



Using Unicode

Skywire Software, L.L.C.
3000 Internet Boulevard
Suite 200
Frisco, Texas 75034
www.skywiresoftware.com

Phone:	(U. S.)	972.377.1110
	(EMEA)	+44 (0) 1372 366 200
FAX:	(U. S.)	972.377.1109
	(EMEA)	+44 (0) 1372 366 201
Support:	(U. S.)	866.4SKYWIRE
	(EMEA)	+44 (0) 1372 366 222
		support@skywiresoftware.com

PUBLICATION COPYRIGHT NOTICE

Copyright © 2008 Skywire Software, L.L.C. All rights reserved.

Printed in the United States of America.

This publication contains proprietary information which is the property of Skywire Software or its subsidiaries. This publication may also be protected under the copyright and trade secret laws of other countries.

TRADEMARKS

Skywire® is a registered trademark of Skywire Software, L.L.C.

Docucorp®, its products (Docucreate™, Documaker™, Docupresentment™, Docusave®, Documanager™, Poweroffice®, Docutoolbox™, and Transall™), and its logo are trademarks or registered trademarks of Skywire Software or its subsidiaries.

The Docucorp product modules (Commcommander™, Docuflex®, Documerge®, Docugraph™, Docusolve®, Docuword™, Dynacomp®, DWSD™, DBL™, Freeform®, Grafxc commander™, Imagecreate™, I.R.I.S.™, MARS/NT™, Powermapping™, Printcommander®, Rulecommander™, Shuttle™, VLAM®, Virtual Library Access Method™, Template Technology™, and X/HP™) are trademarks of Skywire Software or its subsidiaries.

Skywire Software (or its subsidiaries) and Mynd Corporation are joint owners of the DAP™ and Document Automation Platform™ product trademarks.

Docuflex is based in part on the work of Jean-loup Gailly and Mark Adler.

Docuflex is based in part on the work of Sam Leffler and Silicon Graphic, Inc.

Copyright © 1988-1997 Sam Leffler.

Copyright © 1991-1997 Silicon Graphics, Inc.

Docuflex is based in part on the work of the Independent JPEG Group.

The Graphic Interchange Format® is the Copyright property of CompuServe Incorporated. GIFSM is a Service Mark property of CompuServe Incorporated.

Docuflex is based in part on the work of Graphics Server Technologies, L.P.

Copyright © 1988-2002 Graphics Server Technologies, L.P.

All other trademarks, registered trademarks, and service marks mentioned within this publication or its associated software are property of their respective owners.

SOFTWARE COPYRIGHT NOTICE AND COPY LIMITATIONS

Your license agreement with Skywire Software or its subsidiaries, authorizes the number of copies that can be made, if any, and the computer systems on which the software may be used. Any duplication or use of any Skywire Software (or its subsidiaries) software in whole or in part, other than as authorized in the license agreement, must be authorized in writing by an officer of Skywire Software or its subsidiaries.

PUBLICATION COPY LIMITATIONS

Licensed users of the Skywire Software (or its subsidiaries) software described in this publication are authorized to make additional hard copies of this publication, for internal use only, as long as the total number of copies does not exceed the total number of seats or licenses of the software purchased, and the licensee or customer complies with the terms and conditions of the License Agreement in effect for the software. Otherwise, no part of this publication may be copied, distributed, transmitted, transcribed, stored in a retrieval system, or translated into any human or computer language, in any form or by any means, electronic, mechanical, manual, or otherwise, without permission in writing by an officer of Skywire Software or its subsidiaries.

DISCLAIMER

The contents of this publication and the computer software it represents are subject to change without notice. Publication of this manual is not a commitment by Skywire Software or its subsidiaries to provide the features described. Neither Skywire Software nor its subsidiaries assume responsibility or liability for errors that may appear herein. Skywire Software and its subsidiaries reserve the right to revise this publication and to make changes in it from time to time without obligation of Skywire Software or its subsidiaries to notify any person or organization of such revision or changes.

The screens and other illustrations in this publication are meant to be representative, not exact duplicates, of those that appear on your monitor or printer.

Contents

Using Unicode

- 2 Overview
 - 2 Related standards
 - 2 Implementing Unicode
- 4 Enabling Languages
 - 5 Choosing Languages in Windows 2000
 - 6 Choosing Languages in Windows XP
 - 8 Selecting the Input Language
- 10 Setting Up Unicode Fonts
 - 10 Using AGFA MonoType Unicode Fonts
 - 11 Licensing fonts
 - 11 Installing Fonts in Windows
- 12 Creating Forms with Docucreate
 - 12 Adding Unicode Fonts to the FXR File
 - 15 Adding Unicode Text
- 20 Using the Rules Processor
 - 20 Using Variable Fields
 - 21 Mapping Data
 - 22 Viewing UTF-8 Data
- 23 Printing Unicode Data
 - 23 GDI print driver
 - 23 PCL6 print driver
 - 25 PDF print driver
 - 25 Using Archive/Retrieve
 - 25 Using the Sample Unicode MRL
- 27 Using XML and Unicode
 - 27 Importing XML Files
 - 27 Encoding standards
 - 27 Memory representation
- 28 Frequently Asked Questions
 - 28 Why does text appear jumbled or change to question marks

after you change the font ID?

31 Index

Using Unicode

This document discusses how to use Unicode with Documaker version 10.2 and later.

This document includes information on these topics:

- [Overview on page 2](#)
- [Enabling Languages on page 4](#)
- [Setting Up Unicode Fonts on page 10](#)
- [Creating Forms with Docucreate on page 12](#)
- [Using the Rules Processor on page 20](#)
- [Printing Unicode Data on page 23](#)
- [Using XML and Unicode on page 27](#)
- [Frequently Asked Questions on page 28](#)

OVERVIEW

What is Unicode?

Basically, computers understand only numbers. Through various encoding systems (code pages), computers can associate numbers with characters for the purposes of display, printing, and processing text. Originally, encoding systems mapped values in a single byte (values between 0 and 255). Such a system (known as a Single Byte Character Set or SBCS) can only address up to 256 characters. The European Union alone requires several different encodings to cover all its languages.

Far eastern languages (Japanese, Chinese, and so on) contain 1000s of characters – too many for a SBCS. As a result, encoding systems using a mixture of one and two byte values were developed. These encoding systems are known as Double Byte Character Sets (DBCS) but would more accurately be called Multi-Byte Character Sets (MBCS).

In all, there are literally hundreds of encoding systems. It is not surprising that these encoding systems conflict with one another. That is, the character that should be associated with a number (computers only understand numbers) changes depending on the encoding system.

Unicode was developed to provide a single encoding for all of the world's major languages. Unicode provides a unique number for every character. Industry leaders such as IBM, Microsoft, HP, Sun, Oracle, SAP, Sybase and many others have adopted the Unicode Standard. See the Unicode web site for more information on the Unicode Standard:

www.unicode.org

Related standards

XML (eXtensible Markup Language) is a markup language for interchange of structured data. The Unicode Standard is the reference character set for XML content.

UTF-8 (Unicode Transformation Format, 8-bit encoding form) is a format for writing Unicode data in text files (which are normally processed sequentially, one byte at a time). Unicode values (code points) are written as a sequence of one to four bytes. When writing Unicode data into Documaker text files (FAP files, and so on), the Unicode data will be written in UTF-8 format.

Implementing Unicode

Documaker version 10.2 includes the first phase of Unicode support. This phase targets the ability to compose, print, and present documents for these Far Eastern languages: Chinese, Japanese, Korean, and Vietnamese.

Version 10.2 lets you...

- Use the Unicode 3.0 character set (two-byte Unicode values)
- Compose FAP files containing Unicode text
- Run the RP batch process to assemble documents from FAP files and data that contain Unicode text (on a limited set of platforms)
- Print documents with Unicode (using a limited set of printers)
- Archive, retrieve, and view the documents, including thick-client on Windows, and also PDF over the Internet

There are some limitations in version 10.2:

- All user interfaces, help, error messages, and documentation for tools and runtimes would still be single-byte ANSI characters, primarily in English.

- The target user is an English-speaking developer that needs to be able to create documents for Asian language users.
- Names of objects (field names, image names, DDT Edit Data text, and so on) will still be single-byte ANSI characters, primarily in English.
- Limited formatting or word-wrapping or other syntax related functions for Unicode text.
- No support for vertical writing style used in Japanese newspapers, magazines.
- No support for bi-directional languages (Arabic, Hebrew, and so on).
- Limited ability to do data entry of Unicode text, either using the thick-client or thin-client entry system.
- Development tools will require Windows 2000 or Windows XP platforms to develop Unicode forms
- Production runtime platforms that support Unicode will be Windows 2000, Windows XP and Solaris
- Printer support for Unicode forms will be through GDI (Windows print drivers), PCL (via new PCL6 driver), PDF (via TrueType font support). Unicode support will not be added to the Metacode print driver.

Future releases of Documaker may incorporate additional Unicode support such as...

- Support other languages requiring Unicode
- Data entry in Unicode variable fields
- Special formatting rules, data types, and text wrapping options
- Printer support for Unicode forms using PDF (with PostScript font support), PostScript, and AFP print drivers. Unicode support will not be added to the Metacode print driver.
- Additional platforms, printers, imports, exports, 3rd-party interfaces, as required
- Convert data into Unicode from other double-byte formats

Eventually, Skywire Software plans to create fully internationalized versions of the products. This involves translating the user interface, online help, error messages, and documentation, for tools and runtimes.

Experience shows that having the runtime version translated is probably more critical than having the development tools translated. Therefore, translation focus will be on the runtime system before the Docucreate tools.

ENABLING LANGUAGES

The Documaker Studio and Docucreate tools require Windows 2000 or Windows XP platforms to develop Unicode forms. To type characters in the languages you will be using on forms, you may need to enable these languages in Windows.

To enable additional languages in Windows 2000 or Windows XP, open the Windows Control Panel and double-click on the Regional and Language Options icon.

For instance, in Windows 2000,
double-click here



The remaining steps differ slightly, depending on which version of Windows you use.

If you use

See

Windows 2000

[Choosing Languages in Windows 2000 on page 5](#)

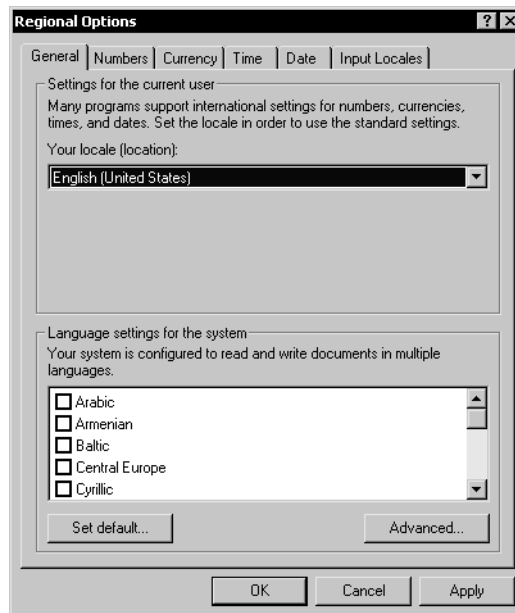
Windows XP

[Choosing Languages in Windows XP on page 6](#)

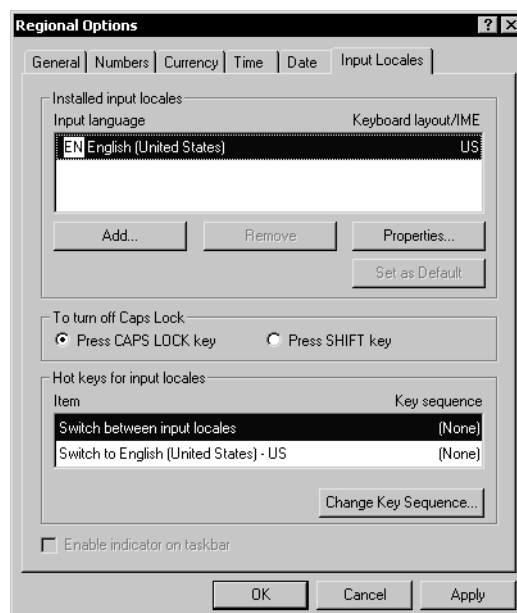
Choosing Languages in Windows 2000

In Windows 2000, the Regional Options window appears as shown below. In the bottom half of the window is a list of languages that your system is configured to use. Click on any additional languages you want to use and click Apply.

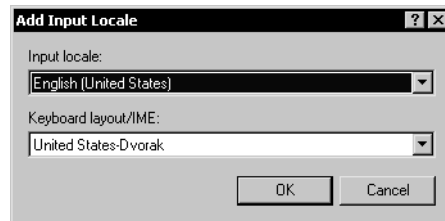
NOTE: This may require files from the Windows installation CD and a reboot.



After rebooting your machine, return to the Regional Options window and click on the Input Locales tab to view or change the languages and methods you use to enter text.



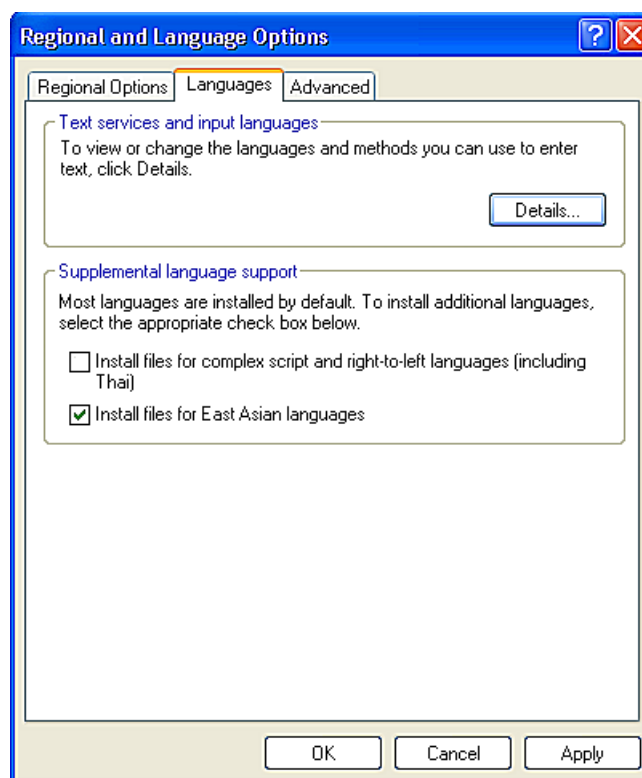
Click Add to add additional languages you want to input. Then select the input language and the keyboard layout/IME (input method editor) you want to use.



Choosing Languages in Windows XP

In Windows XP, the Regional and Language Options window appears as shown below. Click the Languages tab. Click the checkboxes in the Supplement language support section if you need to install support for East Asian languages and then click Apply.

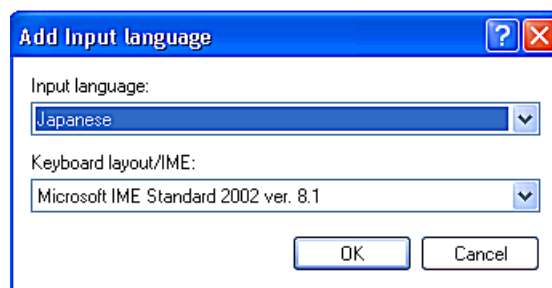
NOTE: This may require files from the Windows installation CD and a reboot.



After rebooting your machine, return to the Regional and Language Options window, click on the Languages tab and then click the Details button to view or change the languages and methods you can use to enter text.

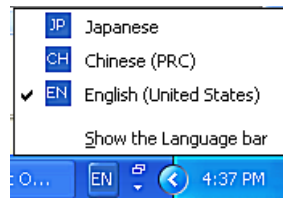


Click Add to add additional languages you want to input. Then select the input language and the keyboard layout/IME (input method editor) you want to use.



SELECTING THE INPUT LANGUAGE

Once you have selected the input languages you want to use, you should test the new input languages in a simple environment such as Microsoft Word. On the task bar, you should see a small icon with the letters, *EN* (assuming your original language for Windows was English), or you may see an icon for a typewriter. If you click on the *EN* or typewriter icon, you will see a list of the input languages now available in Windows.



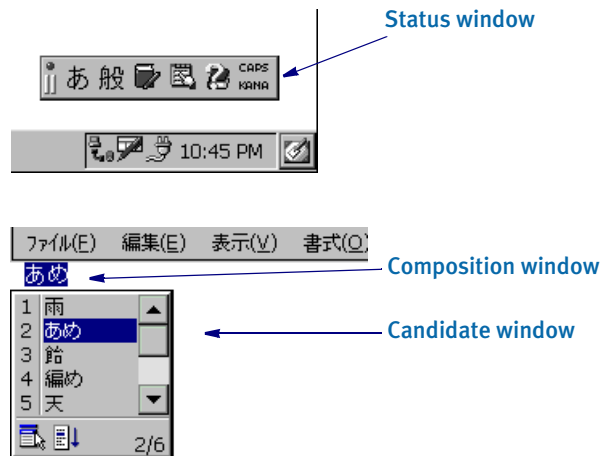
You can change input languages by clicking on the input language icon and using the mouse to select the new input language from the list provided. Alternatively, you can change the current input language by pressing and holding the ALT key and then pressing and releasing the SHIFT key until the desired input language is selected (it's easier than it sounds).

Once you have selected a new input language, you should be able to enter characters in the new input language. With some input languages, you can just use the keyboard as you normally would for English. For example, suppose you had installed and selected Russian as the current input language. If you typed the letters *ABC* on an English style keyboard, you would see these characters:

ФНС

Some input languages (like Japanese and Chinese) use an input method editor (IME) to type characters. The basic operation of an IME is as follows.

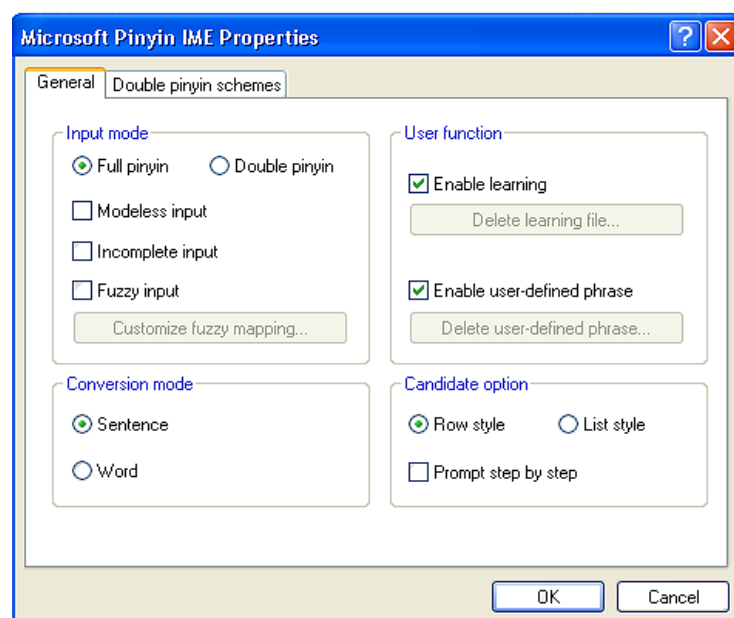
The IME user interface (UI) consists of a Status window, a default Composition window, a Candidate window, and a Guideline window. The appearance will vary based on the IME used.



As you type, characters are built into a composition string. The string appears in the default Composition window. The way you indicate the appropriate characters to use differs, depending on the IME. You can configure how the IME works by right clicking on the EN or typewriter button and selecting Settings.



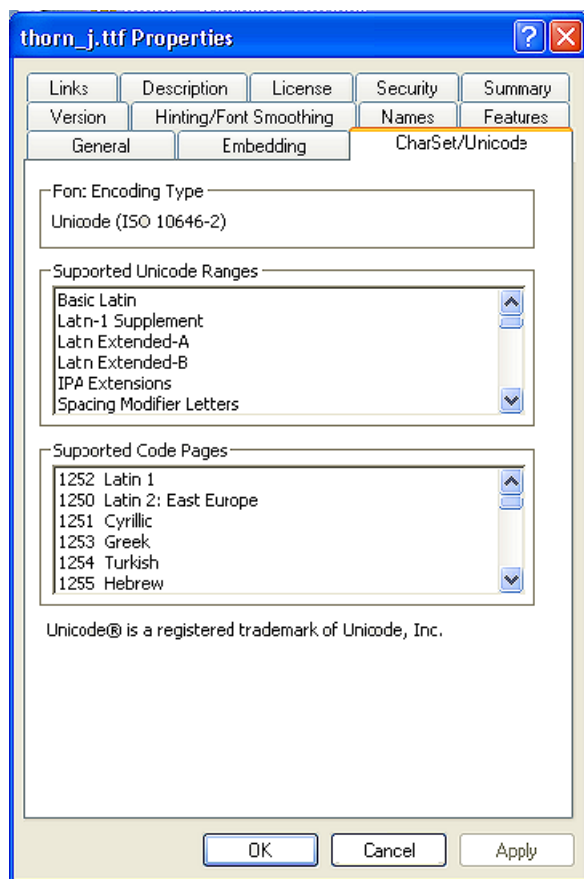
Highlight the IME you want to configure and click Properties. The Properties window varies based on the IME you use.



SETTING UP UNICODE FONTS

Most TrueType fonts contain Unicode encoding information. The other types of encoding formats for TrueType fonts are Symbol (used for fonts with graphic symbols), ShiftJIS (Japanese Industrial Standard), Big5 (used in Taiwan and Hong Kong and associated with traditional characters), PRC (used in Singapore and mainland China and associated with simplified characters), Wansung (Korean), and Johab (Korean). Claims of Unicode conformance in a font can be misleading when there is no common understanding about the number of languages supported.

Documaker version 10.2 supports TrueType fonts with Unicode encoding information. One way to determine the Unicode encoding information and languages supported for a TrueType font is to use Microsoft's Font Properties Extension utility. To find this utility, use the Search option at www.microsoft.com to search for the *Font Properties Extension utility*. Once installed, you can right click on a TrueType font to get a properties window such as the following:



USING AGFA MONOTYPE UNICODE FONTS

AGFA Monotype has provided a TrueType font, THORN_J.TTF, which you can use **solely for testing purposes**. This font, named Thorndale WT J, is one of AGFA Monotype's WorldType (WT) fonts. The *J* in the name of this font indicates it uses the Japanese style in the shared character positions of the Unified Han range. The Thorndale WT J font has 39196 glyphs, which means it can produce nearly all of the characters in the Unicode 2.1 character set.

NOTE: You *cannot* use the Thorndale WT J font for production purposes without first licensing it from AGFA Monotype. There are other WorldType fonts from AGFA Monotype you may want to license, such as the Andale Mono WT font, which contains the complete Unicode 3.0 character set.

Licensing fonts

You are not restricted to using AGFA MonoType fonts. You can license TrueType fonts from other vendors, as well. In addition, you can license fonts that support a only few languages or you can license fonts that support the entire Unicode 3.0 character set.

There are several legal issues to be aware of before converting fonts from other vendors. Font vendors generally copyright the fonts they create. You can legally convert fonts only if the font vendor grants permission.

The converted font is bound by the same copyright restrictions that apply to the original font. For example, if your license does not permit you to use the font on more than one computer at a time, then you are not permitted to use the converted font on more than one computer at a time. In addition, it may be a copyright violation to copy converted fonts to other platforms running on the same computer.

Generally, if you plan to use a TrueType font on more than one machine or plan on permanently installing the font on a printer, your license expressly must give you the right to do so.

Installing Fonts in Windows

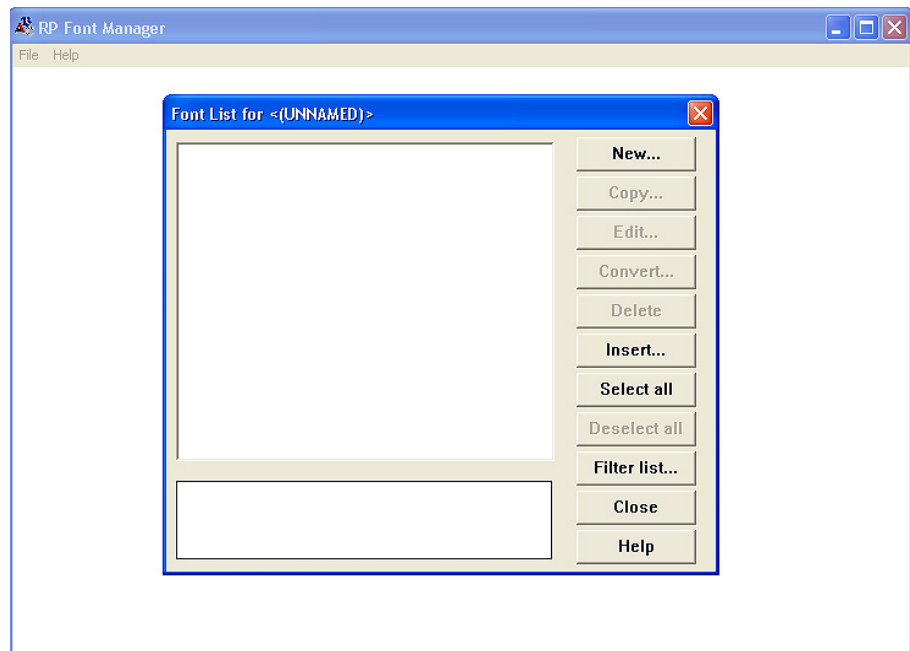
You must install the TrueType font into Windows before you can begin using the TrueType font to design forms. You can install fonts into Windows using the Fonts folder (located in the Control Panel). After opening the Fonts folder, select the File, Install New Font option. The Add Fonts window appears and asks for the drive and directory in which the new TrueType font files are located. When you finish selecting the fonts you want to install, click Ok to install them.

CREATING FORMS WITH DOCUCREATE

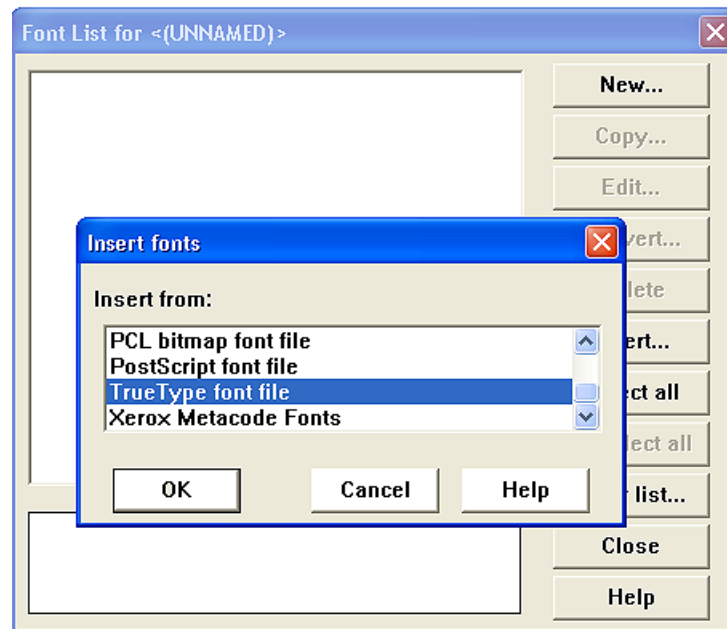
Before you can create forms containing static text or variable field text with Unicode characters, you must first build a font cross-reference (FXR) file from a Unicode TrueType font.

ADDING UNICODE FONTS TO THE FXR FILE

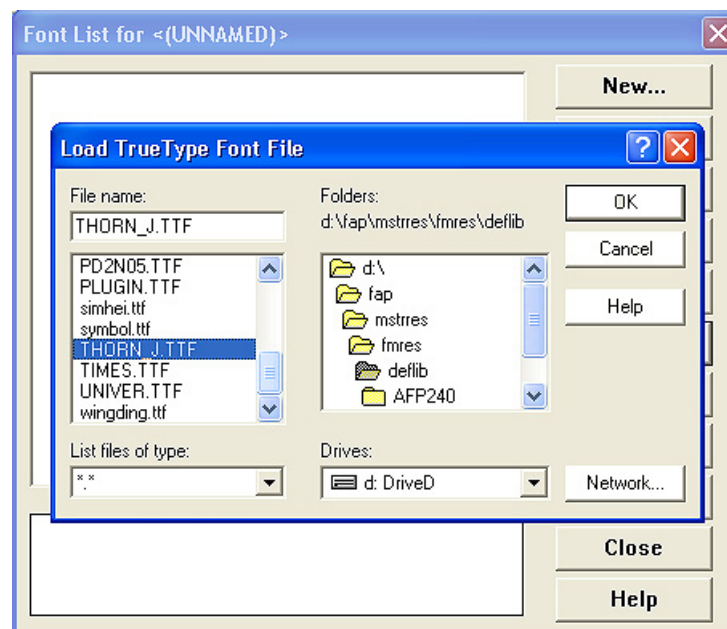
You can add Unicode-enabled fonts to a font cross-reference file via Font Manager. In this example, use Font Manager to create a new FXR file that will contain Unicode-enabled fonts. After starting Font Manager, use the File, New option to create a font cross-reference file.



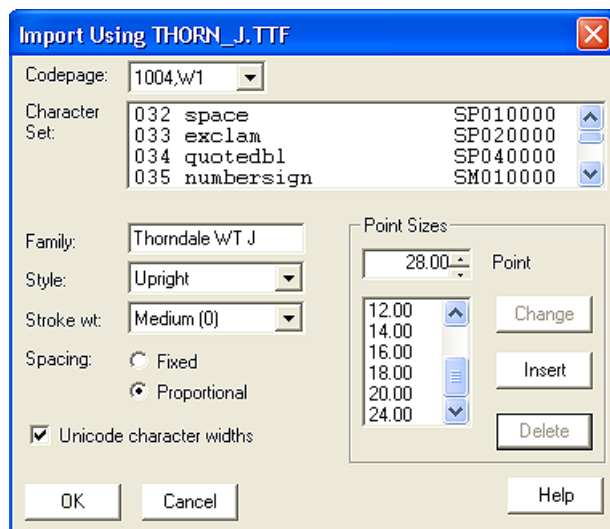
Click Insert to see a window which lists the types of fonts you can insert. Scroll the list down until you see *TrueType font file*. Highlight it and click Ok.



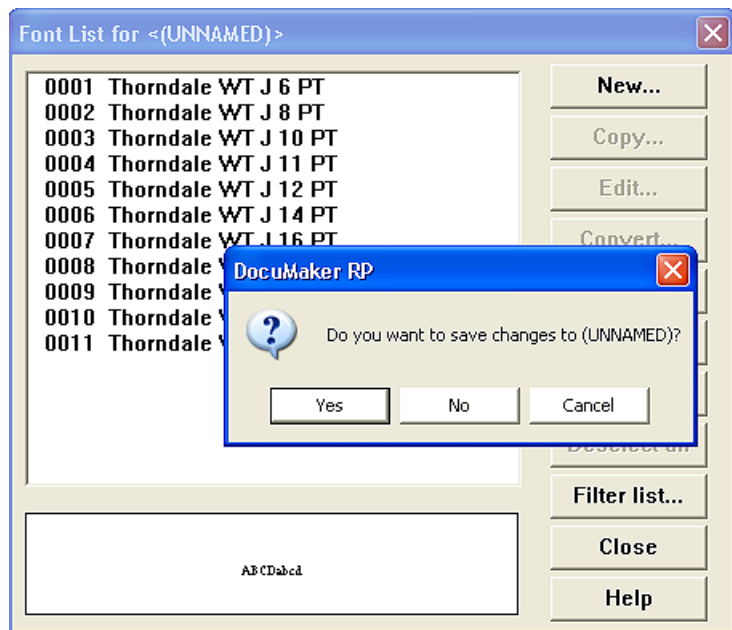
The File Open window appears. Select the TrueType font to insert, such as the Thorndale WT J (THORN_J.TTF) TrueType font.



To enable Unicode support for this TrueType font, you must click the Unicode Character Widths field on the Import window. After you have added all of the point sizes you want to use for this TrueType font, click Ok.



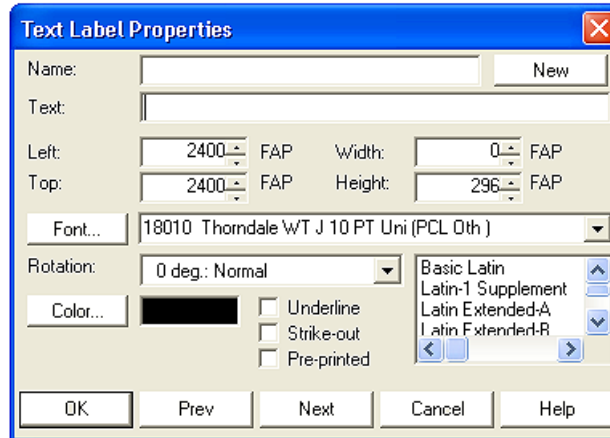
After the system has inserted the point sizes you specified for this TrueType font into the font cross-reference, you will see the main font list window again. Click Close to save your font cross-reference file.



ADDING UNICODE TEXT

After you have created a font cross-reference file that contains Unicode-enabled fonts and specified that font cross-reference in your master resource library, then you are ready to begin creating forms with Unicode characters.

In the Image Editor, insert a new text label as you normally would. The Text Label Properties window lets you enter Unicode characters.



If the font you have selected for this text label is Unicode-enabled then you will see the letters *Uni* in the font description after the point size. If you have also installed the Unicode font as a screen font, you will see a list of Unicode character ranges supported by the screen font.

The list of Unicode character ranges is purely informational; selecting one of the entries has no effect. If you only see the word, *ANSI*, in the list, then the screen font is not Unicode-enabled.

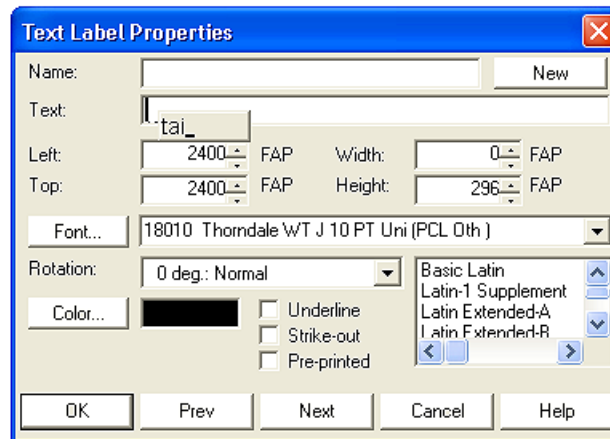
To enter Unicode text into the Text field, select the input method editor (IME) you want to use by clicking the EN icon on the task bar. After you have selected the IME to use, you can begin typing characters for that language. As you accept characters from the IME, they will appear in the Text Label Properties window.

The mechanics of each input method editor (IME) are different so you will need to learn the specifics of the IME you will use. Some input method editors let you type the letters that represent the pronunciation of the word you want to enter.

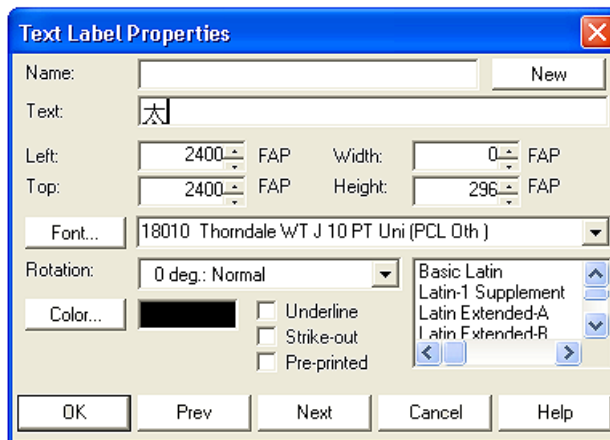
For example, the word *sun* is pronounced as *tai yang* in Chinese and is displayed as shown here:

太陽

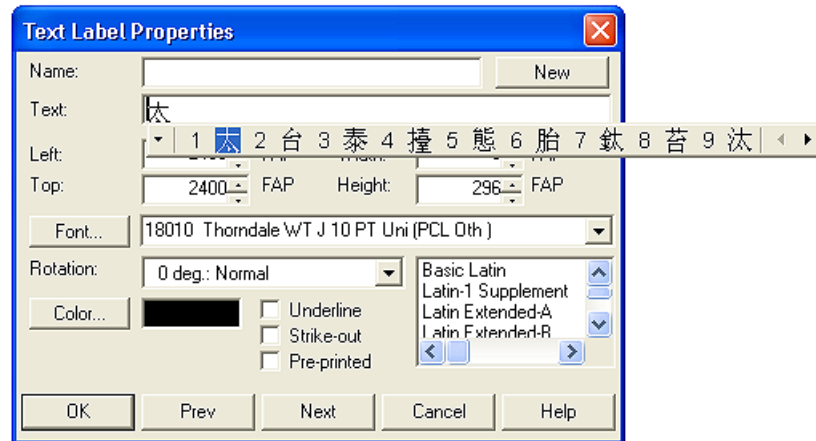
After enabling a Chinese IME, you would begin by typing the letters *tai*. The composition window might appear as follows:



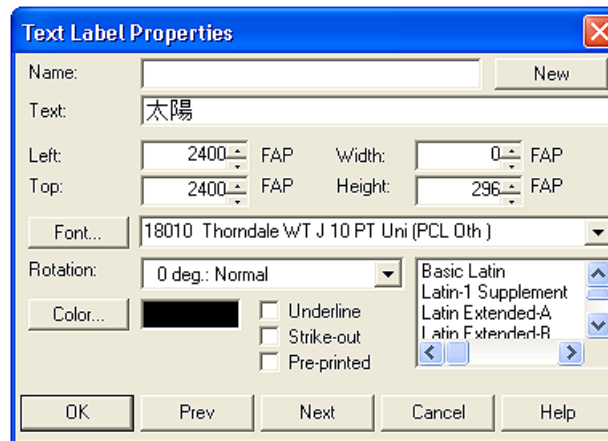
When you pressed the SPACEBAR, the letters in composition window disappear and the first character appears in the Text field.



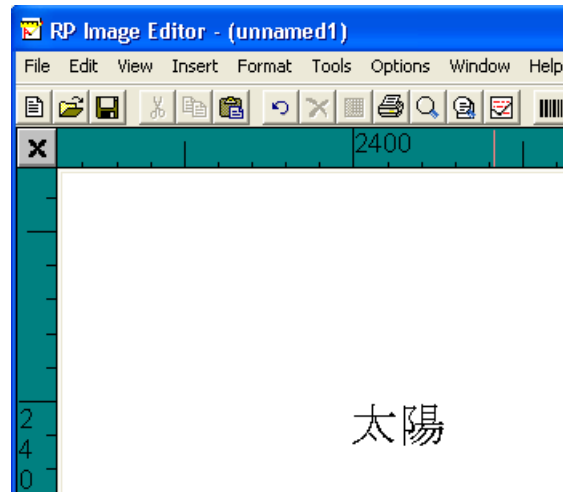
The character displayed is just the first choice of a list of candidate characters that sound similar to *tai*. Some IMEs show you this list of candidate characters immediately and some require you to mouse click on the initial character.



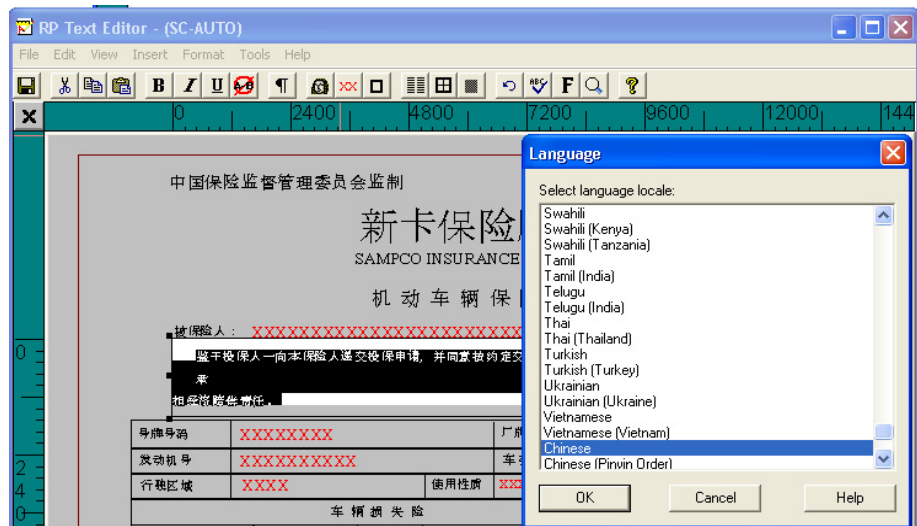
Once you have selected the character you want to use, press the SPACEBAR to finalize the selection. The display of the character and cursor will change slightly to indicate that the IME has completed the character. At this point, you would begin typing *yang* and selecting 陽 as the final character.



After you finish typing the characters for the text label, click Ok and the characters appear on the form.



Similarly, you can create text areas with characters for the IME you have selected. As you accept characters from the IME, they appear in the text area. To handle word wrapping for a given language, use the Tools, Language option in the text area editor as shown below.



When creating variable fields, you must select a font that is Unicode-enabled — more on this later. When you have finished composing your form in the Image Editor, it might look something like this:

中国保险监督管理委员会监制 限在上海市销售

新卡保险股份有限公司
SAMPACO INSURANCE COMPANY LIMITED OF CHINA

机动车辆保险单 (正本) No: XXXXXXXX

被保险人: XXXXXXXX 保险单号: XXXXXXXX

鉴于投保人向本保险人递交投保申请, 并同意按约定交纳保险费, 本保险人依据承保险别及此单对应条款和特别约定, 承担经济赔偿责任。

号牌号码	XXXXXXX	厂牌型号	XXXXXXXXXXXXXXXXXXXX
发动机号	XXXXXXXXXX	车架号	XXXXXXXXXXXXXXXXXXXX
行驶区域	XXXX	使用性质	XXXXXX
		座位/吨位	XXXXXXX
		初次登记年月	XX年 XX月

车辆损失险				第三者责任险	
保险价值	保险金额	费率 (%)	基本保险费	保险费小计	赔偿限额
XXXXXXXXXX	XXXXXXXXXX	XXXXXX	XXXXXXXXXX	XXXXXXXXXX	XXXXXXXXXXXXXX
险 别			保险金额 (赔偿限额)	费率 (固定保险费)	保险费小计
全车盗抢险			XXXXXXXXXXXXXX	XXXXXX	XXXXXXXXXXXXXX
车上责任险	车上座位	XXXXXXXX元/座 X XXXXX座	XXXXXXXX元/座	XXXXXXXX	XXXXXXXXXXXXXX
	车上货物	XXXXXXXXXXXXXX	XXXXXX	XXXXXXXX	XXXXXXXXXXXXXX
无过失责任险			XXXXXXXXXXXXXX	XXXXXX	XXXXXXXXXXXXXX
车载货物掉落责任险			XXXXXXXXXXXXXX	XXXXXX	XXXXXXXXXXXXXX
玻璃单独破碎险			XXXXXXXXXXXXXX	XXXXXX	XXXXXXXXXXXXXX

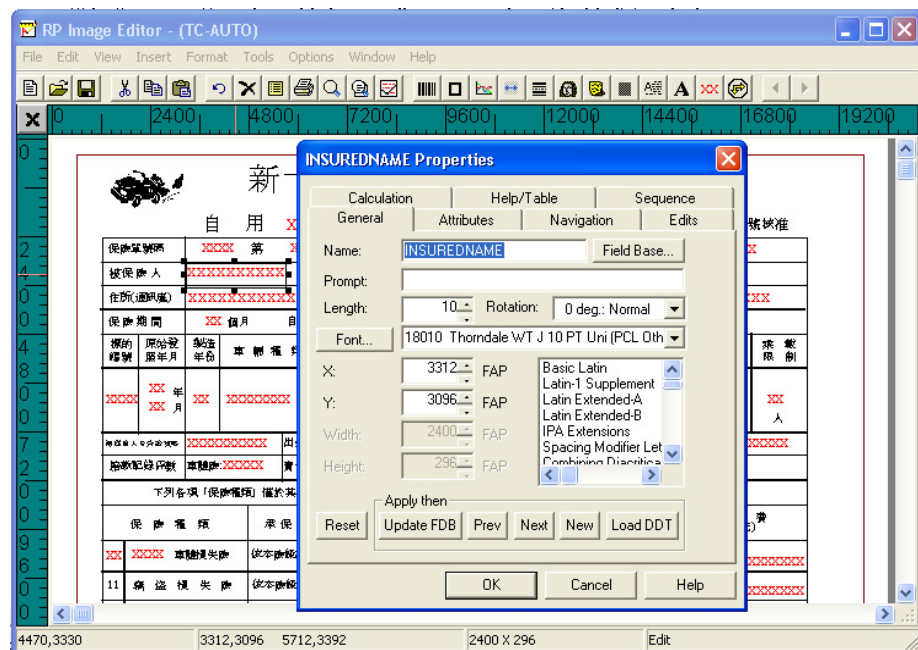
USING THE RULES PROCESSOR

The Rules Processor uses resources you create using Docucreate to process information and forms. This processing includes merging external data onto forms, processing data according to rules you set up, creating print-ready files, archiving data and forms, and, if applicable, sending incomplete forms to Documaker for completion by a user.

USING VARIABLE FIELDS

To enable a variable field for Unicode data, you must use a font ID that is Unicode-enabled. In this example, a field called INSUREDNAME has been created and it uses a font ID of 18010. You can tell that this font ID is Unicode-enabled because of the characters, *Uni*, displayed after the point size in the font description.

If you have also installed the Unicode font as a screen font, you will see a list of Unicode character ranges supported by the screen font. The list of Unicode character ranges is purely informational; selecting one of the entries has no effect. If you only see *ANSI* in the list, the screen font is not Unicode-enabled.



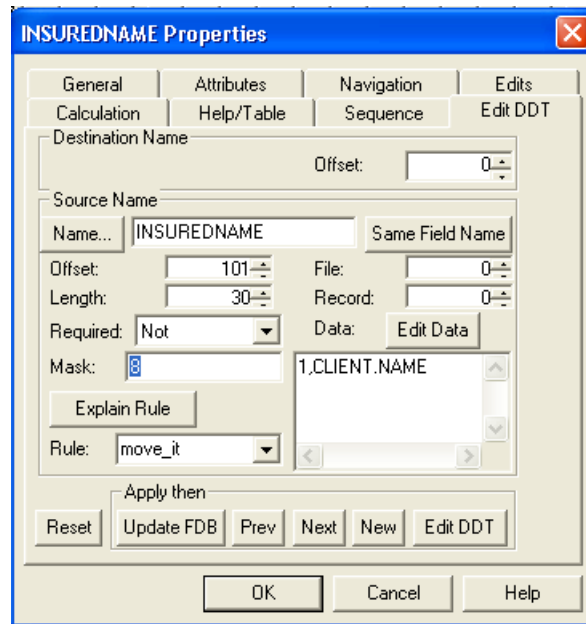
NOTE: The variable field cannot be a multi-line text variable field if the mapped data requires Unicode fonts.

MAPPING DATA

Once you select a font ID that is Unicode-enabled for this variable field, use the Move_It rule to map the data into this field. The Move_It rule can move UTF-8 formatted data into a Unicode-enabled variable field.

UTF-8 (Unicode Transformation Format, 8-bit encoding form) is a format for writing Unicode data in text files (which are normally processed sequentially, one byte at a time). UTF-8 data can be written into an extract file and subsequently used by the Rules Processor. Be sure to set the LoadCordFAP INI option to Yes when using Unicode variable fields.

The Move_It rule supports a mask of 8 to indicate that the extract data for this field is stored in the UTF-8 format. Here is how you specify the 8 mask (UTF-8) for a variable field in the Image Editor:



As you can see, the field *INSUREDNAME* uses the Move_It rule and has a mask of 8 to indicate that the extract data is in UTF-8 format.

Now look at the extract file data to make sure this is correct. Note in the window above, the record for *INSUREDNAME* is identified by the text, *CLIENT.NAME* starting in column 1. The actual data to be used for *INSUREDNAME* is stored at Offset 101 for a length of 30. Here is an example of this record in an extract file:

```
CLIENT.NAME      :â••æ»â,fæ••
```

The colon (:) indicates column 100, the UTF-8 data follows the colon at column 101.

The UTF-8 data (â••æ»â,fæ••) is not very readable in a regular text editor.

VIEWING UTF-8 DATA

One way to view this data is through the Internet Explorer. By loading the extract file in the browser, the example record in the extract file would appear as follows:

CLIENT.NAME : 喬治布殊

In other words, the UTF-8 data (â--æ²»â , fæ@š) can be interpreted as the following characters (喬治布殊). In many cases, the browser will automatically detect the UTF-8 data and display the UTF-8 data appropriately. If it does not, choose View, Encoding and select the Unicode (UTF-8) option.

PRINTING UNICODE DATA

To print Unicode data, you must have printer fonts and printer drivers that support Unicode. Documaker has three printer drivers that support Unicode: GDI, PCL6, and PDF (via TrueType fonts). Unicode support for AFP, PostScript and PDF (using PostScript fonts) may be added at a later date.

GDI print driver

The GDI print driver allows you to use a Windows print driver for printing. There is nothing special you need to do to print Unicode using the GDI driver. For information on using the GDI print driver, see the *Using the GDI Driver* topic in the Setting Up Printers chapter in the [Documaker Server System Reference](#) and other manuals. Since our PostScript driver does not support Unicode yet and our PCL5 driver will not support Unicode, you can use the GDI driver to print to these types of printers.

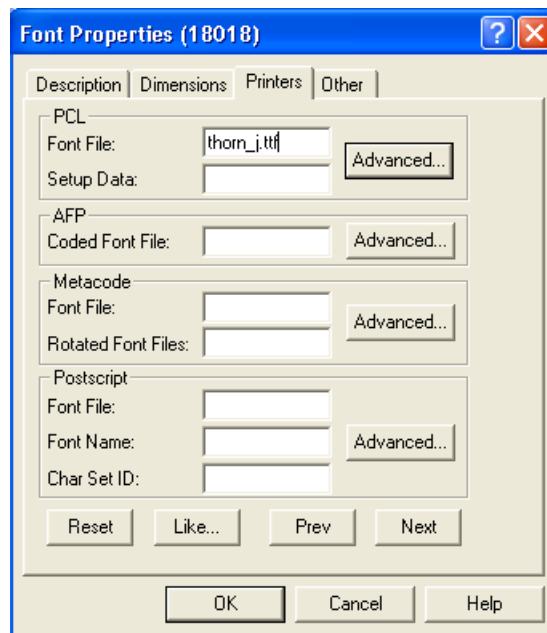
PCL6 print driver

Documaker includes both the original PCL printer driver, PCLLIB, which supports PCL 5, 5c, and 5e and a new PCL printer driver, PXLLIB, which supports PCL 6.

PCL 6 is a newer version of the PCL language, not a superset of PCL 5. It is a completely different page description language (PDL). PCL 6 is a stack-based protocol (similar to PostScript) composed of attributes and operators that let you define paths, clip paths, pens, brushes, fonts, raster patterns, and so on. PCL 6 also supports 16-bit character codes, which makes it a better choice for supporting Unicode than PCL 5.

Skywire Software's PCL6 print driver will support Unicode data, the PCL5 driver will not.

Both the PCL5 and PCL6 print drivers use the PCL printer section of the FXR file. The PCL6 print driver supports PCL5 bitmap fonts and TrueType fonts. You can use PCL5 bitmap fonts for printing ASCII data (SBCS). However, to printing Unicode data, you must use TrueType fonts. To specify a TrueType font in the PCL printer section of the FXR, enter the name of the TrueType font in the Font File field as shown below:



The TrueType font must reside in the FontLib directory as specified in your master resource library (MRL).

You must define the necessary printer options for the GenPrint and other Documaker programs to produce PCL 6 output. These options specify PCL 6 output and are located in a `PrtType:xxx` control group, such as `PrtType:PXL`. Many of the PCL 5 and PCL 6 printer INI settings are the same.

The following INI options are different for PCL 6:

```
Module           = PXLW32
PrintFunc        = PXLPrint
DownloadFonts    = Yes (internal font selection is not supported)
```

These INI options are not supported for PCL 6:

- `SendOverlays`
- `OverlayPath`
- `OverlayExt`
- `FitToWidth`
- `AdjLeftMargin`
- `StapleBin`
- `PJLCommentScript`
- `PJLCommentOn`
- `OutputBin`
- `HighlightColor`
- `Tray#` (where # is 1,2,3, and so on)

Keep in mind...

Because the PCL 6 driver supports PCL bitmap fonts, you can use existing MRLs. Just remember to change your INI options as discussed above.

The PCL 5 driver does not support TrueType fonts

When printing with a TrueType font using the PCL 6 driver, only the characters used on the form are downloaded into the print stream. This reduces the size of print stream files, particularly if the TrueType font includes support for Asian languages.

In comparison to the PCL 5 printer driver, the PCL 6 driver has these limitations:

- No overlay support
- No support for a separate downloadable font file which contains multiple PCL fonts
- No internal printer font support
- Less paper tray support, no INI options to specify which PCL commands to use
- No INI options to specify PCL commands to output bin or staple bin
- No highlight color support
- No comment script support

PDF print driver

The PDF print driver now supports Unicode data. Previously, PDF files produced by the PDF driver could be viewed in Acrobat version 3.0 and above. As a result of adding Unicode support to the PDF driver, you must now use Acrobat version 4.0 and above (preferably version 5.0 and above).

To create a PDF file containing Unicode data, you must specify a TrueType font to be embedded (downloaded) into the PDF file. You cannot use the internal Acrobat fonts or embed PostScript fonts when printing Unicode data; you must embed a TrueType font. The process for embedding a TrueType font into a PDF file is the same as it was before. See the *Using the PDF Converter* chapter in the Internet Document Server User Guide for more information on embedding TrueType fonts.

Using Archive/Retrieve

If you use Documaker to produce DAP archives containing Unicode data, you will not be able to view the Unicode data with earlier releases of DAP/RP. The Unicode data in the NA file will be written out as UTF-8 data. To look at the UTF-8 data as Unicode characters, see [Viewing UTF-8 Data on page 22](#).

Using the Sample Unicode MRL

The sample master resource library for demonstrating Unicode support is relatively small and easy to understand. Create a directory for your sample MRL (such as \fap\mstres\uc_test) and copy the ZIP file (UC_TEST.ZIP) to this directory. The ZIP file, contains several subdirectories that are created when you unzip the file. After unzipping UC_TEST.ZIP, you should see these files and directories:

Directory or file	Description
...ARC	Contains DAP archives
...DATA	Contains batches, NA/POL files, and print streams
...DEFLIB	Contains DDT, DFD, JDT, and FXR files
...EXTRACT	Contains extract files and double_byte.ext files
...FONTS	Contains the Thorndale WT J TrueType font (THORN_J.TTF)
...FORMS	Contains sample forms
...HELP	Not used
...RESULTS	Not used
...TABLE	Not used
...WIP	Contains WIP files
RUNPXL.BAT	Runs the GenTrn, GenData, GenPrint, and GenArc programs
USERINFO.DBF	Login User database
FAPARC.INI	Used by the GenArc program

Directory or file	Description
FAPCOMP.INI	Used by Image Editor
FSISYS.INI	Used by the GenTrn, GenData, GenPrint, and GenArc programs
FSIUSER.INI	Used by the GenTrn, GenData, GenPrint, and GenArc programs
TRANSLAT.INI	Error messages for the GenPrint program
USERINFO.MDX	Login User database index
MEN.RES	Used by Documaker Workstation
USER.RES	Used by the USRMAINT program

From this directory, you should be able to run any of the Docucreate programs (Image Editor, Font Manager, Logo Manager and so on) and the Documaker programs (GenTrn, GenData, GenPrint, GenWIP, and GenArc).

USING XML AND UNICODE

Documaker version 10.2 included the initial support of Unicode character set. This topic explains how you can use the Unicode capabilities with XML. Please note that XML files created from Documaker 10.2 or IDS 1.8 (and above) are written using UTF-8 encoding.

IMPORTING XML FILES

The Documaker and IDS XML parser supports the following encodings: UTF-8, ISO-8859-1, and US-ASCII. The input XML file must be using one of these encodings or should not specify an encoding at all. Here is an example of an XML header that specifies UTF-8 encoding:

```
<?xml version="1.0" encoding="UTF-8" ?>
```

If no encoding is specified, then an encoding of ISO-8859-1 is assumed.

Encoding standards

Here is a description of the encoding standards supported by the Documaker and IDS XML parser.

Use	When the XML document contains
US-ASCII	Only ASCII characters (code points 32-126). ASCII characters are the typeable characters found on a U.S. English keyboard.
ISO-8859-1	A subset of the characters in the Windows code page 1252 (ASCII characters at code points 32-126, plus additional characters at code points 160-255).
UTF-8	Any character supported by Unicode. Unicode 3.0 supports over 49,000 characters.

Memory representation

Use this information if you have to write Documaker or IDS code and you need to know how data is presented in memory on the XML tree.

- The data on XML tree is always stored in UTF-8 format. The Documaker rules or any other code has to do a conversion into the appropriate format, either Unicode or the ANSI character set, based on the destination format you need.
- The code that uses Skywire Software libraries to deal with the Unicode characters should be passing in the flag to create XML text (DXMFLAG_WIDECHAR). If this flag is passed in, the XML library converts the Unicode string into a UTF-8 encoded string.
- The code that needs to create XML text from already UTF-8 encoded string should use the DXMFLAG_UTF8 flag to create text. When this flag is used, no conversion occurs.
- The code that has to create text from an ANSI string should not set any flags. The CreateText API takes care of converting strings into a UTF-8 representation in memory.
- When getting text from XML library, the calling application should do its own conversion from UTF-8 into the appropriate format, either ANSI or Unicode.

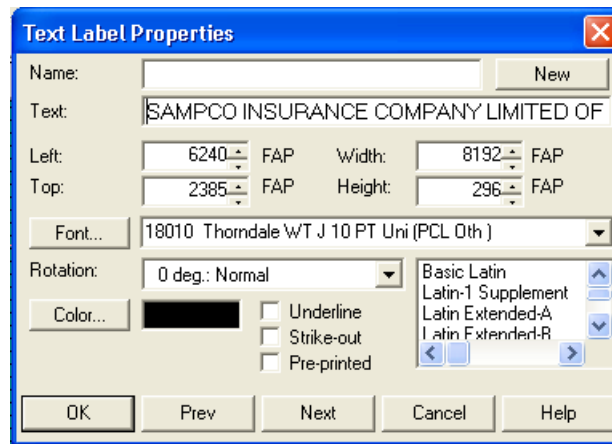
Skywire Software may add an API or a parameter to the existing API to help with this conversion, but for now you must handle the conversion. There are a couple of APIs in UTLlib, UTLUtf8ToAnsi and Utf8ToUtf16, that convert a UTF-8 string into ANSI characters and Unicode characters.

FREQUENTLY ASKED QUESTIONS

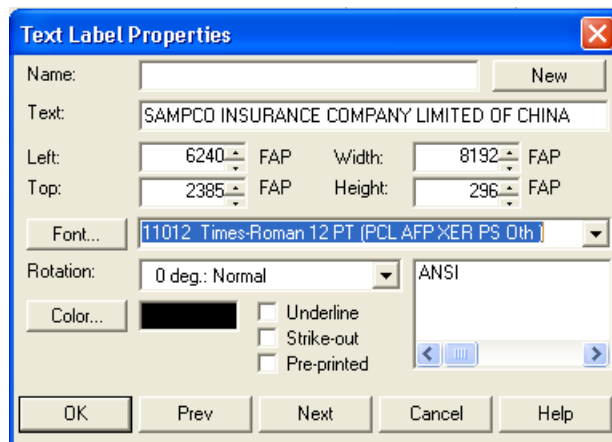
Why does text appear jumbled or change to question marks after you change the font ID?

When you create a text label with a Unicode-enabled font ID, the text label contains two-byte Unicode characters, even if the text is English. Font IDs that are not Unicode-enabled do not contain character width information for Unicode characters. Therefore, if you use the Edit, Font Change option in the Image Editor to change the font ID to a font ID that is not Unicode-enabled, the Unicode characters will have no width and will appear jumbled on top of one another.

If you change the font ID from within the Text Label window to a font ID that is not Unicode-enabled, the system tries to convert the text from Unicode to normal ANSI characters. For example, the following text label starts out as Unicode text because it uses a Unicode enabled font ID:



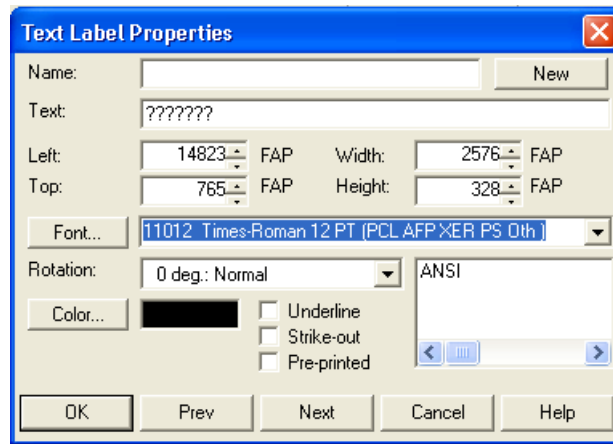
Now if you change the font ID to a normal (non-Unicode) font ID, the text is changed to ANSI characters (notice that the characters *Uni* do not appear in the font description and the list box only contains the word, *ANSI*):



However, if you tried this with a text label containing Chinese Unicode text, the text would be lost when switching to a non-Unicode font ID:



Watch what happens when you change the font ID to a normal font ID:



At this point, the Chinese text is lost unless you close the window. The system tried to convert the Unicode Chinese text to ANSI characters but of course, it could not. At this point, the text label actually consists of ANSI question marks (???????). Changing the font ID back to a Unicode-enabled font ID just results in Unicode question marks.

Index

A

AFP print drivers 3
AGFA Monotype 10
Andale Mono WT font 11
Arabic 3
archives 25

B

Big5 10

C

Chinese 2
code pages 2
comment scripts 24
CreateText 27

D

DAP archives 25
Double Byte Character Set 2
downloadable font files 24
DownloadFonts option 24

F

font cross-reference files
 building 12
Font File field 23
font IDs 28
 for variable fields 20
Font Properties Extension utility 10

G

GDI print driver 23

H

Hebrew 3

I

input method editors 6, 7
ISO-8859-1 27

J

Japanese 2
Johab 10

K

Korean 2

L

LoadCordFAP option 21

M

master resource library
 samples 25
Metacode print drivers 3
Module option 24
Move_It rule 21
Multi-Byte Character Sets 2

O

overlays 24

P

paper trays 24
parser 27
PCL
 PCL 6 driver 3
 printer drivers 23
PDF print driver 25
PostScript print drivers 3
PRC 10
PrintFunc option 24

S

ShiftJIS 10
Single Byte Character Sets 2
Solaris 3
staple bins 24
Symbol 10

T

thick-clients 3
thin-clients 3
Thorndale WT J 10, 11
TrueType fonts
 building an FXR file 12
 installing 11
 Unicode information 10

U

Unicode
 defined 2
 printing 23
 sample MRL 25
 supported platforms 3, 4
US-ASCII 27
UTF-8 2
 XML files 27
Utf8ToUtf16 27
UTLUtf8ToAnsi 27

V

variable fields 20
Vietnamese 2

W

Wansung 10
Windows
 print drivers 3, 23
 supported platforms 3, 4
WorldType fonts 10

X

XML
 and Unicode 2
 importing 27

