

# XF Rendering Server 2007 - Programmers Reference

Copyright© 2002-2006 ECRION Software. All Rights Reserved.

This document is designed to help programmers to develop custom solutions using XF Rendering Server 2007. Please contact Technical Support at [support@ecrion.com](mailto:support@ecrion.com) if you need additional information about this product.

## Table of Contents

About XF Rendering Server 2007 .....	3
Product Features .....	3
Introduction .....	3
Requirements .....	3
Feedback and Support .....	3
Other Resources .....	4
COM+ Programmers Reference .....	5
Summary .....	5
Interface IXFRenderer .....	5
baseUrl .....	6
encoding .....	6
inputFormat .....	7
logFormat .....	7
outputFormat .....	7
pageCount .....	8
pageWidth .....	8
pageHeight .....	8
renderedImagesBaseUrl .....	8
renderedImagesFolder .....	9
renderedImages .....	9
timeout .....	9
exportTo .....	9
load .....	9
loadUrl .....	10
printTo .....	10
render .....	11
renderUrl .....	12
setProperty .....	12
Interface IImageItem .....	14
name .....	14
stream .....	14
Interface IImageCollection .....	14
Count .....	14
Item .....	14
_NewEnum .....	15
Appendix A: Configuring PCL Rendering .....	16
Configuring a Windows Driver for PCL rendering .....	16
PCL Configuration in XF Management Console .....	16

Last updated: August 2006

**Important Notice:** This document and the information within is furnished "as is" and is subject to change without notice. In no event shall the author be liable for any damages whatsoever (including, without limitation, damages for loss of business profits, business interruption, loss of business information, or any other pecuniary loss) arising out of the use of or inability to use this product, even if the author has been advised of the possibility of such damages.

This PDF document was generated using XF Rendering Server 2007. For the latest version, visit [Technical Resources](#) section on our web site.

## About XF Rendering Server 2007

**XF Rendering Server 2007** is a highly scalable, enterprise class rendering product. It can be used to automate the creation of electronic documents like technical manuals, brochures, proposals, business reports containing charts and graphs, by dynamically generating them from XML.

**XF Rendering Server 2007** supports two major industry standards: XSL-FO (Extensible Style Language Formatting Objects) describing how an XML document should be formatted for a variety of media as well as SVG (Scalable Vector Graphics) used to describe two-dimensional vector and mixed vector/raster graphics in XML.

In addition high quality, all-purpose charts can be generated directly from XML using **xChart** XML language. More information can be found in [xChart 1.0 Language Reference](#).

### Product Features

- Supports XSL-FO, SVG, xChart as input.
- Produces PDF, Postscript, HTML, GIF, JPEG, PNG, BMP and other formats.
- Supports TrueType and Postscript Type1 font embedding.
- Scalable server architecture that can run across multiple CPUs, meeting the high-performance needs of your applications.
- Is accessible from a multitude of development environments: C++, VB, ASP, .NET, Java.
- Includes XF Designer 2004 XSL-FO authoring tool.

## Introduction

This reference describes COM+ components and interfaces exposed by XF Rendering Server 2007.

This SDK is not intended to be an in-depth tutorial on XSL-FO, SVG or on any related standards, and does not replace the World Wide Web Consortium (W3C) specifications for these standards. The Other Resources section below provides links to these standards on the Internet.

Among the core services, XF Rendering Server 2007 provides, is developer support for the following:

- Programatically convert XSL-FO, SVG and xChart to PDF as well as other output formats using the COM+ library, a set of interfaces designed to be used in C++ and ASP environments.
- Generate PDF documents directly in Web Server's output stream.

### Requirements

XF COM+ library can be used in the Microsoft Windows® 2000, Windows XP and Windows Server 2003 environments.

- Minimum Intel Pentium III, AMD Athlon 500 MHz or better. Intel Pentium IV 2.4 GHz recommended for development computers, dual XEON 3.0 GHz for production servers.
- Minimum 128 Mb RAM, 512 Mb recommended for development computers, 1Gb for productions servers.
- The server product must be installed. Please consult [XF Rendering Server Users Guide](#) for the install procedure

### Feedback and Support

Send error reports, feature requests, and comments about the XF documentation or samples directly to the [Technical Support team](#).

Further information about support options can be found on the [Ecrion Web site](#).

## Other Resources

[XML Recommendation](#)

[XSLT Recommendation](#)

[XSL-FO Recommendation](#)

[SVG Recommendation](#)

[XF Rendering Server Users Guide](#)

[xChart 1.0 Language Reference](#)

[XF Tutorial and QuickStarts](#)

[XF.NET Tutorial and QuickStarts](#)

## COM+ Programmers Reference

### Summary

The following table lists the core objects/interfaces:

Interface/Class	Description
<a href="#">IXFRenderer</a>	XSL-FO, SVG and xChart rendering engine. Provides <a href="#">render</a> and <a href="#">renderUrl</a> methods that can render a document in one of the supported output formats.
<a href="#">IImageCollection</a>	Collection of IImageItem. Used by IXFRenderer's <a href="#">renderedImages</a> property to hold streams for all generated images when output format is HTML.
<a href="#">IImageItem</a>	A generated image, when output format is HTML.

**Implementation:** XFCom.dll

**Header File:** XFCom.h

**Type Library File:** XFCom.tlb

Please note that the easiest way to use XF when using the Microsoft compiler is to actually import the type library (placed under %WINDIR%/System32 by the installer), as shown below:

```
#import "XFCom.tlb" rename_namespace("XF")

// ...
XF::IXFRendererPtr    renderer(__uuidof(XF::IXFRenderer));

renderer->renderUrl(srcFile, destFile);
```

### Interface IXFRenderer

**IXFRenderer** represents a single instance of a renderer. Use-it to convert XSL-FO, SVG, or xChart into PDF, HTML, GIF, PNG, JPG, etc.

#### Properties

Name	Description
<a href="#">baseUrl</a>	The base URL to be used when computing relative paths. Write-only
<a href="#">encoding</a>	Overwrites input's encoding. Write-only.
<a href="#">inputFormat</a>	Contains input's format. Read/write.
<a href="#">pageCount</a>	Gets the number of pages in a loaded document. Read-only.
<a href="#">pageWidth</a>	Gets the width of a specified page in a loaded document. Read-only.
<a href="#">pageHeight</a>	Gets the height of a specified page in a loaded document. Read-only.

Name	Description
<a href="#">logFormat</a>	Specifies the format in which the log generated by a rendering job is formatted. Read/write.
<a href="#">log</a>	Contains the log generated by a rendering job. Read-only.
<a href="#">outputFormat</a>	Contains output's format. Read/write.
<a href="#">renderedImagesBaseUrl</a>	Specifies the base URL used to compose URLs for generated images, when output format is HTML. Read/Write.
<a href="#">renderedImagesBaseFolder</a>	Specifies physical location for generated images, when output format is HTML. Read/Write.
<a href="#">renderedImages</a>	A collection of streams for all generated images in the document, when output format is HTML. Read-only.
<a href="#">timeout</a>	The time in seconds that the rendering can not exceed. Read/Write.

### Methods

Name	Description
<a href="#">exportTo</a>	Render a loaded document.
<a href="#">getProperty</a>	Get various rendering parameters.
<a href="#">load</a>	Load a document from a string or a stream.
<a href="#">loadUrl</a>	Load a file from the specified location.
<a href="#">printTo</a>	Prints the loaded document.
<a href="#">render</a>	Shortcut method for loading and rendering a document. The input can be a string or a stream.
<a href="#">renderUrl</a>	Shortcut method for loading and rendering a document. The input must be a file.
<a href="#">setProperty</a>	Set various rendering parameters.

### baseUrl

Write-only BSTR property that represents the URL of the current XML document. This property is very useful when the input is provided as XML text or IStream.

The renderer uses this base URL to resolve relative links. To resolve these links, the reader uses the base URL in much the same way that an Internet browser uses a Web page's URL to resolve relative links contained on that page. The base URL may be a complete URL. However, the reader only uses that portion of the URL up to and including the final backslash when resolving links. For instance, a base URL for a document is "C:\Documents\XSL-FO\test.fo". The reader would then only use "C:\Documents\XSL-FO\" when resolving links. The same result would be obtained if you specify "C:\Documents\XSL-FO\".

As a final note the base URL can also be a http or https type of URL. In this case, if you specify only the folder of the current document you have to include a final backslash.

### encoding

Write-only BSTR property that can be used to override input's encoding. If this property is not set, the engine will use the encoding specified in the <xml> declaration, or if not present, it will try to autodetect-it.

**inputFormat**

Read/write property containing input's format. Can take one of the following values:

XIF_Auto = 1	Input format is detected automatically. This is the default value.
XIF_Svg = 2	Input is an SVG document.
XIF_XslFo = 3	Input is an XSL-FO document.
XIF_XChart = 4	Input is an <a href="#">xChart</a> document.
XIF_WordML = 10	Microsoft Word document in XML format.

**logFormat**

Read/write property containing the format in which the rendering's job log is generated. Can take one of the following values:

XLF_PlainText = 1	The log will be formatted as plain text. Each line contains a message generated by the server, together with the severity (Information, Error, Warning).
XLF_Xml = 2	The log will be formatted as XML.

**outputFormat**

Read/write property containing input's format. Can take one of the following values:

XOF_Bitmap = 1	Windows Bitmap.
XOF_Gif = 2	GIF89
XOF_Png = 3	Portable Network Graphics
XOF_Jpeg = 4	JPEG
XOF_TIFF = 10	TIFF.
XOF_PDF = 5	Adobes' PDF (Portable Document Format).
XOF_HTML = 7	HTML (Hypertext Markup Language).
XOF_TXT = 15	TEXT (Unicode or ASCII text).
XOF_SVG = 9	Scalable Vector Graphics.
XOF_PostScript = 8	PostScript.
XOF_PCL = 11	PCL. XF can render PCL in two modes, as described in <a href="#">Appendix A</a> .
XOF_AFP = 13	Advanced Function Presentation (IBM High-Volume Printing Solutions).
XOF_Layout = 12	XML dump of a XSL-FO area tree.
XOF_ZipLayout = 14	Zipped XML dump of a XSL-FO area tree.
XOF_Default = 6	If the output is a file, the output format can be determined from the output file's extension. If the extension is not recognized or if the output's destination is not a file, an exception is thrown, that is, the output format must be specified explicitly.

This is the default value.
----------------------------

**pageCount**

Read-only integer property that returns the number of pages in a document. The document must be loaded previously by using either [load](#) or [loadUrl](#).

**pageWidth**

Read-only integer property that returns the width of a specified page in pixels (1/96 inch units). The document must be loaded previously by using either [load](#) or [loadUrl](#). The following example shows how to generate a thumbnail for the first page in a document.

```
void renderUrlToThumbnail(bstr_t inputFile, bstr_t outputFile)
{
    try
    {
        IXFRendererPtr      eng(__uuidof(XFRenderer));

        eng->loadUrl(inputFile);

        if (eng->pageCount > 0)
        {
            double width = eng->pageWidth[0];
            double height = eng->pageHeight[0];
            double zoom;
            double thumbSize = 256;           // the thumbnail will be 256
                                              // pixels width and 256
                                              // pixels height

            if (thumbSize/width < thumbSize/height)
                zoom = thumbSize/width*100.;
            else
                zoom = thumbSize/height*100.;

            eng->setProperty(L"Compression", 100);
            eng->setProperty(L"Zoom", zoom);
            eng->setProperty(L"Pages", 0);      // To render multiple pages,
                                              // use a string containing the
                                              // specific pages indexes:
                                              // eng->setProperty(L"Pages", L"2 4");

            eng->exportTo(outputFile);
        }
    }
    catch (_com_error& e)
    {
        if ((LPCWSTR)e.Description())
            printf("Error: %S\n", (LPCWSTR)e.Description());
        else
            printf("Error: %S\n", (LPCWSTR)e.ErrorMessage());
    }
}
```

**pageHeight**

Read-only integer property that returns the height of a specified page in pixels (1/96 inch units). The document must be loaded previously by using either [load](#) or [loadUrl](#).

**renderedImagesBaseUrl**

When the output is HTML, this property specifies the base URL used to compose the SRC value for generated images. For example, if your document includes an SVG or XChart graphic, and you specify "images/" for this property the renderer will generate a IMG tag, with SRC="image/EDD8EC08-7256-48aa-972A-ED67080E3AD0.jpg (the file name will be always unique)". You can also specify a more complex URL, like

"https://www.mydomain.com/imgProvider.aspx?sessionId=AA32DE&fileName=".

The generated IMG element will have SRC="https://www.mydomain.com/imgProvider.aspx?sessionId=AA32DE&fileName=EDD8EC08-7256-48aa-972A-ED67080E3AD0.jpg".

### renderedImagesFolder

When the output is HTML, this property specifies physical location where images generated from SVG or XChart elements must be placed. The folder must exist. If this property is not set, the renderer will create a stream for each generated image and store it in renderedImageStreams.

### renderedImages

When the output is HTML, and renderedImagesFolder is not set, the images generated from SVG or XChart are placed in this collection. The name of each image is a unique name (for example EDD8EC08-7256-48aa-972A-ED67080E3AD0.jpg). You can use the streams to return the images to the client directly from memory, or you can save them in a preferred location (although for this you should use renderedImagesFolder).

### timeout

Read/Write integer property representing the maximum amount of time (in seconds) that the rendering should take. This property can be used to prevent the server being blocked by large jobs.

The initial value is 0, that is, the rendering will never be interrupted.

### exportTo

Renders the loaded document.

```
HRESULT exportTo(VARIANT dest);
```

### Parameters

dest

The **file path** where the renderer should place the output (variant type VT\_BSTR), or an **IStream** (variant type VT\_UNKNOWN or VT\_DISPATCH), or **IResponse** (variant type VT\_UNKNOWN or VT\_DISPATCH) - the interface that exposes the methods of IIS's Reponse object.

### Return Values

S\_OK

The value returned if successful.

E\_FAIL

The value returned if the rendering fails. Use IErrorInfo to retrieve the error.

### load

Loads the supplied input. The document can be then rendered using exportTo, or printed using printTo.

```
HRESULT load(VARIANT input)
```

### Parameters

**input**

A valid XML document containing text to load. Can be a BSTR (VT\_BSTR) or an IStream (variant type VT\_IUnknown)

**Return Values****S\_OK**

The value returned if successful.

**E\_FAIL**

The value returned if the rendering fails. Use IErrorInfo to retrieve the error.

**loadUrl**

Loads the specified file. The document can be then rendered using exportTo, or printed using printTo.

```
HRESULT loadUrl(BSTR url)
```

**Parameters****url**

Path to a valid XML document containing text to load.

**Return Values****S\_OK**

The value returned if successful.

**E\_FAIL**

The value returned if the rendering fails. Use IErrorInfo to retrieve the error.

**printTo**

Prints the loaded document.

```
HRESULT printTo(BSTR dest, XFPrintMode printMode);
```

**Parameters****dest**

The name of a installed printer, as displayed in Control Panel\Printers.

**printMode**

Specifies the method used to print the document. It can take one of the following values:

XPR_Postscript = 1	First, the engine will render the document to a temporary Postscript file, then the file is copied to the printer as RAW data. The printer must support Postscript language.
XPR_PCL = 2	First, the engine will render the document to a temporary PCL file, then the file is copied to the printer as RAW data. The printer must support PCL language. Please note that most printers that support PCL also support POSTSCRIPT. Whenever possible, you should use XPR_Postscript instead of XPR_PCL because PCL rendering involves additional setup as described in <a href="#">Appendix A</a> .
XPR_GDI = 2	Prints the document using Windows GDI. Any installed printer can be used.

**Return Values**

**S\_OK**

The value returned if successful.

**E\_FAIL**

The value returned if the rendering fails. Use `IErrorInfo` to retrieve the error.

**render**

Loads the supplied input and performs the rendering in one step.

```
HRESULT render(VARIANT input, VARIANT output);
```

**Parameters****input**

A valid XML document containing text to render. Can be a `BSTR` (`VT_BSTR`) or an `IStream` (variant type `VT_IUnknown`)

**output**

The **file path** where the renderer should place the output (variant type `VT_BSTR`), or an **IStream** (variant type `VT_UNKNOWN` or `VT_DISPATCH`), or **IResponse** (variant type `VT_UNKNOWN` or `VT_DISPATCH`) - the interface that exposes the methods of IIS's Reponse object.

**Return Values****S\_OK**

The value returned if successful.

**E\_FAIL**

The value returned if the rendering fails. Use `IErrorInfo` to retrieve the error.

**Remarks**

The format of the input text is determined by [inputFormat](#) property while the format of the output is determined by [outputFormat](#) property. Both of them should be set before calling `render`.

The following example shows how to render PDF documents directly in IIS's response object:

```
<!-- WriteToResponse.asp -->

<%@ Language=VBScript %>
<%
Dim engine
Dim output

Set engine = CreateObject("XFCOM.XFRenderer")

output.contentType = "application/pdf"
engine.outputFormat = 5 'PDF
engine.render "<fo:root xmlns:fo='http://www.w3.org/1999/XSL/Format'>" & _
    "<fo:layout-master-set>" & _
    "<fo:simple-page-master master-name='default' margin='1in'>" & _
    "<fo:region-body/>" & _
    "</fo:simple-page-master>" & _
    "</fo:layout-master-set>" & _
    "<fo:page-sequence master-reference='default'>" & _
    "<fo:flow flow-name='xsl-region-body'>" & _
    "<fo:block>ASP Hello World!</fo:block>" & _
    "</fo:flow>" & _
    "</fo:page-sequence>" & _
    "</fo:root>", Response

%>
```

## renderUrl

Loads the supplied input and performs the rendering in one step.

```
HRESULT renderUrl(BSTR url, VARIANT output)
```

### Parameters

url

Path to a valid XML document containing text to render.

output

The **file path** where the renderer should place the output (variant type VT\_BSTR), or an **IStream** (variant type VT\_UNKNOWN or VT\_DISPATCH), or **IResponse** (variant type VT\_UNKNOWN or VT\_DISPATCH) - the interface that exposes the methods of IIS's Reponse object.

### Remarks

The format of the input text is determined by [inputFormat](#) property while the format of the output is determined by [outputFormat](#) property. Both of them should be set before calling render.

### Return Values

S\_OK

The value returned if successful.

E\_FAIL

The value returned if the rendering fails. Use IErrorInfo to retrieve the error.

## setProperty

Set various rendering parameters.

```
HRESULT setProperty(BSTR propertyName, VARIANT newVal);
```

### Parameters

propertyName

The name of the property you wish to change. For a list of valid property names see "Remarks" section.

newVal

The new value for this property. The variant type can be VT\_INT or VT\_BSTR.

### Return Values

S\_OK

The value returned if successful.

E\_FAIL

The value returned if the rendering fails. Use IErrorInfo to retrieve the error.

### Remarks

Name	Description
ColorDepth	The number of bits used to describe the color of a single pixel, when the output is a raster image format (XOF_GIF, XOF_JPEG, etc). The value can be integer value 1 or the string "Monochrome" for 1 bit monochrome images, 8 or "Indexed" for 256 colors images, -8 or "GrayscaleIndexed" for 256 colors

Name	Description
XResolution YResolution	grayscale images, and finally, 24 or "TrueColor" for true color images (also the default value).
Compression	<p>When output format is <i>TIFF</i>, this can take the following values: "NONE" (the default), "RLE", "LZW", "CCITT3", "CCITT4". "CCITT3" and "CCITT4" can be used only for monochrome images.</p> <p>When output format is <i>JPEG</i>, this is an integer value between 0 and 100.</p>
JobPrinterDriver	<p>When output format is PCL, you can specify which printer driver must be used to generate PCL. For information on how to setup a printer driver to be used by XF, read <a href="#">Appendix A</a>.</p> <p>This property performs an override of the global settings configured though XF Rendering Server 2007 Management Console, and it will affect only the current rendering job.</p>
Antialiasing	<p>When output format is a raster format (GIF, JPEG, PNG, etc), you can specify the process used to render text.</p> <p>It can take one of the following values:</p> <ul style="list-style-type: none"> <li>• <i>"SystemDefault"</i> - Specifies that a character is drawn using the currently selected system font smoothing mode. This is the default value.</li> <li>• <i>"SingleBitPerPixelGridFit"</i> - Specifies that a character is drawn using its glyph bitmap and hinting to improve character appearance on stems and curvature.</li> <li>• <i>"SingleBitPerPixel"</i> - Specifies that a character is drawn using its glyph bitmap and no hinting.</li> <li>• <i>"AntiAliasGridFit"</i> - Specifies that a character is drawn using its antialiased glyph bitmap and hinting. This results in much better quality due to antialiasing.</li> <li>• <i>"AntiAlias"</i> - Specifies that a character is drawn using its antialiased glyph bitmap and no hinting. Stem width differences may be noticeable because hinting is turned off.</li> <li>• <i>"ClearTypeGridFit"</i> - Specifies that a character is drawn using its glyph ClearType bitmap and hinting.</li> </ul> <p><b>Windows XP and Windows .NET Server only:</b> ClearType rendering is supported only on Windows XP and Windows .NET Server.</p>
InterpolationMode	<p>When output format is a raster format (GIF, JPEG, PNG, etc), you can specify the algorithm used to calculate the intermediate values between two endpoints when scaling images.</p> <p>It can take one of the following values:</p> <ul style="list-style-type: none"> <li>• <i>"LowQuality"</i> - Specifies low quality interpolation. The fastest scaling method.</li> <li>• <i>"HighQuality"</i> - Specifies high quality interpolation.</li> <li>• <i>"Bilinear"</i> - Specifies bilinear interpolation.</li> </ul>

Name	Description
	<ul style="list-style-type: none"> <li>"<i>Bicubic</i>" - Specifies bicubic interpolation.</li> <li>"<i>NearestNeighbor</i>" - Specifies nearest-neighbor interpolation.</li> <li>"<i>HighQualityBilinear</i>" - Specifies high quality bilinear interpolation.</li> <li>"<i>HighQualityBicubic</i>" - Specifies high quality bicubic interpolation. This is the default value; it is the slowest scaling method, but it produces the best results.</li> </ul>

## Interface IImageItem

**IImageItem** represents a rendered image when output format is HTML. A rendering job can produce multiple images that will be stored in [renderedImages](#) collection.

### Properties

Name	Description
<a href="#">name</a>	Contains the unique name of the image. Read-only.
<a href="#">stream</a>	IStream containing image's bytes. Read-only.

#### name

Read only property containing the image name. The name is a unique GUID followed by .jpg extension.

#### stream

Read only IStream containing image's bytes.

## Interface IImageCollection

**IImageCollection** represents a collection of all rendered images when output format is HTML. See [renderedImages](#) property for more details.

### Properties

Name	Description
<a href="#">Count</a>	Returns number of images in collection. Read-only.

### Methods

Name	Description
<a href="#">Item</a>	Given an index, returns an IImageItem in the collection.
<a href="#">_NewEnum</a>	The <code>_NewEnum</code> property returns the IEnumVARIANT enumerator interface for the collection.

#### Count

Read only property containing the number of images in collection.

#### Item

Given an index, returns an IImageItem in the collection.

```
HRESULT Item([in] long Index, [out, retval] IImageItem** val);
```

#### Parameters

Index

The image index.

val

The IImageItem at the specified index.

#### Return Values

S\_OK

The value returned if successful.

#### \_NewEnum

The \_NewEnum property returns the IEnumVARIANT enumerator interface for the collection.

```
HRESULT _NewEnum([out, retval] IUnknown** val);
```

#### Parameters

val

The new enumerator.

#### Return Values

S\_OK

The value returned if successful.

## Appendix A: Configuring PCL Rendering

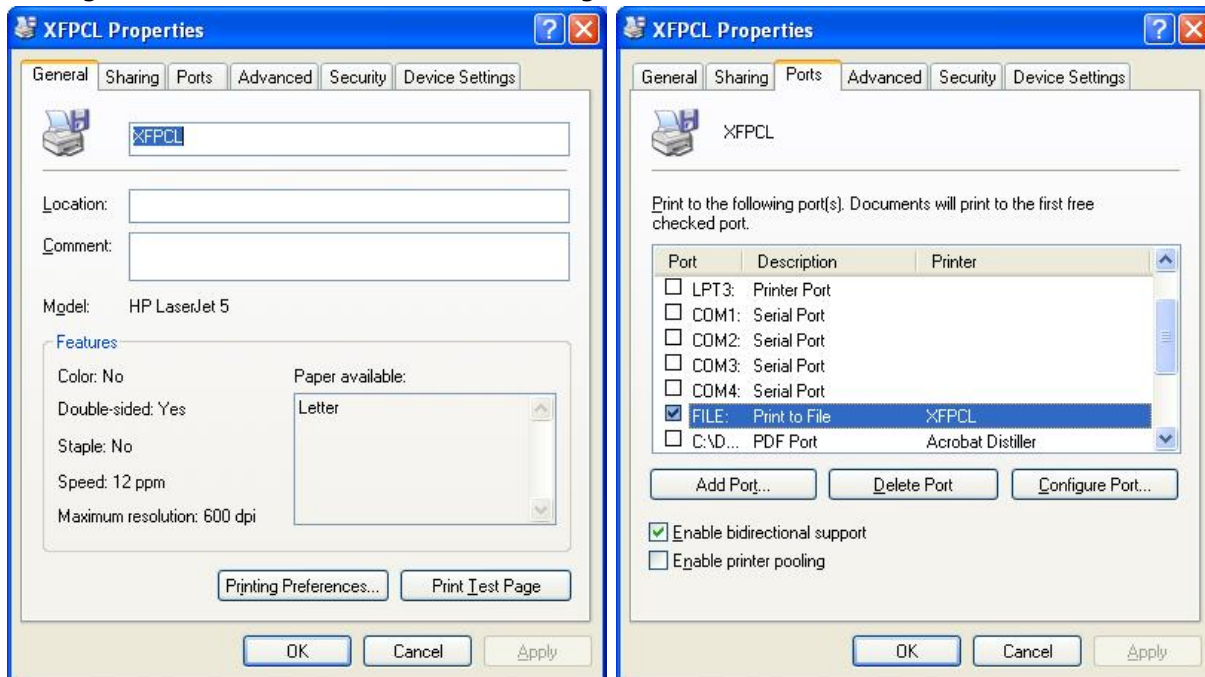
**XF Rendering Server 2007** can render PCL documents by using a printer driver, configured as described below.

### Configuring a Windows Driver for PCL rendering

First, you must install a local printer; XF will use this printer driver to generate PCL.

- Open Control Panel/Printers and double click Add Printer. Welcome to Add Printer Wizard should appear.
- Select "Local printer attached to this computer" and uncheck "Automatically detect and install my Plug and Play printer".
- Select FILE:(Print to File).
- You can select your exact printer model, or a generic one; for a generic driver, select HP from Manufacturer list, then select HP LaserJet 5. Name your printer XFPCL (any name will do, but it will be better if you can recognize-it easily). Check "No" for "Do you want to use this printer as the default printer".
- Check "Do not share this printer".
- Check "No" for "Do you want to print a test page?" question.
- Click Finish.

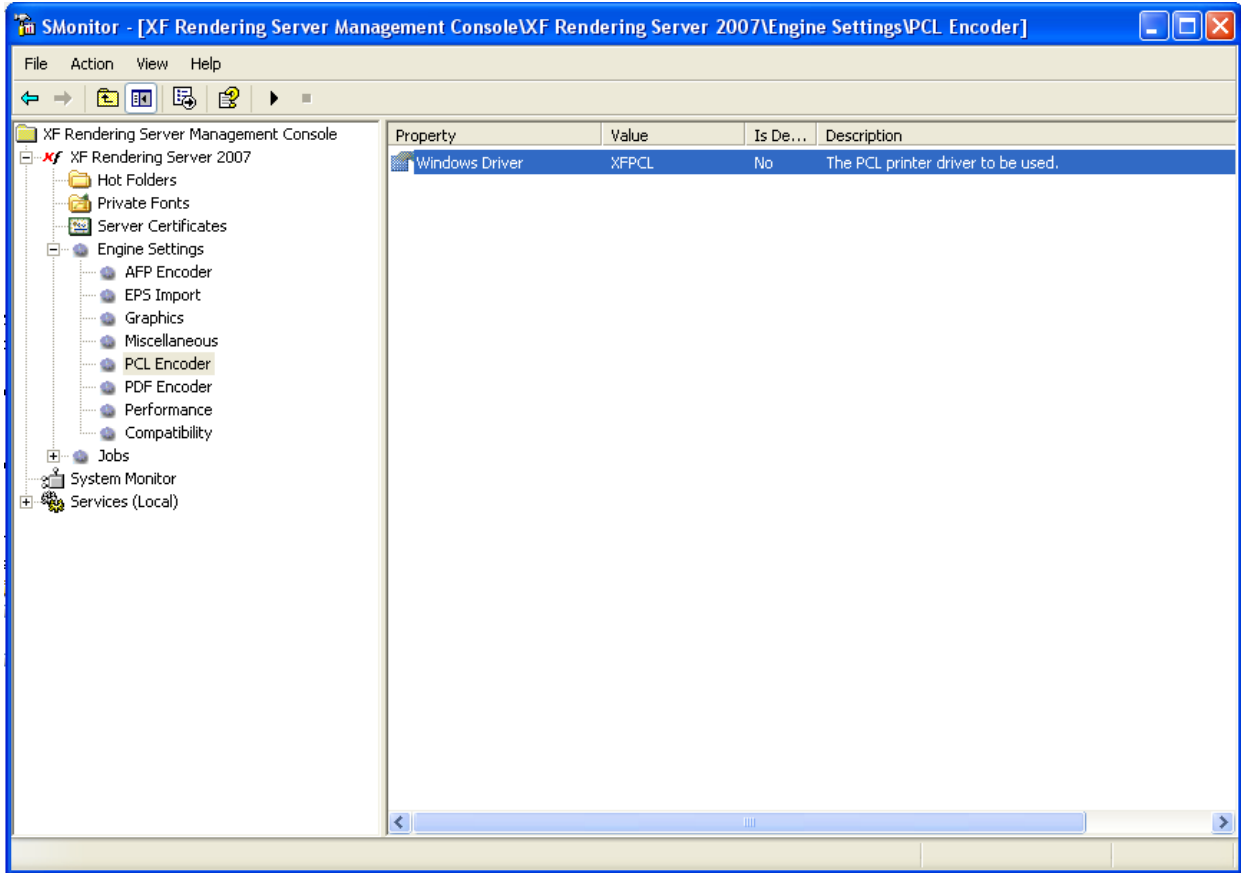
After the new printer is added in Control Panel printers, you can inspect the settings by selecting the printer icon, then choosing Properties from the contextual menu. These settings should be similar with the following screenshots:



### PCL Configuration in XF Management Console

To configure XF to use this driver:

- Open XF Management Console and go to Engine Settings/PCL Encoder.
- Set "Windows Driver" to "XFPCL".



Please note that after any changes in XF Management Console, the Windows service must be restarted to reload the new values.