



## **UNDERSTANDING UNI<sup>P</sup>RI<sup>N</sup>T DRIVER OPTIMIZATIONS**

*TECHNICAL WHITE PAPER*



## Copyright and Trademarks

Copyright ©2005 INGENICA, a division of Bell Business Solutions. This documentation cannot be reproduced in full or in part by any means without our prior written consent.

We provide this documentation as is without warranty of any kind, express or implied, including but not limited to implied warranties of appropriateness for specific purposes or merchantability. In no event or under any circumstances shall we or our suppliers or distributors be liable for any damages whatsoever, including and without limitation, damages resulting from business loss, which may arise from the use or inability to use this documentation, even if we, our suppliers or our distributors have been previously advised of the possibility of such damages. Since some states do not allow the exclusion or limitation of liability for consequential or incidental damages, the above limitation may not apply to you.

Our name and logo are trademarks. Other brand and product names are the trademarks or the registered trademarks of their respective corporations.

# Table of Contents

<b>Introduction .....</b>	<b>1</b>
<b>Terms and Definitions .....</b>	<b>1</b>
What is the PDF file format? .....	1
What is file compression? .....	1
What are fonts? .....	2
What are ASCII and UNICODE? .....	2
<b>UniPrint Driver Property Selections Affecting Print Job Quality and Size .....</b>	<b>2</b>
Configuration Tab .....	3
Image Compression Tab .....	4
Advanced Options Tab .....	6



## Introduction

The objective of this paper is to provide the reader with a complete understanding of the UniPrint driver property selections and their relationship with print job quality and print job size.

## Terms and Definitions

### ***What is the PDF file format?***

PDF (Portable Document Format) is a superset of the PostScript language. PostScript is a language that describes a page (often referred to as a *page description language*). A PostScript file contains a set of instructions that describes how to *draw* a printable page of text and objects. PostScript files contain information about the dimensions of a page, and the location and size of the objects to be drawn. PostScript printers contain processors capable of interpreting the PostScript instructions to create hardcopy representations. PDF is built on the PostScript language but adds a pre-press presentation layer to the underlying instructions. They are a presentation of PostScript instructions but portable in the sense that they can be displayed on any *raster image*<sup>1</sup> device such as a monitor or printer. In addition to drawing instructions, PDF files supersede PostScript by describing a greater variety of objects such as fonts, image types, and include formatting instructions.

### ***What is file compression?***

File compression is a metaphor used to generalize a process for reducing file size understanding that hard disk space is finite and network bandwidth is limited. Strictly speaking, compression is a process that removes data or replaces redundant data with more efficient representations. The net result with either process is a smaller file. The former is categorized as *lossy compression* because data is actually removed and not replaced. An example of lossy compression is the JPEG image format. JPEG compression removes data indiscernible to the human eye. The resulting image file is smaller but no less characteristic to the human eye. The latter is categorized as *lossless compression* because no data is lost when the compressed file is reconstituted. An example of lossless compression is WinZip<sup>TM</sup>. WinZip *compresses* a file by using mathematical representations of data that use fewer bits than the original data being replaced. The net result is a smaller file but when the file is *uncompressed* it is restored to its original state, bit for bit.

Keep in mind that file compression is an encoding process and decompression is a decoding process. In the case of files being transferred from one computer to another, the appropriate software must be available to encode and decode the file respectively.

---

<sup>1</sup> PostScript instructions are processed by a Raster Image Processor (RIP). Adobe Acrobat Reader is an example of an application that uses a RIP to interpret and display PDF files. PostScript printers also use a Raster Image Processor to interpret PostScript instructions to render print jobs. UniPrint-ready Lexmark printers also contain a RIP used in conjunction with the printer's native device driver

## What are fonts?

A font refers to a *set* of letters or characters. Fonts are comprised of *mathematically drawn* representations of letters and characters referred to as *glyphs*. Glyphs are *not* individual bitmap images stored in various typographic styles. A computer draws a particular character (glyph) by referencing information, *metrics*, contained in a digital font file (usually loaded in memory). A digital font file contains information about the font family's font faces or *typefaces* (bold, italic, underline, etc.). This information is used to calculate the shape, size and kerning of individual glyphs, and subsequently used by the computer to draw the *raster image* on a monitor or printer. Fonts are constructed mathematically for consistent representation across a vast number of raster display devices.

There are three main categories of fonts used today, Type 1, TrueType, and Open Type.

- Type 1 Fonts** Type 1 fonts were developed by Adobe and are distributed with most Adobe products.
- TrueType Fonts** TrueType fonts were developed by Apple and then made available to Microsoft.
- Open Type Fonts** Open Type fonts are relatively new and jointly developed by Adobe and Microsoft. Open Type fonts are an extension of TrueType and work on Windows and Macintosh computers. Microsoft distributes Open Type fonts with Windows 2003 and are considered more efficient and extensible than Type 1 or TrueType fonts.

## What are ASCII and UNICODE?

- ASCII** ASCII is an acronym for the *American Standard Code for Information Interchange* which describes a computer standard for representing letters, numbers, and symbols as numeric constants. This standard only addresses 256 characters and was initially implemented when 8-bit computers were popular and most operating systems were English only.
- Unicode** Unicode significantly extends ASCII by describing a standard that provides a numeric constant for every character across all computer platforms and written languages. "The Unicode Standard is a character coding system designed to support the worldwide interchange, processing, and display of the written texts of the diverse languages and technical disciplines of the modern world. In addition, it supports classical and historical texts of many written languages." (<http://www.unicode.org/standard/standard.html>).

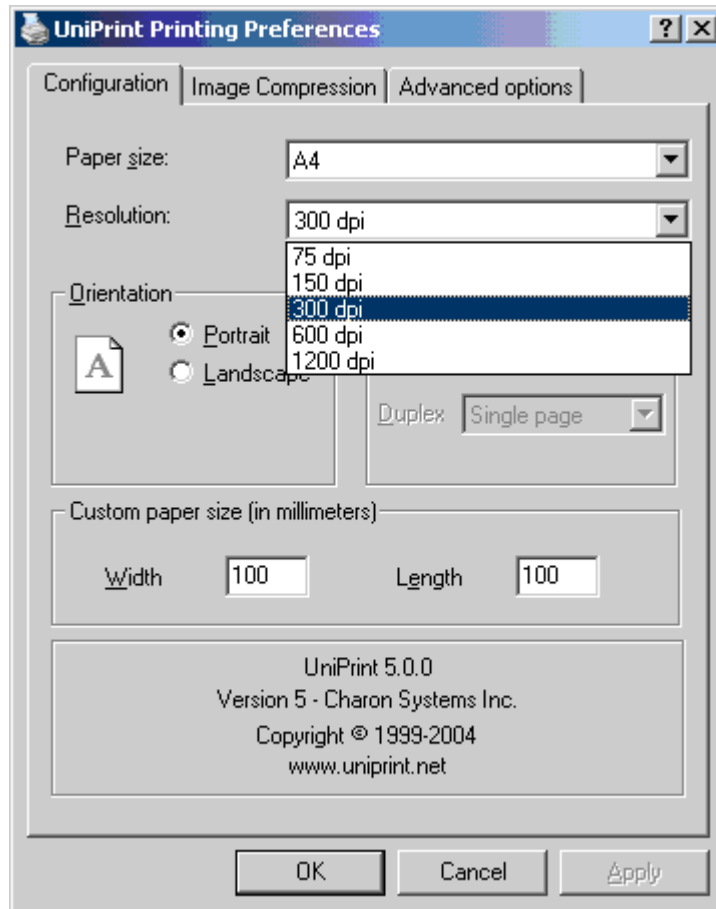
Modern operating systems are written using and supporting the Unicode standard. Fonts are also written using the Unicode standard.

## UniPrint Driver Property Selections Affecting Print Job Quality and Size

The UniPrint print driver is a universal driver used with UniPrint Server, UniPrint Gateway, and UniPrint Web Module. Therefore the property sheets and their settings apply to all of the above products. To access the property sheets shown:

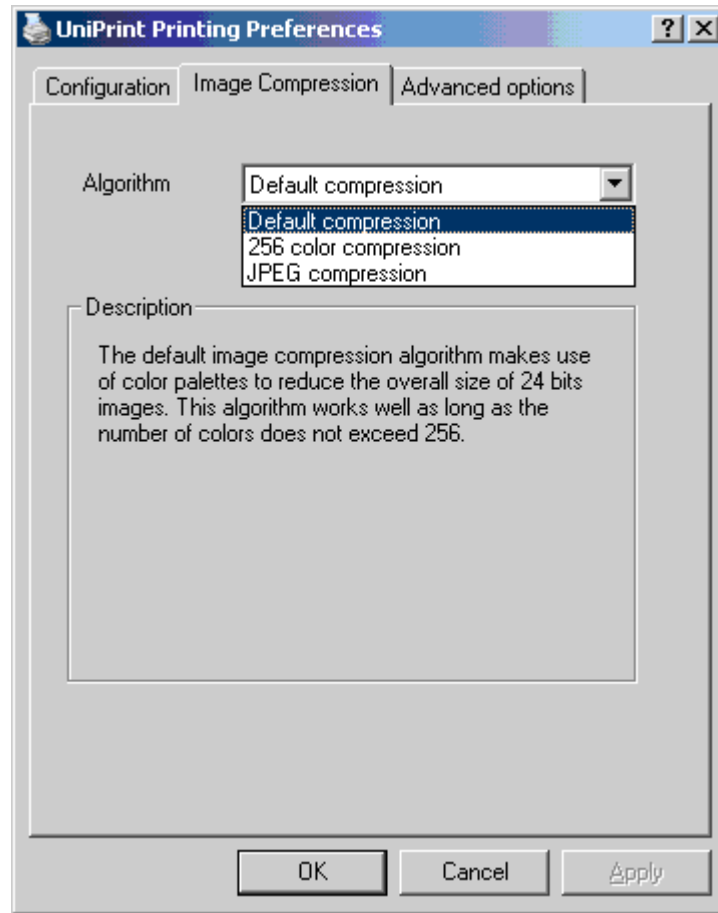
1. Open **Printer and Faxes**
2. Right-click on **UniPrint** and then click **Properties**
3. Click **Printing Preferences**

## Configuration Tab



**Resolution** Print resolution using the UniPrint Driver refers to the quality of images and the resolution of the destination printer and the created PDF print job. The selected dots per inch (dpi) value is entered into the *PCF* (Printer Configuration File) sent to the UniPrint client along with the generated PDF file. The settings contained in the PCF file are temporarily applied to the destination printer's driver settings until the job is spooled to the destination printer (See the Technical Sales Paper, *Understanding the UniPrint Suite* which can be found at [www.uniprint.net](http://www.uniprint.net)).

## Image Compression Tab



Selecting an appropriate compression algorithm depends largely on the type of documents or images being converted.

### **Default Compression**

The default image compression algorithm makes use of color palettes to reduce the overall size of 24-bit images. This algorithm works well as long as the number of colors does not exceed 256. Default Compression works best for word processing documents with a mix of text and bitmap graphics such as this document.

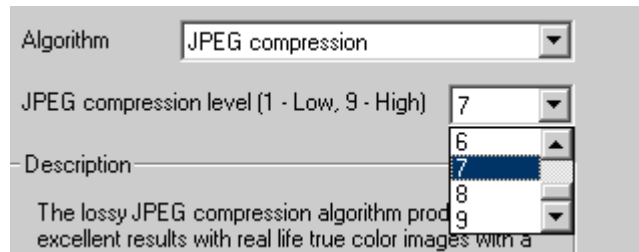
### **256 color Compression**

The 256 color compression algorithm calculates the most used 256 colors in an image and replaces all other color pixels by their closest match. This lossless compression algorithm usually works very well with most applications but is relatively slow on large images with a large number of colors. 256 Color Compression works best with documents containing images with less than 256 colours, e.g., 256 gray scale images or an Excel spreadsheet with a variety of colour graphs.

### **JPEG Compression level 1 – 9**

The JPEG (Joint Photographic Experts Group) compression method works best for reducing the size of colour images such as continuous-tone photographs. Typically, there is more detail in a 24-bit colour image than discernible on a display monitor or colour printout.

JPEG is a lossy compression algorithm. It achieves large compression factors by removing image data that may reduce image quality. The degree to which the JPEG algorithm removes data is fully configurable.

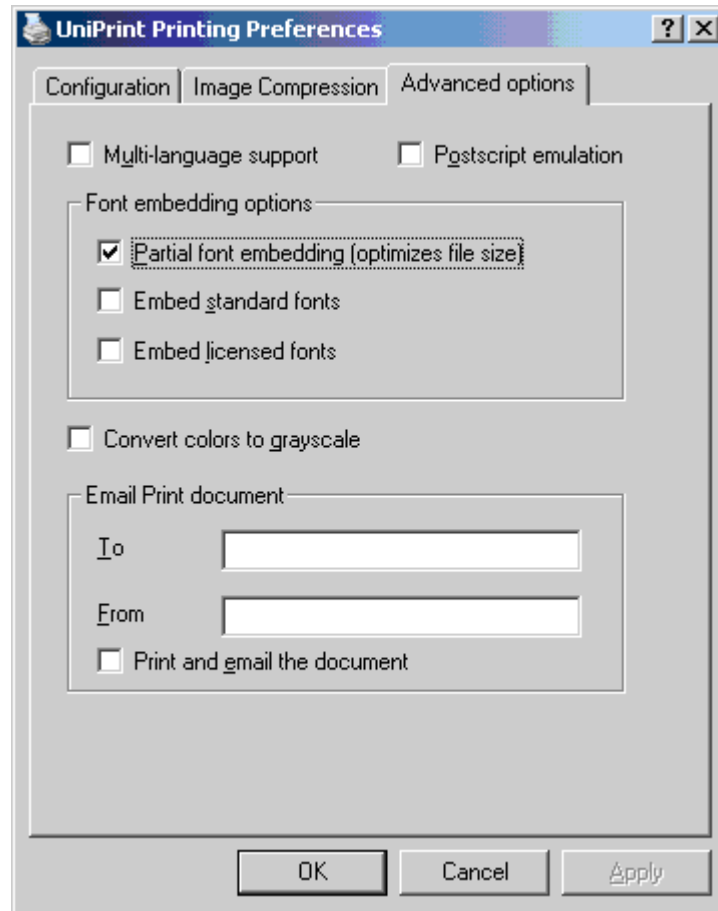


The JPEG compression level setting allows users to choose from 9 degrees of compression. The scale refers to the quality of the resulting image and is purely arbitrary, lowest quality to highest quality. Level 1 has the largest compression ratio that produces the smallest file but may affect the quality of the image. Level 9 has the lowest compression ratio but maintains the highest level of quality. The default level is 7, which produces images of suitable size and quality.

Use this compression algorithm to reduce the size of printed PowerPoint presentations containing high quality graphics or photographs. Greater compression factors will be achieved when the original image contains a lot of data. By contrast, grayscale images will not achieve large compression factors compared to colour images. The human eye is more sensitive to changes in grayscale than high colour – increased compression on grayscale images will have a much more noticeable effect on their quality. Since JPEG algorithmically removes data to create smaller files, experiment with its settings to find the optimal trade-off in file size and quality for the intended viewing audience.

Compressing images that have already been compressed using JPEG will lead to additional loss of information. It is, therefore, not recommended for documents with embedded graphics that have already been compressed or are already at a minimum level of quality prior to PDF conversion.

## Advanced Options Tab



### Multi-language support

When this option is checked, the UniPrint driver will *convert* non-Western European character sets or *Unicode* languages, e.g., East Asian or Eastern European. When selected, the *Embed standard fonts* option is automatically set. Provided this additional option is not manually cleared, UniPrint will attempt to embed available non-Western European fonts before performing character conversion. Character conversion consumes more CPU and RAM resources and increases print job file size and spooling time. Additionally, some PDF viewers may not correctly display converted fonts.

Only use this setting if font embedding is unavailable due to font licensing restrictions (See ["Font embedding options" on page 7](#)). Because of increased spool time and file size, this setting should not be used with source-documents containing large amounts of non-Western European language text for which the accompanying font cannot be embedded.

### Postscript emulation

Postscript emulation provides support for Type 1 fonts (See ["What are fonts?" on page 2](#)). Selecting this option enables UniPrint to fully or partially embed Type 1 fonts (See ["Font embedding options" on page 7](#)). Type 1 fonts are usually installed when Adobe products or PostScript print drivers are installed. Enable *Postscript emulation* for maximum portability of PDF print jobs and accurate native print driver rendering if printing from an application that uses Type 1 fonts. Type 1 fonts appear

in the Windows Font folder with an Adobe 'A' symbol, TrueType fonts appear 'TT', and Open Type font appear with 'O'.

Open Type fonts are considered more efficient than Type 1 or True Type fonts when spooled with native print drivers. It is recommended that either Open Type or TrueType fonts be used whenever possible. Open Type and TrueType fonts comprise higher than 95% of existing fonts on modern operating systems.

**Font embedding options**

Embedded fonts are digital font files contained as part of a document. Typically, digital documents do not contain all the details of each font used. Instead it contains references to the digital font files required for identical document recreation.

A font is an operating system resource. If the actual font files are not available to the operating system (not installed), then UniPrint cannot embed the font. If Terminal Servers where UniPrint is installed also contain native print drivers, then some fonts may appear to applications that are not actually installed. Some printers contain built-in fonts and communicate that to the operating system when their native drivers are installed but do not necessarily *install* the actual fonts. *Phantom fonts* appear and disappear to some applications when the default printer is changed. Keep in mind that a font cannot be embedded if it is not installed on the host system.

When a document is transported from one computer to another, the destination computer's font files are used to recreate or *rasterize* the document for viewing. If the destination computer does not contain the referenced font file, then an available substitute is used. In most cases, substituted fonts satisfactorily recreate the original document but not necessarily as the original author intended. In some cases a substitute font can alter the readability or even the intended meaning of a word. For example, if the substituted font did not contain the italic typeface, then italicized words might lose their intended meaning. In more extreme cases, the substituted font might display entirely different characters, gibberish to the reader. Font embedding is an option to include the digital fonts used in a source-document as a part of the document's metadata to ensure that any destination computer is able to recreate it exactly as the author intended.

Embedded fonts are not unique to PDF files. Many applications such as Microsoft Word, PowerPoint, and Internet Explorer (just a few examples) provide options for embedded fonts. Remember that digital font files *describe* how a character is to be drawn and do not contain bitmap images of those characters. Therefore all applications that use fonts for transportable documents must either reference the fonts being used or embed the digital font file as part of the document.

The trade-off between embedding fonts and not embedding fonts is size and possibly quality. This is a decision made by the author not the computer. If the author is distributing a document to colleagues who use the same company issued workstation, then not embedding fonts will produce a smaller file and likely render at the destination computer as intended. If the author is distributing a marketing brochure to the public, then embedding the fonts is the most likely way to ensure that the message is accurately received. There is a third and reasonable option not yet mentioned: Only use fonts in source-documents that are standardized across all computer platforms, e.g., Arial, Courier, and Times. Of course this may do just fine for whitepapers but seriously stifle marketing creativity. Additionally, even standard fonts have subtle differences, like the differences between Times and Times New Roman.

**Partial font embedding** Partial font embedding is set by default and instructs UniPrint to embed only those portions of the digital font file that are actually used by the source-document. For example, if the *Verdana* font is used without any characters appearing in italic or bold, then the Verdana digital font file is embedded without the font metrics for italic and bold. This setting manages the increase in file size as a result of the included font data, and ensures the source-document will be rendered accurately at the destination computer or printer.

Clear this setting if the source-document will be distributed to computers containing the same font sets used in the source-document. This can substantially reduce file size without compromising the quality of the print job.

**Embed standard fonts** *Embed standard fonts* instructs UniPrint to embed the digital font files for Arial and Times New Roman fonts in the source-document.

**Embed licensed fonts** Commercial font manufacturers may place license restrictions on some fonts prohibiting embedding. By default, UniPrint will not embed fonts tagged with licensing restrictions. Only check this option if the font to be embedded is licensed for embedding (consult the font vendor).

**Convert to grayscale** The *Convert to grayscale* setting significantly reduces the size of colour source-documents by converting colour text and images to grayscale. This setting only works with coloured text and the following image types: WMF, EMF, JPEG and BMP images will not be converted. This setting works best to reduce the size of UniPrint print jobs containing coloured Windows clipart images that will be printed to non-colour printers.