

# XF Rendering Server 2007 - .NET 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
.NET Programmers Reference .....	5
Summary .....	5
Class Ecrion.XF.XFEngine .....	5
Log .....	5
InputFormat .....	6
EncodingOverride .....	6
BaseUrl .....	6
Timeout .....	6
OutputFormat .....	6
OutputMimeType .....	7
SetProperty .....	7
Render .....	9
RenderUrl .....	9
Print .....	9
PrintUrl .....	10
Class Ecrion.XF.Document .....	10
PrintTo .....	10
SendFax .....	11
ExportTo .....	11
GetPageCount .....	11
GetPageWidth .....	12
GetPageHeight .....	12
Load .....	12
LoadUrl .....	12
Class Ecrion.XF.XFMergeContext .....	12
OutputFormat .....	13
SetOutput .....	13
Class Ecrion.Windows.Forms.XFViewerControl .....	13
CurrentPage .....	14
DefaultZoom .....	14
PageCount .....	14
StatusBar .....	15
ToolBar .....	15
XFViewerControl .....	15
Relayout .....	15
LoadXml .....	15

LoadUrl	15
Print	15
SendEmail	15
Export	16
ZoomIn	16
ZoomOut	16
Zoom	16
GotoPage	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 the XF Rendering Server 2007 programming interfaces available in the .NET environment.

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.

This SDK is not intended to demonstrate every possible use of XF, and does not document every known problem with XF. See Help and Support, below, for links to support resources.

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 XF.NET component.
- Generate PDF documents directly in Web Server's output stream.

### Requirements

XF.NET 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.

### 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 SDK Tutorial and QuickStarts](#)

# .NET Programmers Reference

## Summary

The following table lists the core objects:

Class	Description
<a href="#">XFEngine</a>	Class representing a single instance of a rendering engine. Use-it to render XSL-FO, SVG or xChart documents in one single step.
<a href="#">XFDocument</a>	Class representing a single instance of a document. Use-it to manage XSL-FO, SVG or xChart documents.
<a href="#">XFMergeContext</a>	Class representing the result of multiple rendering operations. Use-it in conjunction with XFDocument to generate on large output from multiple input documents.
<a href="#">XFViewerControl</a>	XSL-FO viewer .NET control. Use-it to display XSL-FO documents in desktop applications.
<a href="#">XFEditorControl</a>	XSL-FO editor .NET control. Use-it to display and XSL-FO documents in desktop applications. It also provides basic editing capabilities, including insertions on tables and images.

## Class Ecrion.XF.XFEngine

XFEngine represents a single instance of a renderer. Use-it to render both XSL-FO, SVG or xChart documents into PDF or a raster format.

### Properties

Name	Description
<a href="#">Log</a>	The list of events occurred during rendering.
<a href="#">InputFormat</a>	Gets or sets the input format.
<a href="#">EncodingOverride</a>	Overwrites input's encoding.
<a href="#">BaseUrl</a>	Gets or sets the base URL to be used when computing relative paths.
<a href="#">Timeout</a>	Gets or sets timeout period assigned to the rendering job, in seconds.
<a href="#">OutputFormat</a>	Gets or sets the output's format.
<a href="#">OutputMimeType</a>	Provides output's MIME format.

### Methods

Name	Description
<a href="#">SetProperty</a>	Set various rendering parameters.
<a href="#">Render</a>	Render a document using the supplied input. Overloaded.
<a href="#">RenderUrl</a>	Render the document from the specified location. Overloaded.
<a href="#">Print</a>	Prints the document using the supplied input. Overloaded.
<a href="#">PrintUrl</a>	Prints the document from the specified location.

### Log

The list of events occurred during rendering.

## InputFormat

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

InputFormat.AUTO	Specifies that the input format should be auto-detected. This is the default value for the input format.
InputFormat.SVG	The XML input is in SVG (Scalable Vector Graphics) format.
InputFormat.XSLFO	Input is an XSL-FO document.
InputFormat.XCHART	Input is an <a href="#">xChart</a> document.
InputFormat.WordML	Input is Microsoft Word document in XML format.

## EncodingOverride

Write-only 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.

## BaseUrl

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

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 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.

## 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.

## OutputFormat

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

OutputFormat.BITMAP	Windows Bitmap.
OutputFormat.GIF	GIF89
OutputFormat.PNG	Portable Network Graphics
OutputFormat.JPEG	JPEG
OutputFormat.TIFF	TIFF.
OutputFormat.PDF	Adobes' PDF (Portable Document Format).
OutputFormat.HTML	Specifies that the output format must be HTML.

OutputFormat.TEXT	TEXT (Unicode or ASCII text).
OutputFormat.SVG	Scalable Vector Graphics.
OutputFormat.POSTSCRIPT	Specifies that the output format must be PostScript. You should always render to a file instead of a stream because PostScript files can be very large. This will preserve server's memory and result in a more scalable application.
OutputFormat.PCL	PCL.
OutputFormat.AFP	Advanced Function Presentation (IBM High-Volume Printing Solutions).
OutputFormat.DEFAULT	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.

### OutputMimeType

Read-only property that provides output's MIME format.

### SetProperty

Set various rendering parameters.

```
void SetProperty(String propertyName, String newVal);
void SetProperty(String propertyName, int newVal);
```

### Parameters

propertyName

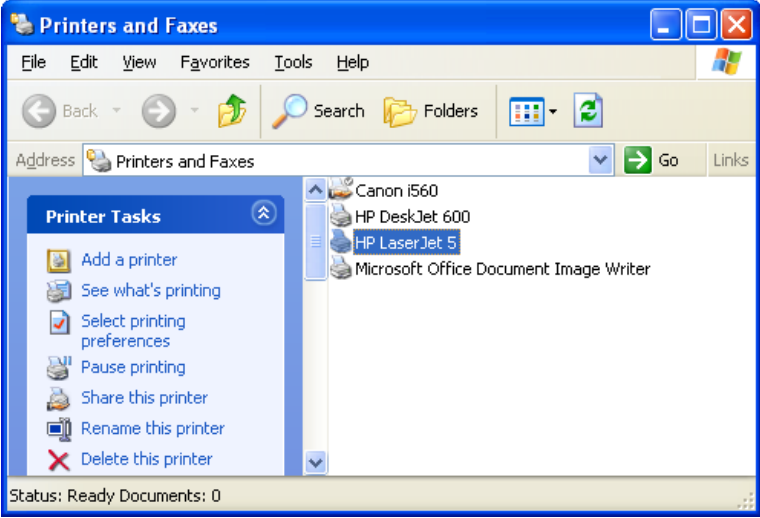
The name of the property you wish to change. For a list of valid property names see "Remarks" section.

newVal

The new value for this property.

### 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 (GIF, JPEG, etc). The value can be 1 (Monochrome), 8 (256 Colors) or 24 (True Color, also the default value).
XResolution YResolution	Integer property describing the output image's resolution in DPI (Dots Per Inch). The default value for both resolutions is 96.
Compression	When output format is <i>TIFF</i> , this can take the following values: "NONE" (the default), "RLE", "LZW", "CCITT3", "CCITT4".  When output format is <i>JPEG</i> , this is an integer value between 0 and 100.

Name	Description
JobPrinterDriver	<p>When output format is PCL, you can specify which printer driver must be used to generate PCL. For example, considering the following setup:</p>  <p>You can specify "HP LaserJet 5" or "HP DeskJet 600" for this property.</p> <p>Note: A Windows printer driver can be installed without actually having a printer connected. This property performs an override of the global settings configured though XF Rendering Server 2007 Management Console. 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</p>

Name	Description
	and Windows .NET Server.

## Render

Render a document using the supplied input.

```
void Render(String xml, String outputFileName)
void Render(String xml, Stream outputStream)
void Render(Stream inputStream, String outputFileName)
void Render(Stream inputStream, Stream outputStream)
```

### Parameters

xml

A valid XML document containing text to render.

inputStream

A stream containing XML text to render.

outputFileName

The file path where the renderer should place the output.

outputFileStream

The destination stream to for rendering's output.

## RenderUrl

Render the document from the specified location.

```
void RenderUrl(String url, String outputFileName)
void RenderUrl(String url, Stream outputStream)
```

### Parameters

url

Path to a valid XML document containing text to render.

outputFileName

The file path where the renderer should place the output.

outputFileStream

The destination stream to for rendering's output.

## Print

Prints a document using the supplied input.

```
public void Print(string xml, string printerName, PrintMode mode)
public void Print(System.IO.Stream inputStream, string printerName, PrintMode mode)
```

### Parameters

xml

A valid XML document containing text to print.

inputStream

A stream containing XML text to print.

printerName

The name of the printer that should print the document.

PrintMode

The mode in which printing would be made. It supports three modes GDI, PCL, POSTSCRIPT

## PrintUrl

Prints the document from the specified location.

```
public void PrintUrl(string url, string printerName, PrintMode mode)
```

### Parameters

url

A valid location containing text to print.

printerName

The name of the printer that should print the document.

PrintMode

The mode in which printing would be made. It supports three modes GDI, PCL, POSTSCRIPT

## Class Ecrion.XF.Document

XFDocument represents a single instance of a document. Use-it to render both XSL-FO, SVG or xChart documents into PDF or a raster format.

### Properties

Name	Description
<a href="#">Log</a>	See the documentation for XFEngine.
<a href="#">InputFormat</a>	See the documentation for XFEngine.
<a href="#">EncodingOverride</a>	See the documentation for XFEngine.
<a href="#">BaseUrl</a>	See the documentation for XFEngine.
<a href="#">Timeout</a>	See the documentation for XFEngine.
<a href="#">OutputFormat</a>	See the documentation for XFEngine.
<a href="#">OutputMimeType</a>	See the documentation for XFEngine.

### Methods

Name	Description
<a href="#">SetProperty</a>	See the documentation for XFEngine.
<a href="#">PrintTo</a>	Prints a document previously loaded.
<a href="#">SendFax</a>	Send a fax with the document previously loaded.
<a href="#">ExportTo</a>	Exports the document previously loaded. Overloaded.
<a href="#">GetPageCount</a>	Returns the number of pages for a loaded document.
<a href="#">GetPageWidth</a>	Returns the width of a page in pixels.
<a href="#">GetPageHeight</a>	Returns the height of a page in pixels.
<a href="#">Load</a>	Loads the XML document contained in the given context. Overloaded.
<a href="#">LoadUrl</a>	Loads the XML document specified by the given url.

## PrintTo

Prints a document using previously loaded with load or loadUrl.

```
public void PrintTo(string printerName, PrintMode mode)
```

### Parameters

printerName

The name of the printer that should print the document.

PrintMode

The mode in witch printing would be made. It supports three modes GDI, PCL, POSTSCRIPT

## SendFax

Send a fax with the document previously loaded.

```
public void SendFax(FaxSender faxSender, FaxReceiver faxRecipient)
```

### Parameters

faxSender

The name of the sender and the name of the company.

faxRecipient

The fax number that is mandatory and the receiver's name.

## ExportTo

Exports the document previously loaded. Output format can be recognized from the output file's extension when the export context is a file. When the export context is a stream the OutputFormat must be set before calling this function.

```
public void ExportTo(string outputFileName)
public void ExportTo(Stream outputStream)
public void ExportTo(XFMergeContext ctx)
```

### Parameters

outputFileName

Name of the file to receive rendering output.

outPutStream

Stream to receive rendering output.

ctx

A merge context that will receive the output.

## GetPageCount

Returns the number of pages for a loaded document.

```
public int GetPageCount()
```

### GetPageWidth

Returns the width of a page in pixels.

```
public double GetPageWidth(int idx)
```

#### Parameters

idx  
Page index

### GetPageHeight

Returns the height of a page in pixels.

```
public double GetPageHeight(int idx)
```

#### Parameters

idx  
Page index

### Load

Loads the XML document contained in the given context.

```
public void Load(string xml)  
public void Load(Stream inputStream)
```

#### Parameters

xml  
Load the XML document contained in the given string.  
inputStream  
Load the XML document contained in the given stream.

### LoadUrl

Load the XML document specified by the given url

```
public void LoadUrl(string url)
```

#### Parameters

url  
Path to document to be loaded.

### Class Ecrion.XF.XFMergeContext

XFMergeContext can be used to render multiple input documents into one output file or stream. This feature can very useful for printing jobs which require all inputs to be sent to a printer into one large file Postscript file. It can also be used for archiving purposes.

The following code fragment exemplifies this feature:

```

using Ecrion.XF;

static void mergeOutputs(string[] inputFiles, string outputFile,
                        Ecrion.XF.OutputFormat fmt)
{
    using(XFMergeContext ctx = new XFMergeContext())
    {
        ctx.OutputFormat = fmt;
        ctx.SetOutput(outputFile);

        for each(string inFile in inputFiles)
        {
            using (XFDocument doc = new XFDocument())
            {
                doc.LoadUrl(inFile);
                doc.ExportTo(ctx);
            }
        }
    }
}

```

You can also use a `System.IO.Stream` object as parameter to `setOutput` method.

Please note that the merge context must be closed gracefully in order to obtain a valid output file.

The close operation is automatically performed by `XFMergeContext`'s `IDisposable` implementation, that is why in the example above, we used 'using' statement to define the scope at the end of which the merge context is disposed.

#### Properties

Name	Description
<a href="#">OutputFormat</a>	Gets or sets the output's format.

#### Methods

Name	Description
<a href="#">SetOutput</a>	Sets the output file or stream.

### OutputFormat

Sets or gets the output format of the merged output.

### SetOutput

Sets the output file or stream

```

void SetOutput(string url)
void SetOutput(Stream outputStream)

```

#### Parameters

url

The complete path of the file that will hold the output.

outputStream

The `System.IO.Stream` object that will hold the output.

## Class Ecrion.Windows.Forms.XFViewerControl

Ecrion XSL-FO Viewer .NET Control is a component that allows displaying of XSL-FO documents in a .NET windows application. It includes support for exporting the document in PDF format, sending the PDF document attached in an email, printing, etc. A sample application is included in the distribution.

### Properties

Name	Description
<a href="#">CurrentPage</a>	First visible page in the document. Read only.
<a href="#">DefaultZoom</a>	Default zoom factor to be used when opening a new document. Read/Write.
<a href="#">IsDocumentLoaded</a>	True if a document is loaded in the viewer. Read only.
<a href="#">PageCount</a>	Number of pages in this document. Read only.
<a href="#">StatusBar</a>	The status bar displayed by the control. Read only.
<a href="#">ToolBar</a>	The tool bar displayed by the control. Read only.

### Methods

Name	Description
<a href="#">XFViewerControl</a>	Create a new instance of XFViewerControl.
<a href="#">Relayout</a>	Forces layout after repositioning the toolbar or status bar.
<a href="#">LoadXml</a>	Load and display the specified XSL-FO text. Overloaded.
<a href="#">LoadUrl</a>	Load and display an XSL-FO document from the specified location. Overloaded.
<a href="#">Print</a>	Display "Print" dialog and print the document.
<a href="#">SendEmail</a>	Send an email containing the document as a PDF file using the default EMail program.
<a href="#">Export</a>	Display "Export To" dialog and convert the document to PDF.
<a href="#">ZoomIn</a>	Increase the zoom factor.
<a href="#">ZoomOut</a>	Decrease the zoom factor.
<a href="#">Zoom</a>	Increase or decrease the zoom factor. Overloaded.
<a href="#">GotoPage</a>	Display "Goto Page" dialog and navigate to a page. Overloaded.

### CurrentPage

Read only property containing the first visible page in the document. The first page in a document has the index of 1. This property is usefull if you want to implement custom navigation.

### DefaultZoom

Read/write property that specifies the default zoom factor to be used when opening a new document.

### PageCount

Read only property containing the number of pages in this document. Usefull when implementing custom navigation.

### StatusBar

Read only property containing the control's status bar. Use this property to customize the look of the status bar.

### ToolBar

Read only property containing the control's toolbar bar. Through this property, the toolbar can be hidden and the position where the ToolBar object is docked can be changed. In addition, existing buttons can be removed and new ones can be added.

### XFViewerControl

Create a new instance of XFViewerControl.

### Relayout

Forces layout after repositioning the toolbar or status bar.

### LoadXml

Load and display the specified XSL-FO text.

```
void LoadXml(String xml)
void LoadXml(String xml, String baseUrl)
void LoadXml(String xml, String baseUrl, String title)
```

#### Parameters

xml

XSL-FO text to load and display.

baseUrl

Url used to resolve images included in the document and specified by a relative path. Can be null.

title

Document's display title, used when sending the document as email attachment, printing, etc. Can be null.

### LoadUrl

Load and display an XSL-FO document from the specified location.

```
void LoadUrl(String url)
void LoadUrl(String url, String title)
```

#### Parameters

url

String containing a URL that specifies the location of the XSL-FO file.

title

Document's display title, used when sending the document as email attachment, printing, etc.

### Print

Display "Print" dialog and print the document.

### **SendEmail**

Send an email containing the document as a PDF file using the default EMail program.

### **Export**

Display "Export To" dialog and convert the document to PDF.

### **ZoomIn**

Increase the zoom factor.

### **ZoomOut**

Decrease the zoom factor.

### **Zoom**

Increase or decrease the zoom factor. Overloaded.

```
void Zoom(float factor)
void Zoom(Ecrion.Windows.Forms.ZoomFactor factor)
```

### **Parameters**

factor

Zoom factor greater than 0 or a predefined zoom factor.

### **GotoPage**

Display "Goto Page" dialog and navigate to a page. Overloaded.

```
void GotoPage()
void GotoPage(int page)
```

### **Parameters**

page

The page index to navigate to. The first page in a document has the index of 1.