MAPI, they may not support inter-program access correctly. Also note that a fax application without an e-mail application will be totally dependent on the selected e-mail application for MAPI and may not work correctly as a remote service (Winfax Pro is an example of this – while it supports MAPI as an Outlook helper application, it does not support MAPI as the default fax for Outlook).

16.5 Examples

Sending an e-mail with Microsoft Outlook as the e-mail reader:

```
Mapisend /s"Subject line" /n"Note for the body" /euser@domain.com /p"My Profile"  
file1.txt file2.xls
```



Sending an e-mail with Microsoft Outlook and an “include” file for the body:

```
Mapisend /s"Subject line" /ibody.txt /euser@domain.com /p"My Profile" file1.txt
file2.xls
```

Sending an e-mail with Outlook Express or Netscape Messenger as the e-mail reader:

```
Mapisend /s"Subject line" /n"Note for the body" /euser@domain.com file1.txt
file2.xls
```

Sending an e-mail with Microsoft Outlook and an “include” file for the body:

```
Mapisend /s"Subject line" /ibody.txt /euser@domain.com file1.txt file2.xls
```

Sending a fax with Microsoft Outlook as the e-mail reader:

```
Mapisend /p"My Profile" /f1-503-624-0760 file1.txt file2.xls
```

16.6 Errors

The following are just some of the possible errors (taken from the Microsoft include libraries). See MAPI or Outlook documentation for more information on each error. We do not guarantee the accuracy of these as they may change from release to release of Mapi. These are merely here as an example and to perhaps help understand what error Windows Mapi may be passing to you.

MAPI_E_NO_SUPPORT	02
MAPI_E_BAD_CHARWIDTH	03
MAPI_E_STRING_TOO_LONG	05
MAPI_E_UNKNOWN_FLAGS	06
MAPI_E_INVALID_ENTRYID	07
MAPI_E_INVALID_OBJECT	08
MAPI_E_OBJECT_CHANGED	09
MAPI_E_OBJECT_DELETED	0A
MAPI_E_BUSY	0B
MAPI_E_NOT_ENOUGH_DISK	0D
MAPI_E_NOT_ENOUGH_RESOURCES	0E
MAPI_E_NOT_FOUND	0F
MAPI_E_VERSION	10
MAPI_E_LOGON_FAILED	11
MAPI_E_SESSION_LIMIT	12
MAPI_E_USER_CANCEL	13
MAPI_E_UNABLE_TO_ABORT	14
MAPI_E_NETWORK_ERROR	15
MAPI_E_DISK_ERROR	16
MAPI_E_TOO_COMPLEX	17
MAPI_E_BAD_COLUMN	18
MAPI_E_EXTENDED_ERROR	19
MAPI_E_COMPUTED	1A
MAPI_E_CORRUPT_DATA	1B
MAPI_E_UNCONFIGURED	1C
MAPI_E_FAILONEPROVIDER	1D
MAPI_E_UNKNOWN_CPID	1E
MAPI_E_UNKNOWN_LCID	1F

```
/* Flavors of E_ACCESSDENIED, used at logon */

MAPI_E_PASSWORD_CHANGE_REQUIRED          20
MAPI_E_PASSWORD_EXPIRED                  21
MAPI_E_INVALID_WORKSTATION_ACCOUNT       22
MAPI_E_INVALID_ACCESS_TIME               23
MAPI_E_ACCOUNT_DISABLED                  24

/* MAPI base function and status object specific errors and warnings */

MAPI_E_END_OF_SESSION                    100
MAPI_E_UNKNOWN_ENTRYID                   101
MAPI_E_MISSING_REQUIRED_COLUMN           102
MAPI_W_NO_SERVICE                         103
```

