XSL - FO
Visual Design
XF Designer
• WYSIWYG Editing XSL-FO Documents
• Build XSL-FO based templates using Drag & Drop
Server Runtime
XF Rendering Server
• Performant hight-throughput document processing solution
• Wide range of output formats: PDF, PS, AFP, XPS, TIFF
Single source publishing to multiple formats

XSL-FO Tutorial

ecrion.software

Table of Contents

1 XSL-FO Tutorial

This document is designed to help XML programmers to develop XSL-FO documents to be rendered using **XF Rendering Server**.

About XF Rendering Server 2008

XF Rendering Server 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 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 bidimensional vector and mixed vector/raster graphics in XML.

Product Features

- Supports XSL, XSL-FO, SVG and SVG Zipped, XPS, XChart, XML, HTML, XHTML formats, Microsoft Office 2003 and Word 2008 documents as input.
- Produces PDF, AFP, XPS, SVG, PostScript, HTML, Image Files (PNG, GIF, JPEG, BMP, TIFF), IOCA, PCL, INX and Text Files.
- The first XSL-FO formatter in the world able to generate XML Paper Specification(XPS) documents.
- Supports partial document rendering.
- Supports Type1, Type 1 with Postscript Glyphs (OTF), TrueType, AFM font embedding and private fonts.
- 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 and ASPX, .NET and Java Application Programming Interfaces(API's).
- Multiprocessor/Multithreaded (multiple documents can be processed in the same time; threads can be processed on different CPUs if available).
- Is available in 32 bit and 64 bit editions.
- Supports 128 bit strength encryption of generated PDF files.
- Has advanced pagination and layout capabilities.
- Advanced SVG support, with additional support for charts and barcodes
- Can achieve significant speed improvements by enabling the use of more system memory.
- Generates print-ready PDF documents, compliant with PDF/X standards and with full color management support (ICC profiles).
- Includes XF Designer XSL-FO authoring tool.

IMPORTANT: For the latest features and updates please visit our website.

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 website.

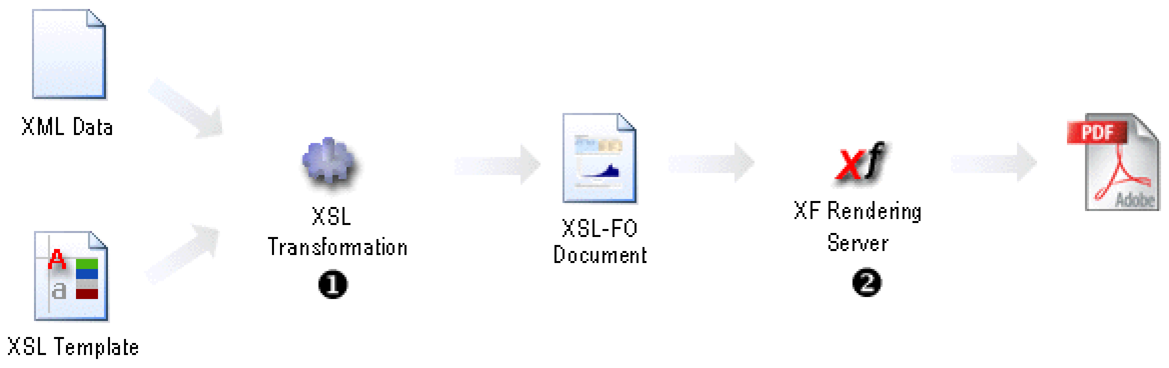
1.1 What is XSL-FO

XSL-FO is an **XML** language designed for describing all visual aspects of paginated documents. The well known HTML is another language for specifying formatting semantics, but is more usable for documents that are presented on screen, and less for materials created for printing, because it does not support pagination elements like headers and footers, page size specifications, footnotes, etc.

XSL-FO is part of XSL language family:
- **XSLT** - (XSL Transformations) a language for transforming XML.
- **XSL-FO** - (XSL Formatting Objects) a language that can be used in XSLT for the purpose of "presenting" the XML.

XSL Formatting Objects is a W3C standard used by **Ecrion XF Rendering Server 2008** to produce print-ready documents in **PDF, AFP, Postscript, TIFF** and other formats.
XF Designer can edit XSL-FO documents in the same way like a HTML editor can edit HTML pages.

The following image depicts the steps required to produce a PDF document (or any other supported output format) using XSL:



As you can see the XML data is transformed together with the XSL stylesheet to produce an XSL-FO document and the document is then converted to PDF.

How hard it is to learn?

The **XSL-FO language** uses **CSS** (Cascading Style Sheets) to describe formatting attributes like fonts, colors and borders, so from this point of view, it should be easy to learn by HTML developers. This manual will help you understand the language and accomplish more complicated tasks.

Here is the traditional **Hello World**, XSL-FO style:

```
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format"> (1)
      <fo:layout-master-set> (2)
            <fo:simple-page-master master-name="LetterPage" page-width="6.5in"
                                                           page-height="1in"> (3)
                  <fo:region-body region-name="PageBody" margin="0.2in"/>
            </fo:simple-page-master>
      </fo:layout-master-set>
```

```
       <fo:page-sequence master-reference="LetterPage">
              <fo:flow flow-name="PageBody" font-family="Times New Roman"
                                                      font-size="24pt">
                     <fo:block> Hello World </fo:block> (4)
              </fo:flow>
       </fo:page-sequence>
</fo:root>
```

There are several things to notice:

(1) Any XML document must have only one root, and XSL- FO makes no exception. The root element for an XSL-FO document is **fo:root**. The word "**fo**" before column character ":" is called a **namespace prefix**. An XML namespace is collection of names identified by an unique URL. It's main role is to avoid collisions when a single XML document contains elements and attributes defined by multiple software modules. The "**fo**" namespace prefix is linked with an unique URL, in this case "http://www.w3.org/1999/XSL/Format" using xmlns attribute. This syntax is based on W3C XML Namespace Spec**.**

(2)(3) The pages structure is defined using **fo:layout-master-set**; more about this in the chapter Pagination 40. For now, is enough to say that it declares one type of page, 11.5 x 8 inches (US Letter).

(4) The "**Hello World**" paragraph is added into the page.

The result of rendering should be identical with the following figure:

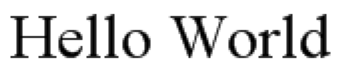

Hello World

Figure 1.

To convert this document into PDF, you can use XF Designer. Open the document and generate the PDF from the tools menu. Or you can use **render.exe**, a console program located in "C:\Program Files\Ecrion Software\XF Rendering Server 2008\bin".

The command line is:

```
       render.exe -fo HelloWorld.fo -pdf C:\Temp\HelloWorld.pdf
```

The command line flag **-pdf** is optional; if not present render will generate a PDF file with a name identical with the input file's name and **.pdf** extension.

1.2 Paragraphs

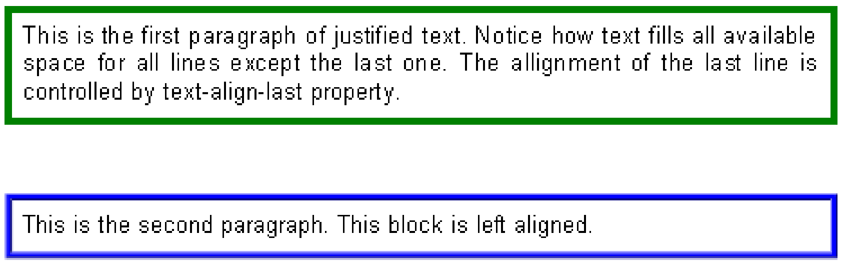
In **XSL-FO**, paragraphs are created using **fo:block** elements.

Various attributes can be set on a paragraph:

- Horizontal alignment is controlled by **text-align** attribute.
- Borders are set using borders attribute.
- Font is set using **font-family**, **font-size**, **font-weight**, etc.
- Spacing between two adjacent paragraphs is set using **space-before** and **space-after**.

```
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
      <fo:layout-master-set>
            <fo:simple-page-master master-name="LetterPage" page-width="5in"
                                                 page-height="2in">
                  <fo:region-body region-name="PageBody" margin="0.2in"/>
            </fo:simple-page-master>
      </fo:layout-master-set>
      <fo:page-sequence master-reference="LetterPage">
           <fo:flow flow-name="PageBody" font-family="Arial" font-size="10pt">(1)
                <fo:block text-align="justify" space-after="0.1cm" (2)
                                    border="3pt solid green" padding="5pt">
                          This is the first paragraph of justified text.
                          Notice how text fills all available space for all lines
                          except the last one.The alignment of the last line is
                          controlled by text-align-last property.
                </fo:block>
                <fo:block space-before="1cm" border="3pt ridge blue" padding="5pt">
                    This is the second paragraph. This block is left aligned.
                </fo:block>
            </fo:flow>
      </fo:page-sequence>
</fo:root>
```

In this example, we have two paragraphs of justified text, surrounded by borders, with 2 centimeters distance between them. The result of rendering should be identical with the following figure:



This is the first paragraph of justified text. Notice how text fills all available space for all lines except the last one. The allignment of the last line is controlled by text-align-last property.

This is the second paragraph. This block is left aligned.

Figure 2.

There are several things to notice:

(1) We set the font to Arial, 12 points height on the parent **fo:flow** element.

(2) The distance between paragraphs is not additive; instead, the largest value will prevail. If a **fo:block** is the first or the last element in a page, you will also notice that the space is not present anymore! This behavior is controlled by **space-before.conditionality** and **space-after.conditionality** attributes. If set to "**retain**", the corresponding space will not be discarded.

Text Alignment

Horizontal alignment of text is controlled by two attributes: **text-align** which will set the alignment for all lines of text, except the last one, which is set to **text-align-last**.

This is important to remember, because if your paragraph has only one line of text, you have to use text-align-last to set the alignment.

Possible values for text-align and text-align-last are:
- **left** (the default) to perform alignment to the right.
- **right**, to perform alignment to the right.
- **center**, to center the paragraph.
- **justify**, to justify the text, so that if fills the whole line.

Vertical alignment of text is controlled by **display-align** attribute. This attribute can have the following values:
- **auto** (the default)
- **before**
- **center**
- **after**

Fonts

There are six properties that control the aspect of text:**font-family**, **font-style**, **font-variant**, **font-weight**, **font-size**, **line-height**.

To set the font face, you use font-family attribute. For example font-family="Arial" will specify that Arial font must be used.
If multiple font families are specified, the renderer will pick the first available, so you should list the fonts from the most specific to the most generic. For example, font-family="Arial, Helvetica" will select "Helvetica" if "Arial" is not present in the system.

The weight of the font can be specified using font-weight attribute. You can set it to either an absolute value ("bold" or "normal"), or to a value relative to parent element's font weight ("bolder" or "lighter").

To specify the font size, use font-size attribute. This size can be a length (1cm, 0.5in, 10pt, etc) or a percentage (0.5, 50%) from parent element's font.

A very important, and often misused property is **line-height**. This property determines the minimum line-height for a block element. The default value for line-height is 120%, that is, the line will be 20% taller than the text.

For example, if the text is 10 points height, the line height will be of 12 points. The text will be centered on the line, 1 point from top, and 1 points from bottom. In the next example we set the line-height to 200%:

```
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
      <fo:layout-master-set>
            <fo:simple-page-master master-name="LetterPage" page-width="5in"
                                                  page-height="1.5in">
                  <fo:region-body region-name="PageBody" margin="0.2in"/>
            </fo:simple-page-master>
      </fo:layout-master-set>
      <fo:page-sequence master-reference="LetterPage">
           <fo:flow flow-name="PageBody">
               <fo:block font-family="Arial" font-size="10pt" line-height="200%"
                                      border="3pt outset blue" padding="5pt">
                  For this paragraph, the line-height is set to 200%.
                  Because the font is 12 points height, an extra 12 points
            </fo:flow>
      </fo:page-sequence>
</fo:root>
```

The result of rendering is displayed in following figure:

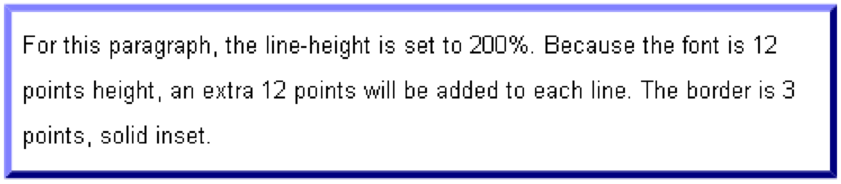

For this paragraph, the line-height is set to 200%. Because the font is 12 points height, an extra 12 points will be added to each line. The border is 3 points, solid inset.

Figure 3.

We have mentioned before that this is the minimum **line height**: if a line contains an image of let's say 100 points in height, the total line height (only for that line) will be of 102 points.

To set all font properties at once you can use **font** shortcut attribute. A shortcut attribute will set the values for multiple attributes at once. The shortcut attribute font has the following syntax:

```
      font="{style} {variant} {weight} {size}/{line-height} family".
```

For example, to achieve the same effect as the first example in this chapter, you could use font="10pt Arial".

Note:
- By using this shortcut attribute, instead of specifying each font property individually, you will reset the values of non specified attributes:

```
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
      <fo:layout-master-set>
            <fo:simple-page-master master-name="LetterPage" page-width="5in"
                                                        page-height="1.5in">
                  <fo:region-body region-name="PageBody" margin="0.2in"/>
            </fo:simple-page-master>
      </fo:layout-master-set>
      <fo:page-sequence master-reference="LetterPage">
            <fo:flow flow-name="PageBody">
```

```
                    <fo:block font="10pt bold Arial" border="1px solid blue">
                          When the child block ends, the remaining of parent's
                          content is displayed.
                    </fo:block>
             </fo:flow>
      </fo:page-sequence>
</fo:root>
```

The result of rendering is displayed as following:

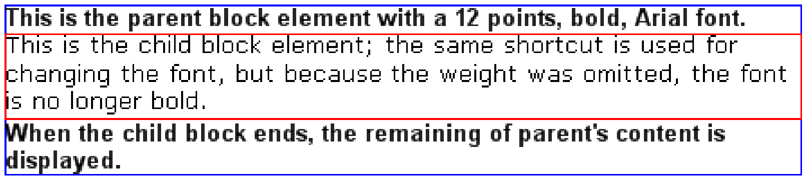

This is the parent block element with a 12 points, bold, Arial font.
This is the child block element; the same shortcut is used for changing the font, but because the weight was omitted, the font is no longer bold.
When the child block ends, the remaining of parent's content is displayed.

Figure 4.

XF Rendering Server 2008 can use both Type1 Postscript fonts and TrueType fonts.

Fonts must be installed in Windows before use.
To install a new font, simply drag and drop the font files in Control Panel/Fonts.

Borders

The **border** properties specify the width, color, and style of the borders a object. These properties apply to all XSL-FO elements.

- Border widths are set using **border-top-width**, **border-right-width**, **border-bottom-width**, **border-left-width** attributes. The value can be specified as a length (1pt, 0.5cm, etc). To set all four widths at once you can use border-width shorthand. For example, to set all borders to 0.1 inches thick, you use border-width="0.1in".
- The border color properties are **border-top-color**, **border-right-color**, **border-bottom-color**, **border-left-color**. The shorthand for setting the color for all four borders at once is border-color.
- The border style properties specify the line style of a box's border (solid, double, dashed, etc.). The properties defined in this section refer to the border-style value type, which may take one of the following:

none or **hidden**	No border
dotted	The border is a series of dots.
dashed	The border is a series of short line segments.
solid	The border is a single line segment.
double	The border is two solid lines. The sum of the two lines and the space between them equals the value of 'border width'.
groove	The border looks as though it were carved into the canvas.
ridge	The opposite of 'grove': the border looks as though it were coming out of the canvas.
inset	The border makes the entire box look as though it were embedded in the canvas.
outset	The opposite of 'inset': the border makes the entire box look as though it were coming out of the canvas.

All borders are drawn on top of the box's background.
To set all attributes for a given border, you can use border-top, border-right, border-bottom, border-left shorthand attributes. The format is:

```
      border-top="{width} {style} {color}".
```

The following notations are equivalent:

```
<fo:block border-top-style="solid" border-top-color="red" border-top-width="1mm">
...
</fo:block>
<fo:block border-top="1mm solid red">
...
</fo:block>
```

The border property is a shorthand property for setting the same width, color, and style for all four borders of a box. The border property cannot set different values on the four borders. To do so, one or more of the other border properties must be used. The format is:

```
      border="{width} {style} {color}".
```

Rounded Borders

XF Rendering Server 2008 supports drawing boxes with rounded corners using the border-radius properties described in the CSS3 Backgrounds and Borders Module.

You can set the radius for each corner individually by changing **border-top-right-radius**, **border-bottom-right-radius**, **border-bottom-left-radius**, **border-top-left-radius** properties. To change all four corners at once, use **border-radius**. The format is:

```
      border-radius="{width} {width}".
```

If only one value is specified, the corner will be drawn as a circle; if two values are specified, the corner will be drawn as an ellipse.
You can read more about CSS3 Backgrounds and Borders Module in the CSS3 W3C Working Draft.

The following sample will draw a paragraph surrounded by a rounded border:

```
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
      <fo:layout-master-set>
            <fo:simple-page-master master-name="all-pages" page-width="6.5in"
                                                           page-height="1in">
                  <fo:region-body region-name="xsl-region-body" margin="0.2in"/>
            </fo:simple-page-master>
      </fo:layout-master-set>
      <fo:page-sequence master-reference="all-pages">
            <fo:flow flow-name="xsl-region-body" font-family="Times New Roman"
                                    font-size="36pt" text-align="center">
                  <fo:block background="pink" border="3pt solid red"
                        border-radius="10pt">Hello World</fo:block>
                  <fo:table space-before="10pt">
                  </fo:table>
            </fo:flow>
      </fo:page-sequence>
</fo:root>
```

Note:
- The background will have rounded corners even if the border is not visible.

The result of rendering is displayed in following figure:

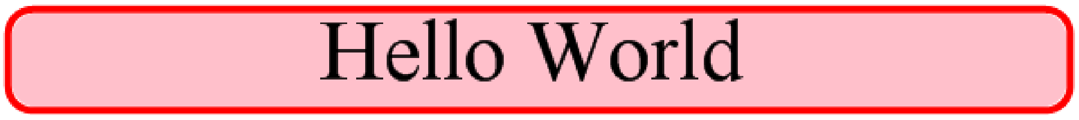


Figure 5

Drop Shadows

Another new feature described in CSS3 is the ability to specify drop shadows for various page elements. You can read more in the CSS3 W3C Working Draft.

In the next example, the shadow is drawn at a 3pt offset in both horizontal and vertical directions, in a light green color.
The shadow is transparent, so that the overlapped text can be seen through it.

```
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
      <fo:layout-master-set>
            <fo:simple-page-master master-name="LetterPage" page-width="6.5in"
                                                           page-height="1in">
                  <fo:region-body region-name="PageBody" margin="0.2in"/>
```

```
             </fo:simple-page-master>
      </fo:layout-master-set>
      <fo:page-sequence master-reference="LetterPage">
          <fo:flow flow-name="PageBody" font-family="Times New Roman"
                                                   font-size="16pt">
              <fo:block>
             The following
                    <fo:inline background="yellow" box-shadow="3pt 3pt lime"
                           padding="3pt" border-radius="4pt"> word </fo:inline>
             has an yellow background and a light green shadow.
             </fo:block>
          </fo:flow>
      </fo:page-sequence>
</fo:root>
```

The result of rendering is displayed in following figure:

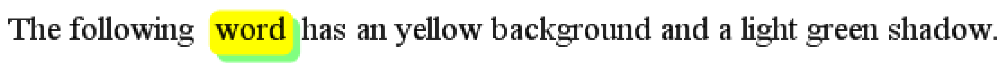

The following word has an yellow background and a light green shadow.

Figure 6.

Backgrounds

XSL-FO object's backgrounds may be colors or images. Background properties allow authors to position a background image, repeat it, etc.

- To set the background color of an element use **background-color** attribute. It can be set to either a color value or the keyword "transparent".
- **background-image** sets the background image of an element. When setting a background image, authors should also specify a background color that will be used when the image is unavailable. When the image is available, it is rendered on top of the background color. (Thus, the color is visible in the transparent parts of the image). Values for this property are either an URL, to specify the image, or 'none', when no image is used.
- If a background image is specified, **background-repeat** attribute specifies whether the image is repeated (tiled), and how. It may take one of the following values:

repeat	The image is repeated both horizontally and vertically.
repeat-x	The image is repeated horizontally only.
repeat-y	The image is repeated vertically only.
no-repeat	The image is not repeated: only one copy of the image is drawn.

- If a background image has been specified, **background-position** property specifies it's initial position. Values have the following meanings:

percentage percentage	With a value pair of '0% 0%', the upper left corner of the image is aligned with the upper left corner of the box's padding edge. A value pair of '100% 100%' places the lower right corner of the image in the lower right corner of padding area. With a value pair of '14% 84%', the point 14% across and 84% down the image is to be placed at the point 14% across and 84% down the padding area.
length length	With a value pair of '2cm 2cm', the upper left corner of the image is placed 2cm to the right and 2cm below the upper left corner of the padding area.
top left and **left top**	Same as '0% 0%'.
top,top center and **center top**	Same as '50% 0%'.
right top and **top right**	Same as '100% 0%'.
left,left center and **center left**	Same as '0% 50%'.
center and **center center**	Same as '50% 50%'.
right right center and **center right**	Same as '100% 50%'.
bottom left and left bottom	Same as '0% 100%'.
bottom bottom center and **center bottom**	Same as '50% 100%'.
bottom right and **right bottom**	Same as '100% 100%'.

If only one percentage or length value is given, it sets the horizontal position only, the vertical position will be 50%. If two values are given, the horizontal position comes first. Combinations of length and percentage values are allowed, (e.g., '50% 2cm').
Negative positions are allowed. Keywords cannot be combined with percentage values or length values (all possible combinations are given above).

To set all attributes for one element's background, you can use background shortcut attribute. The format is:

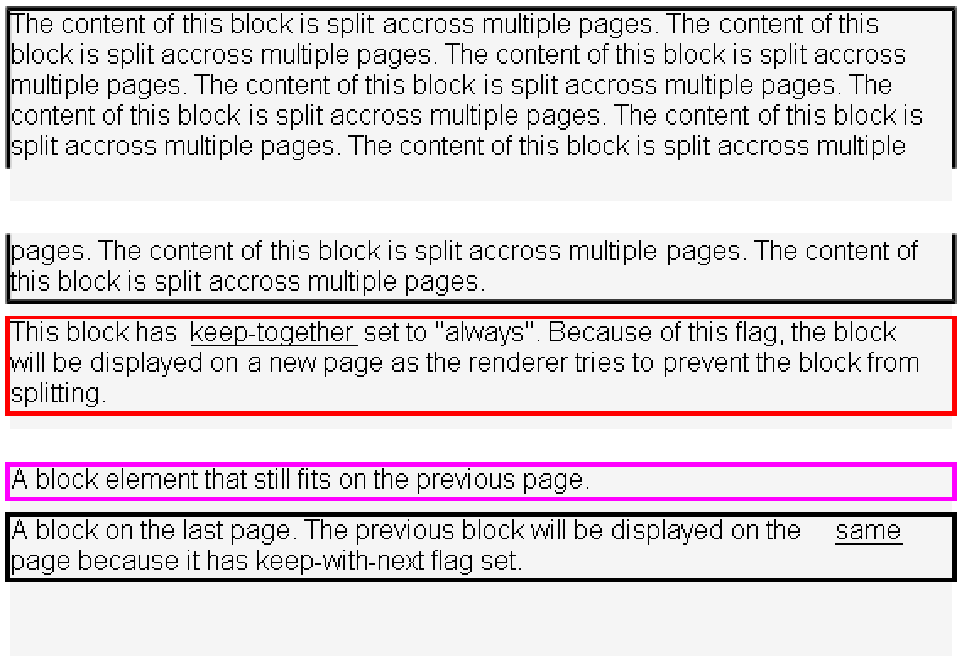
```
    background="{color} {image} {repeat} {position}".
```

1.3 Flow Layout

XSL-FO documents have **flow layout**, that is, content "flows" from one page to the next one:

```
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
   <fo:layout-master-set>
      <fo:simple-page-master master-name="LetterPage" page-width="6in"
                                                     page-height="1.4in">
            <fo:region-body region-name="PageBody" margin="0.1in"
                                  background-color="rgb(245,245,245)"/>
      </fo:simple-page-master>
   </fo:layout-master-set>
   <fo:page-sequence master-reference="LetterPage">
      <fo:flow flow-name="PageBody" font="12pt Arial">
            <fo:block border="2pt solid black" space-after="5pt">
                  The content of this block is split across multiple
                  pages.The content of this block is split .....
                  ..............................................
                  The content of this block is split across multiple pages.
            </fo:block>
            <fo:block border="2pt solid red" keep-together="always"> (1)
                  This block has <fo:inline text-decoration="underline">
                              keep-together</fo:inline> set to "always".
                  Because of this flag, the block will be displayed on
                  a new page as the renderer tries to prevent the block
                  from splitting.
            </fo:block>
            <fo:block border="2pt solid rgb(255,0,255)" space-after="5pt"
                                    keep-with-next="always"> (2)
                  A block element that still fits on the previous page.
            </fo:block>
            <fo:block border="2pt solid black">
                  A block on the last page. The previous block will be
                  displayed on the <fo:inline text-decoration="underline">
                                    same</fo:inline> page because it has
                  keep-with-next flag set.
            </fo:block>
      </fo:flow>
   </fo:page-sequence>
</fo:root>
```

The result of rendering is displayed in following figure:



The content of this block is split accross multiple pages. The content of this block is split accross multiple pages. The content of this block is split accross multiple pages. The content of this block is split accross multiple pages. The content of this block is split accross multiple pages. The content of this block is split accross multiple pages. The content of this block is split accross multiple

pages. The content of this block is split accross multiple pages. The content of this block is split accross multiple pages.

This block has keep-together set to "always". Because of this flag, the block will be displayed on a new page as the renderer tries to prevent the block from splitting.

A block element that still fits on the previous page.

A block on the last page. The previous block will be displayed on the same page because it has keep-with-next flag set.

Figure 7.

There are several properties that control how and when a block of text is split across multiple pages. There are:

- **break-before** and **break-after** attributes will force a page break before or after a block element. By block element we mean elements that fill all available horizontal space like paragraphs 6, tables 32 and lists 37. For example, you might want to use this to start chapters on a new page.
- **keep-together** attribute prevents splitting of a block element. If there is not enough room to display the block on the current page, the block will be displayed on the next one. **(1)**
- **keep-with-next** and **keep-with-previous** will link a block element with the previous/next sibling block. This is useful to prevent page breaks occur between two closely related elements, like chapter title and chapter contents. **(2)**
- **widows** and **orphans** attributes are useful to control contextual information. The default values for this properties is "2", preventing the display of last line of a paragraph by itself at the top of a page (a widow) or the first line of a paragraph by itself at the bottom of a page (an orphan). You can see in the example above the **fo:block** will display the two line on the second page.

1.4 Inline Text Formatting

Inline elements allow XSL-FO developers to specify attributes for individual pieces of inline content (text and images), instead of the whole block.

In the example bellow, a fragment of text is filled with red, and it's font weight is set to bold:

```
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
   <fo:layout-master-set>
      <fo:simple-page-master master-name="LetterPage" page-width="5in"
                                                     page-height="1in">
            <fo:region-body region-name="PageBody" margin="0.2in"/>
      </fo:simple-page-master>
   </fo:layout-master-set>
   <fo:page-sequence master-reference="LetterPage">
      <fo:flow flow-name="PageBody">
            <fo:block font="14pt Arial">
                  Some <fo:inline font-weight="bold" color="red">
                              inline text</fo:inline> formatting.(1)
            </fo:block>
      </fo:flow>
   </fo:page-sequence>
</fo:root>
```

The result of rendering is displayed in following figure:

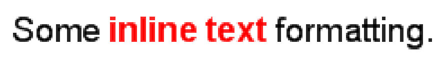


Figure 8.

Things to notice:
(1) The **fo:inline** element wraps the fragment "**inline text**" and sets **font-weight** to bold. The text color is set to red using **color** attribute.

Any color can be described using either a standard color value (see Colors 97) or by using it's red, green and blue components. The following notations are equivalent:

```
      <fo:inline color="red">Hello</fo:inline>
      <fo:inline color="rgb(255,0,0)">Hello</fo:inline>
```

Subscripts and Superscripts

Inline elements also allow creation of sub-scripts of super-scripts:

```
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
   <fo:layout-master-set>
      <fo:simple-page-master master-name="LetterPage" page-width="5in"
                                         page-height="1.6in">
            <fo:region-body region-name="PageBody" margin="0.2in"/>
      </fo:simple-page-master>
   </fo:layout-master-set>
   <fo:page-sequence master-reference="LetterPage">
       <fo:flow flow-name="PageBody" font="10pt Arial">
              <fo:block>
                     Normal text
                     <fo:inline baseline-shift="sub"
                     background="rgb(66,165,255)">sub-script</fo:inline>
                     normal text
                     <fo:inline baseline-shift="super"
                     background="rgb(66,165,255)">super-script</fo:inline>
                     normal text.
              </fo:block>
              <fo:block>
                     Normal text
                     <fo:inline baseline-shift="-50%"
                            background="rgb(66,165,255)">-50%</fo:inline>
                     normal text
                     <fo:inline baseline-shift="50%"
                            background="rgb(66,165,255)">+50%</fo:inline>
                     normal text.
              </fo:block>
              <fo:block>
                     Normal text
                     <fo:inline baseline-shift="-5pt"
                            background="rgb(66,165,255)">-5pt</fo:inline>
                     normal text
                     <fo:inline baseline-shift="5pt"
                            background="rgb(66,165,255)">5pt</fo:inline>
                     normal text.
              </fo:block>
       </fo:flow>
   </fo:page-sequence>
</fo:root>
```

The rendering result is displayed in the next figure:

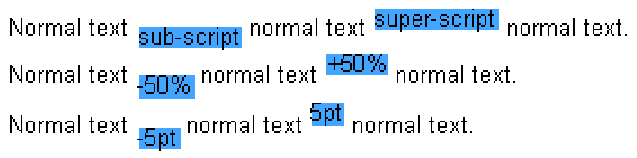

Normal text sub-script normal text super-script normal text.
Normal text -50% normal text +50% normal text.
Normal text -5pt normal text 5pt normal text.

Figure 9.

The property that controls the alignment of an inline element vertically within it's parent line is baseline-shift. As you can see in this example, the text can be shifted vertically using either "**sub**" or "**super**" which will use font metrics to determine the subscript or superscript positions.
You can also use a **percentual** or **absolute** value.

1.5 Graphics

XSL-FO provides the means to display images and vectorial graphics through two elements: **fo:instream-foreign-object** when you have the content embedded in the XSL-FO document and **fo:external-graphic** when the image resides in an external file.

SVG

One of the supported formats for **fo:instream-foreign-object** is **SVG** (Scalable Vector Graphics):

```
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
   <fo:layout-master-set>
      <fo:simple-page-master master-name="LetterPage" page-width="4in"
                                                     page-height="3in">
          <fo:region-body region-name="PageBody" margin="0.1in"/>
      </fo:simple-page-master>
   </fo:layout-master-set>
   <fo:page-sequence master-reference="LetterPage">
      <fo:flow flow-name="PageBody">
          <fo:block font-family="Arial" font-size="10pt"
                                 font-weight="bold">
             SVG Graphics Example
          </fo:block>
          <fo:block>
              <fo:instream-foreign-object content-height="2.2in"> (1)
                  <svg xmlns="http://www.w3.org/2000/svg"
                                        width="480" height="280">
                     <linearGradient id="Grad1"gradientUnits="objectBoundingBox"
                                        x1="0" y1="0" x2="1" y2="1"> (2)
                                 <stop stop-color="rgb(238,130,238)" offset="0"/>
                                 <stop stop-color="blue" offset="0.2"/>
                                 <stop stop-color="lime" offset="0.4"/>
                                 <stop stop-color="yellow" offset="0.6"/>
                                 <stop stop-color="rgb(255,165,0)" offset="0.8"/>
                                 <stop stop-color="red" offset="1"/>
                     </linearGradient>
<!-- Linear gradient on the stroke of a rectangle -->
                     <rect x="20"y="20" width="440" height="80" fill="url(#Grad1)"/>
                           <text font-family="Arial" font-size="14" x="20" y="130">
                                        Multi-color linear gradient.
                           </text>
<!-- Radial gradient on the stroke of a rectangle -->
                     <radialGradient id="Grad2"gradientUnits="userSpaceOnUse"
                           cx="240" cy="210" r="220" fx="240" y="210"> (3)
                                 <stop stop-color="black" offset="0"/>
                                 <stop stop-color="yellow" offset="0.2"/>
                                 <stop stop-color="red" offset="0.4"/>
                                 <stop stop-color="blue" offset="0.6"/>
                                 <stop stop-color="white" offset="0.8"/>
                                 <stop stop-color="green" offset="1"/>
                     </radialGradient>
                     <rect x="20" y="150" width="440" height="80"
                                 fill="url(#Grad2)" stroke-width="40"/>
                           <text font-family="Arial" font-size="14"
                                               x="20" y="260">
                                  Multi-color radial gradient.
                           </text>
                    </svg>
```

```
                </fo:instream-foreign-object>
             </fo:block>
        </fo:flow>
   </fo:page-sequence>
</fo:root>
```

The rendering result is displayed in the next figure:

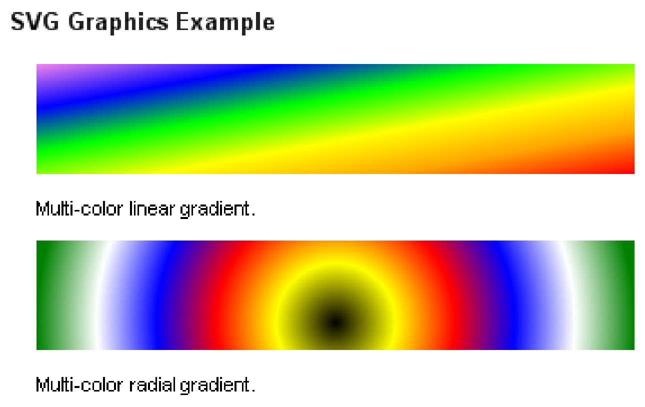


Figure 10.

The important points in example above are:
(1) fo:instream-foreign-object is used to wrap the SVG graphic.
(2)(3) Inside **SVG**, we fill two rectangles with gradients.

XChart

In addition to SVG, **XF Rendering Server** supports **XChart**, an XML language developed by Ecrion Software for the purpose of describing generic charts.
Any **XChart** document has **xc:root** element. This is the top node in an xChart document.
Typically, this element contains the declaration of '**xChart**' namespace **xmlns:xc="http://www.ecrion.com/xc"**. A document can range from an empty fragment (no content inside **xc:root**) to a complex, deeply nested collection of xChart, XSL-FO and SVG elements.

There can be produced **Pie**, **Area**, **Bar**, **Spider**, **Scatter, Spider** and **Line Charts**. It is also able to render combinations of these types.
In the example below is displayed a **3D Pie XChart**:

```
<xc:root xmlns:xc="http://www.ecrion.com/xc"
      xmlns:fo="http://www.w3.org/1999/XSL/Format"
      xmlns:svg="http://www.w3.org/2000/svg" width="450pt" height="180pt">
   <xc:pie-3d x="40pt" y="40pt" width="200pt" height="100pt" base-height="0.3"
                                                     base-shading="0.15">(1)
      <xc:slice-3d fill-color="red" stroke-color="black" stroke-width="0.3pt"
                                                     percent="13.48">(2)
             <xc:title offset="25pt" stroke-color="transparent">13.48%</xc:title>
      </xc:slice-3d>
      <xc:slice-3d fill-color="navy" stroke-color="black" stroke-width="0.3pt"
                                                            percent="8.98">
```

```
            <xc:title offset="25pt" stroke-color="transparent">8.98%</xc:title>
      </xc:slice-3d>
      <xc:slice-3d fill-color="orange" stroke-color="black" stroke-width="0.3pt"
                                                   percent="7.52">(3)
            <xc:title offset="25pt" stroke-color="transparent">7.52%</xc:title>
      </xc:slice-3d>
      <xc:slice-3d fill-color="silver" stroke-color="black" stroke-width="0.3pt"
                                                   percent="1.18">
            <xc:title offset="25pt" stroke-color="transparent">1.18%</xc:title>
      </xc:slice-3d>
      <xc:slice-3d fill-color="violet" stroke-color="black" stroke-width="0.3pt"
                                                   percent="18.59">
            <xc:title offset="25pt" stroke-color="transparent">18.59%</xc:title>
      </xc:slice-3d>
      <xc:slice-3d fill-color="gray" stroke-color="black" stroke-width="0.3pt"
                                                   percent="16.94">
            <xc:title offset="25pt" stroke-color="transparent">16.94%</xc:title>
      </xc:slice-3d>
      <xc:slice-3d fill-color="green" stroke-color="black" stroke-width="0.3pt"
                                                   percent="23.90">
            <xc:title offset="25pt" stroke-color="transparent">23.90%</xc:title>
      </xc:slice-3d>
      <xc:slice-3d fill-color="yellow" stroke-color="black" stroke-width="0.3pt"
                                                   percent="9.68">
            <xc:title offset="25pt" stroke-color="transparent">9.68%</xc:title>
      </xc:slice-3d>
   </xc:pie-3d>
</xc:root>
```

The rendering result is displayed in the next figure:

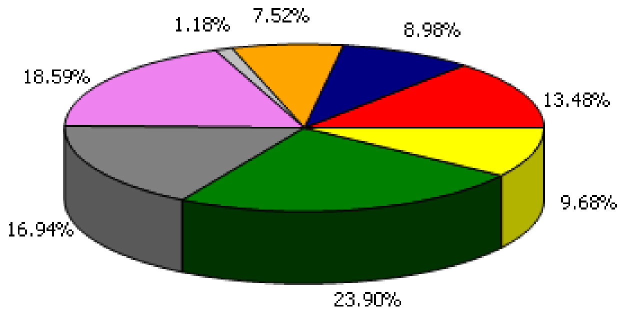


Figure 11.

Things to notice:
(1)(2) xc:pie-3d describes a 3D pie chart and **xc:slice-3d** a slice in this chart.
(3) stroke-color and **stroke-width** are attributes used to paint the outline of an element.

There can be produced different types of bar chart, from simple ones to group-stacked or group-clustered. Here is an example of a group-cluster chart:

```
<xc:root xmlns:xc="http://www.ecrion.com/xc" xmlns:svg="http://www.w3.org/2000/svg"
      xmlns:fo="http://www.w3.org/1999/XSL/Format" width="500pt" height="320pt">
    <xc:graph x="5pt" y="5pt" width="490pt" height="290pt">(1)
```

```
      <xc:plot-area>(2)
            <xc:serie-group stacking="clustered" cluster-spacing="4pt">(3)
                  <xc:serie stroke-color="transparent">(4)
                        <xc:bar width="10pt" fill-color="red"
                              category="1989-01-31" value="10000"/>(5)
                        <xc:bar width="10pt" fill-color="red"
                              category="1989-02-1" value="-18000"/>
                        <xc:bar width="10pt" fill-color="red"
                              category="1989-03-31" value="30000"/>
                        <xc:bar width="10pt" fill-color="lime"
                              category="1989-05-31" value="11931.89251254"/>
                  </xc:serie>
                  <xc:serie stroke-color="transparent">
                        <xc:bar width="10pt" fill-color="blue"
                              category="1989-01-31" value="1865.76193894"/>
                        <xc:bar width="10pt" fill-color="blue"
                              category="1989-02-1" value="-10713.15292294"/>
                        <xc:bar width="10pt" fill-color="blue"
                              category="1989-03-31" value="10931.89251254"/>
                        <xc:bar width="10pt" fill-color="blue"
                              category="1989-04-2" value="1931.89251254"/>
                        <xc:bar width="10pt" fill-color="blue"
                              category="1989-05-31" value="1931.89251254"/>
                  </xc:serie>
            </xc:serie-group>
      </xc:plot-area>
      <xc:value-axis orientation="vertical" unit="5000">
            <xc:title font-weight="bold">Value</xc:title>
            <xc:grid-lines stroke-color="silver"/>
            <xc:axis-labels offset="4pt"/>
      </xc:value-axis>
      <xc:category-axis orientation="horizontal">
            <xc:major-tick-marks style="outside"/>
            <xc:title font-weight="bold">Date</xc:title>
            <xc:grid-lines stroke-color="silver" stroke-dash-array="2px 2px"/>
            <xc:axis-labels offset="3pt"/>
      </xc:category-axis>
    </xc:graph>
</xc:root>
```

The rendering result is displayed in the next figure:

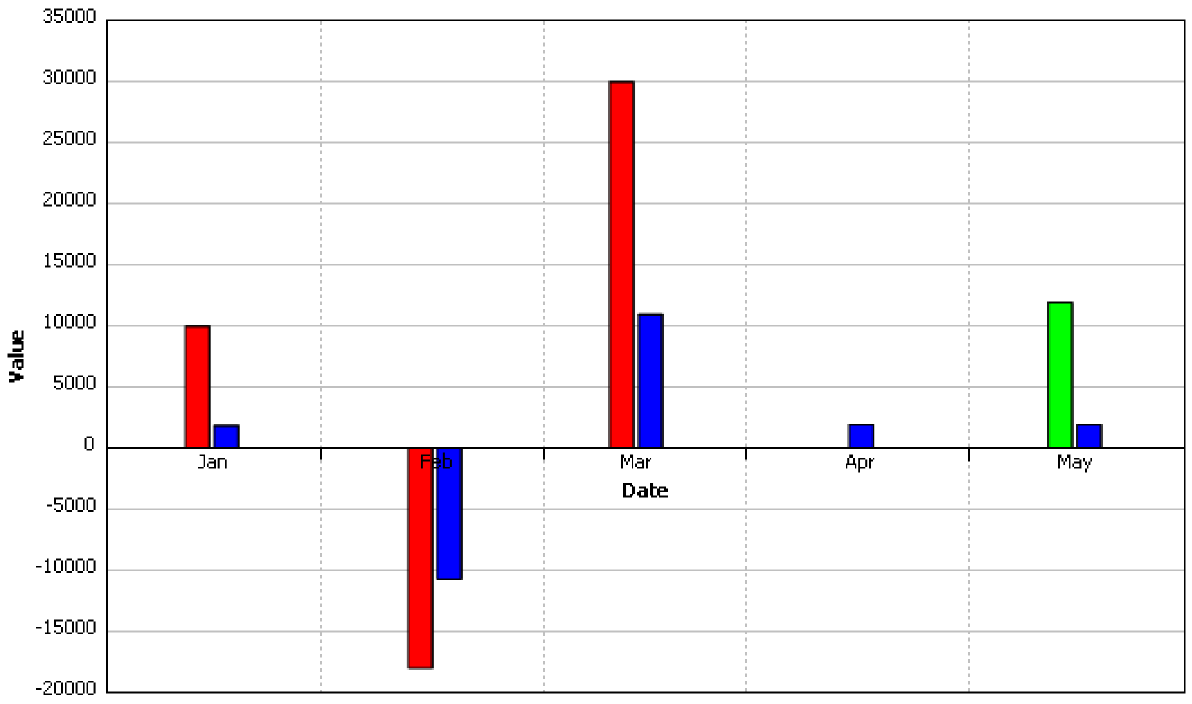


Figure 12.

Things to notice:

(1) xc:graph is the element describing one or more data series in a two dimensional coordinate system.
(2) xc:plot-area sets various properties for the area reserved to draw series of data and **xc:serie** contains one or more data points.
(3) xc:serie-group stacking and **cluster-spacing** attributes defines the type of bar chart: clustered and the space between clusters.
(4)(5) stroke and **fill** attributes are used to paint the outline or the background of an element.

For more examples and information about XChart, please see Chart Samples and xChart 1.0 Language Reference.

External Graphics

To display an image from an external file, use **fo:external-graphic**.

All majors formats are supported, including BMP, JPEG, GIF, PNG, WMF, POSTSCRIPT, TIFF, etc. Unisys U.S. LZW Patent No. 4,558,302 used for GIF image compression expired on June 20, 2003, the counterpart patents in the United Kingdom, France, Germany and Italy expired on June 18, 2004, the Japanese counterpart patents expired on June 20, 2004 and the counterpart Canadian patent expired on July 7, 2004.

For more information see Unisys Website.

```
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
      <fo:layout-master-set>
            <fo:simple-page-master master-name="LetterPage" page-width="3in"
                                                           page-height="1.2in">
                  <fo:region-body region-name="PageBody" margin="0.1in"/>
            </fo:simple-page-master>
      </fo:layout-master-set>
      <fo:page-sequence master-reference="LetterPage">
            <fo:flow flow-name="PageBody" font-family="Arial" font-size="10pt">
                  <fo:block font-weight="bold">
                        External Graphics Example
                  </fo:block>
                  <fo:block>
                        Text Before <fo:external-graphic src="ecrion-logo.png"
                              content-height="0.7in" vertical-align="middle"/>
                        Text After
                  </fo:block>
            </fo:flow>
      </fo:page-sequence>
</fo:root>
```

The rendering result is displayed in the next figure:

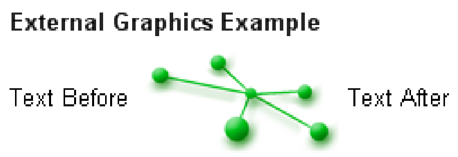


Figure 13.

There are several things to notice in this example:

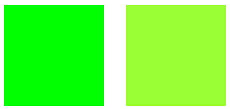
- Image urls can be either absolute or relative. When relative, the location of the XSL-FO document is used to compute the full path to the image. You can use baseUrl property (see XF Rendering Server 2008 Programmers Reference) to override this location.
- Image can be scaled using **content-width** and **content-height** properties. In this example we specify only the desired height and the width is computed automatically by the renderer, preserving the aspect ratio.
- Inline graphics can be shifted vertically using **vertical-align** attribute.

1.6 Color Management

Graphics professionals know the importance of **color management**. No matter how much thought you put into the color scheme for a given project, all of that work is for naught if you can't get your printed results to match your expectations.

Each color is encoded as a combination of Red, Green and Blue values. What do these values mean; e.g. what color is 88/249/17? Without any context, it is a meaningless triplet of numbers. This is where color spaces come in.

A color space provides the definition for what color the numerical combination represents.
If our example of 88/249/17 is interpreted using the Adobe RGB color space, it is a vibrant, attention getting green. If, on the other hand, the same value is interpreted using the sRGB color space, it is a pale, yellowish-green.

(88, 249, 17) in Adobe RGB  The same RGB value in sRGB

RGB Colors

By default all colors used in your XSL-FO documents are considered to be given in the sRGB color space.

To use RGB colors, you can either specify values for the each component (Red, Green and Blue), or you can specify a color name. **XF Rendering Server 2008** supports the extended pallete described in the SVG specifications for both SVG and XSL-FO input.

The following notations are equivalent:

```
      <fo:block color="red">Red text.</fo:block>
      <fo:block color="rgb(255,0,0)">Red text.</fo:block>
      <fo:block color="rgb(100%,100%,100%)">Red text.</fo:block>
      <fo:block color="#FF0000">Red text.</fo:block>
      <fo:block color="#F00">Red text.</fo:block>
```

The last notation is called a short notation. The three-digit RGB notation (#rgb) is converted into six-digit form (#rrggbb) by replicating digits, not by adding zeros. For example, #fb0 expands to #ffbb00.

When generating PDF, all RGB colors are mapped to a predefined PDF color space called DeviceRGB.

CMYK Colors

You can use an ECRION specific extension to specify colors in CMYK (Cyan,Magenta,Yellow,Black) format:

```
      <fo:block color="cmyk(0,100,100,0)">Red text.</fo:block>
```

When generating PDF, all CMYK colors specified using this extension are mapped to DeviceCMYK color space. If you have a CMYK color profile you wish to use, read below.

ICC Profiles

In addition, the XSL-FO recommendation specifies how to use colors described by an external ICC profile. With accurate monitor and printer profiles, your prints will closely match what you see on your monitor.

To declare a color space use **fo:color-profile**:

```
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
    <fo:declarations>
       <fo:color-profile color-profile-name="RGBColorProfile"
                            src="ColorProfiles\RGB\AppleRGB.icc"/>
       <fo:color-profile color-profile-name="CMYKColorProfile"
                     src="ColorProfiles\CMYK\USWebCoatedSWOP.icc"/> (1)
    </fo:declarations>
    <fo:layout-master-set>
       <fo:simple-page-master master-name="all-pages" page-width="8in"
                                                page-height="11in">
          <fo:region-body region-name="xsl-region-body" column-gap="0.25in"
                     padding-top="6pt" padding-left="6pt" padding-right="6pt"
                     padding-bottom="6pt" margin-top="0.7in" margin-left="0.7in"
                     margin-right="0.7in" margin-bottom="0.7in"/>
          <fo:region-before region-name="xsl-region-before" display-align="after"
                     extent="0.7in"padding-top="6pt" padding-left="0.7in"
                     padding-right="0.7in" padding-bottom="6pt"/>
          <fo:region-after region-name="xsl-region-after" display-align="before"
                     extent="0.7in" padding-top="6pt" padding-left="0.7in"
                     padding-right="0.7in" padding-bottom="6pt"/>
       </fo:simple-page-master>
       <fo:page-sequence-master master-name="default-sequence">
             <fo:repeatable-page-master-reference master-reference="all-pages"/>
       </fo:page-sequence-master>
    </fo:layout-master-set>
    <fo:page-sequence master-reference="default-sequence">
       <fo:flow flow-name="xsl-region-body"font-size="12pt"
                                 font-family="Times New Roman">
          <fo:block background="rgb(255,0,0)">RGB Color</fo:block>
          <fo:block background="rgb-icc(255,0,0,#RGBColorProfile,1,0,0)">
                                              ICC RGB Color</fo:block> (2)
          <fo:block background="rgb-icc(255,0,0,#CMYKColorProfile,0,1,1,0)">
                                              ICC CMYK Color</fo:block>
          <fo:block background="rgb(0,255,0)">RGB Color</fo:block>
          <fo:block background="rgb-icc(0,255,0,#RGBColorProfile,0,1,0)">
                                              ICC RGB Color</fo:block>
          <fo:block background="rgb-icc(0,255,0,#CMYKColorProfile,0.63,0,1,0)">
                                              ICC CMYK Color</fo:block>
          <fo:block background="rgb(255,0,255)">RGB Color</fo:block>
          <fo:block background="rgb-icc(255,0,255,#RGBColorProfile,1,0,1)">
                                              ICC RGB Color</fo:block>
          <fo:block background="rgb-icc(255,0,255,#CMYKColorProfile,0.27,0.82,0,0)">
                                              ICC CMYK Color</fo:block>
       </fo:flow>
    </fo:page-sequence>
</fo:root>
```

The important points in this example are:

(1) We declare two color spaces, named "**RGBColorProfile**" and "**CMYKColorProfile**".
(2) We use **rgb-icc** function to specify the intensities of each component in this color space. The name of this function is a little misleading, because **rgb-icc** can be used to describe not only **RGB** (Red, Green, Blue) colors, but also **Grascale**, **CMYK** and spot (**Pantone**) colors.

rgb-icc takes the following arguments:

```
rgb-icc(fallbackRed,fallbackGreen,fallbackBlue,#colorProfileReference,
      component1,..., componentN)
```

- First three parameters are used when the color profile can not be found, or when the document is displayed in XF Designer.
- The fourth parameter is a reference to a color profile declared with **fo:color-profile**.
- The fifth and next parameters are used to specify the color.

If your color profile is a Grayscale color profile, you will use only component1. If you have a RGB color profile, you must use three values, while for a CMYK color profile you must use four values. Each value is a floating point number and must be between 0.00 and 1.00 and represents the intensity of that color.

For a CMYK color profile you have to specify all four components.

Note:
- XF will output an error if the number of components you specify when using **rgb-icc** does not match the number of colors defined in the color profile.

You can also assign color profiles to images and SVG documents using **xf:color-profile**:

```
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format"
        xmlns:xf="http://www.ecrion.com/xf/1.0">
...
<fo:block>
<fo:external-graphic src="redsquare.svg" xf:color-profile="#CMYKColorProfile"/>
</fo:block>
```

Note:
- **xf** namespace must be declared; usually this is done as shown above, in the root element.

Blending Color Space

You may notice that PDF documents that contain transparent RGB images may not print correctly on CMYK printers. This happens because the mathematics of the blending are occurring the color space of the output device. If you are sending in elements in RGB, and the calculation is occurring in CMYK (the default of most printers), you will be getting poor results.

To correct this, you can explicitly set the Blending color space by using **xf:page-settings element**.

```
<xf:page-settings blending-color-space="DeviceNative | DeviceRGB | DeviceCMYK
                 |DeviceGray | custom"/>
```

To use an external ICC profile:

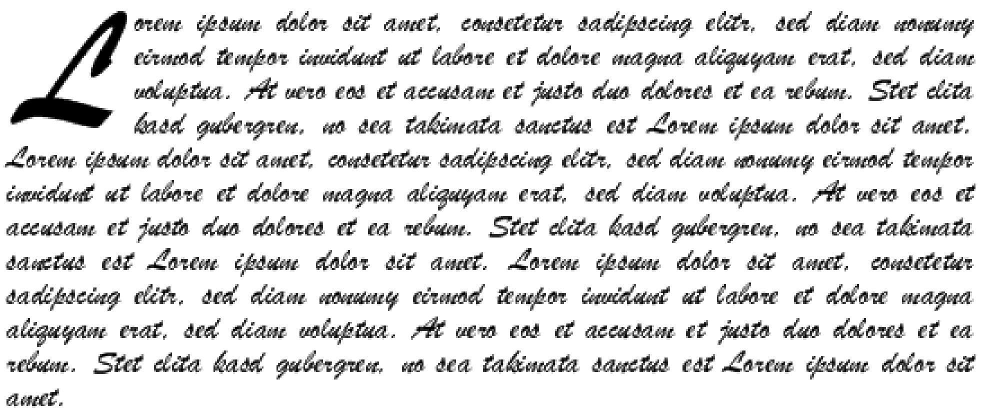
```
<fo:root>
<xf:page-settings blending-color-space="#CustomCMYKProfile"/>
<fo:declarations>
<fo:color-profile color-profile-name="CustomCMYKProfile"
             src="Adobe ICC Profiles\CMYK Profiles\EuroscaleCoated.icc"/>
</fo:declarations>
<fo:layout-master-set>
...
</fo:layout-master-set>
</fo:root>
```

1.7 Floats

fo:float element inserts an out-of-line block-level element such as a figure or a pull quote onto the page. The **float** property determines which side of the page it floats on and the clear property determines whether and where other elements are allowed to float around it.

```
<?xml version="1.0" encoding="utf-8"?>
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
   <fo:layout-master-set>
      <fo:simple-page-master master-name="all-pages" page-width="6in"
                                         page-height="2.8in">
            <fo:region-body region-name="xsl-region-body"
                                column-gap="0.25in" margin="0.2in"/>
      </fo:simple-page-master>
      <fo:page-sequence-master master-name="default-sequence">
            <fo:repeatable-page-master-reference master-reference="all-pages"/>
      </fo:page-sequence-master>
   </fo:layout-master-set>
   <fo:page-sequence master-reference="default-sequence">
      <fo:flow flow-name="xsl-region-body" font-family="Brush Script MT, Arial"
                                                          font-size="14pt">
            <fo:block text-align="justify"><fo:float float="start">
                  <fo:block font-size="72pt" line-height="1" margin="5pt"(1)
                                    text-depth="0">L</fo:block></fo:float>(2)
                  orem ipsum dolor sit amet, consetetur sadipscing elitr,sed diam
                  .............................................................
                  Stet clita kasd gubergren, no sea takimata sanctus est Lorem
                  ipsum dolor sit amet.
                  </fo:block>
      </fo:flow>
   </fo:page-sequence>
</fo:root>
```

The rendering result is displayed in the next figure:



Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

Figure 14.

There are several things to notice in this example:
(1) Each **fo:float** element should specify the floating side as either "**start**" or "**end**".
(2) Depending on the text you want to display, you may wish to eliminate the descent portion by setting the **text-depth** to 0.

1.8 Absolut Positioning

When an object is placed on a page, it can be positioned absolute or relative. Most objects are relative, which means that if the preceding objects grow/become larger, that the relative objects will shift (in most languages it shifts down).

We have seen that **XSL-FO** documents have flow layout, that is, the content flows from one page to the next one, according to the rules imposed by page breaks, spacing, widows and orphans properties. However, sometimes it may be useful to position elements at absolute coordinates.

Only **fo:block-container** can be placed absolutely, and this can be done by setting the **position** property to **'absolute'** or **'fixed'**.

The value **'fixed'** means that the object has a position relative to the page.
The value **'absolute'** means that the object has a position relative to the containing reference-area, typically another **fo:block-container**. This containing reference-area does not need to be positioned absolutely, which means you can position an object on a specific absolute location relative to another object that flows in the page.

In the example below, we have two fragments of text positioned under and over the main flow text:

```
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
   <fo:layout-master-set>
       <fo:simple-page-master master-name="p1" page-width="7in"
                                               page-height="2in">
             <fo:region-body region-name="xsl-region-body"
                                               margin="0.2in"/>
       </fo:simple-page-master>
       <fo:simple-page-master master-name="p2" page-width="7in"
                                               page-height="2in">
             <fo:region-body region-name="xsl-region-body" margin="0.2in"
                                               column-count="2"/>
       </fo:simple-page-master>
       <fo:page-sequence-master master-name="default-sequence">
             <fo:single-page-master-reference master-reference="p1"/>
             <fo:repeatable-page-master-reference master-reference="p2"/>
       </fo:page-sequence-master>
   </fo:layout-master-set>
   <fo:page-sequence master-reference="default-sequence">
       <fo:flow flow-name="xsl-region-body" font-family="Verdana"
                                                      font-size="10pt">
             <fo:block-container position="absolute" top="10pt" left="30pt"
                                               height="14pt" width="100%"> (1)
                   <fo:block font="72pt Arial" color="silver">Under</fo:block>
             </fo:block-container>
             <fo:block>
                   <fo:block>
                         Text Text Text Text Text Text Text Text Text Text
                         .............................................
                         Text Text Text Text Text Text Text Text Text Text
                   </fo:block>
                   <fo:block-container position="absolute" top="20pt" left="40pt"
                                                      height="14pt" width="100%">
                         <fo:block font="72pt Arial" color="red">Over</fo:block>
                   </fo:block-container>
             </fo:block>
             <fo:block break-before="page"/>
             <fo:block-container position="absolute" top="10pt" left="30pt"
                                                height="14pt" width="100%">
                    <fo:block font="72pt Arial" color="silver">Under</fo:block>
             </fo:block-container>
             <fo:block>
                    <fo:block>
                           Text Text Text Text Text Text Text Text Text Text
                           .................................................
                           Text Text Text Text Text Text Text Text Text Text
                    </fo:block>
                    <fo:block-container position="absolute" top="10pt" left="30pt"
                                                      height="14pt" width="100%">
                           <fo:block font="72pt Arial" color="red">Over</fo:block>
                    </fo:block-container>
             </fo:block>
       </fo:flow>
    </fo:page-sequence>
</fo:root>
```

The rendering result is displayed in the next figure:

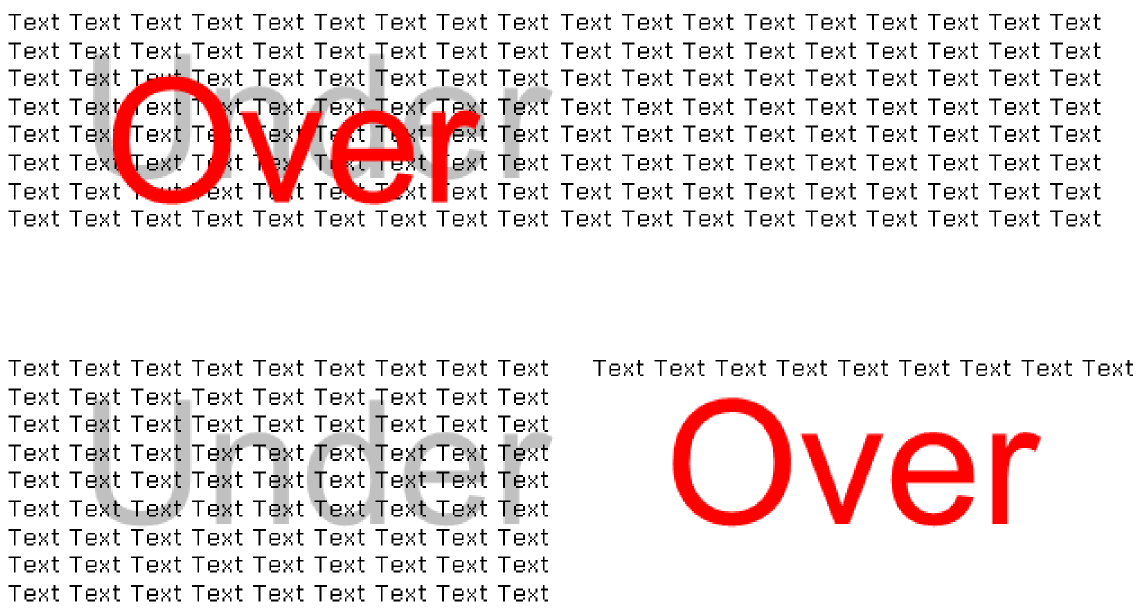


Figure 15.

The important point in this document is:
(1) position attribute is set to absolute ; top, bottom, left and right coordinates also are specified explicitly.

To change the order in which the elements are rendered, use **z-index** attribute:

```
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
       <fo:layout-master-set>
             <fo:simple-page-master master-name="p1" page-width="7in"
                                                     page-height="2in">
                    <fo:region-body region-name="xsl-region-body"
```

```
                                                     margin="0.2in"/>
             </fo:simple-page-master>
      </fo:layout-master-set>
      <fo:page-sequence master-reference="p1">
             <fo:flow flow-name="xsl-region-body" font-family="Verdana"
                                                     font-size="12pt">
                    <fo:block>
                          Before Before Before Before Before Before
                          .........................................
                          Before Before Before Before Before Before
                    </fo:block>
                    <fo:block-container z-index="-1" position="absolute"
                                                     top="38pt" left="0pt">
                          <fo:block font="48pt Arial" color="rgb(192,192,192)"
                                        text-align="center">Under</fo:block>
                    </fo:block-container>
                    <fo:block-container z-index="+1" position="absolute"
                                                     top="48pt" left="10pt">
                          <fo:block font="48pt Arial" color="red"
                                              text-align="center">Over</fo:block>
                    </fo:block-container>
                    <fo:block>
                          After After After After After After After After After
                          .....................................................
                          After After After After After After After After After
                    </fo:block>
             </fo:flow>
      </fo:page-sequence>
</fo:root>
```

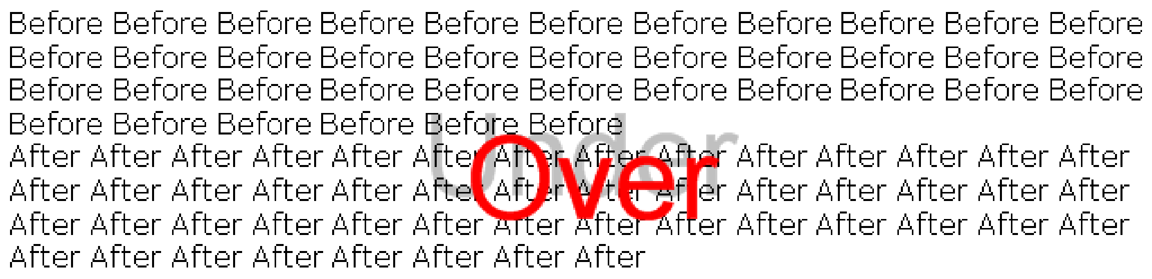

Before Before Before Before Before Before Before Before Before Before Before
Before Before Before Before Before Before Before Before Before Before Before
Before Before Before Before Before Before Before Before Before Before Before
Before Before Before Before Before Before
Under
Over
After After After After After After After After After After After After After After
After After After After After After After After After After After After After After
After After After After After After After After After After After After After After
After After After After After After After After

Figure 16.

All elements are considered to have a **z-index of 0**, therefore, if you want an element to be displayed on the background (like a watermark) use a **negative z-index**, while for a foreground element (like a stamp) use a **positive z-index**.

1.9 Tables

Tables are described in XSL-FO using **fo:table** element. A table can have a header (**fo:tableheader**), a body (**fo:table-body**) and a footer (**fo:table-footer**).

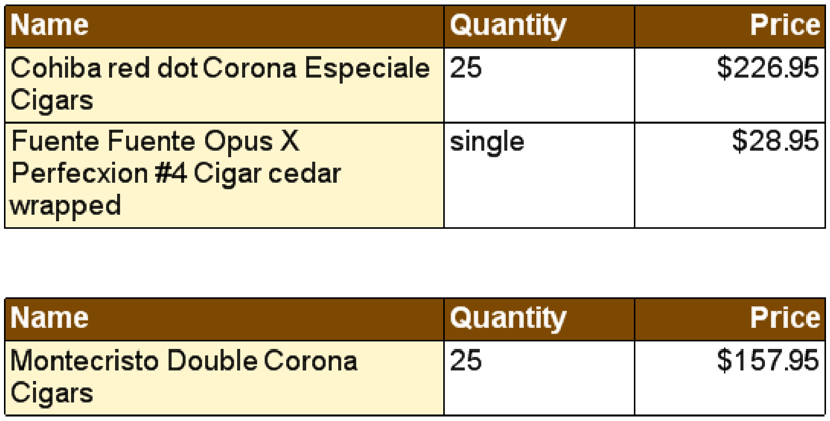
Each of these groups contain rows (**fo:table-row**), which in turn contain cells (**fo:table-cell**).

The columns are described using **fo:table-column** elements.

```
...
<fo:table border-collapse="collapse" font-size="14pt" font-family="Arial"> (1)
      <fo:table-column column-width="3in" background-color="rgb(255,246,206)"/> (2)
      <fo:table-column column-width="50%"/> (3)
      <fo:table-column column-width="50%"/>
      <fo:table-header color="rgb(255,255,255)" background-color="rgb(125,73,2)"
                      font-weight="bold"> (4)
            <fo:table-row>
                  <fo:table-cell padding="2pt" border="1pt solid black">
                        <fo:block>
                              Name
                        </fo:block>
                  </fo:table-cell>
                  <fo:table-cell padding="2pt" border="1pt solid black">
                        <fo:block>
                              Quantity
                        </fo:block>
                  </fo:table-cell>
                  <fo:table-cell padding="2pt" border="1pt solid black">
                        <fo:block text-align="right">
                              Price
                        </fo:block>
                  </fo:table-cell>
                  </fo:table-row>
      </fo:table-header>
      <fo:table-body> (5)
            <fo:table-row>
                  <fo:table-cell padding="2pt" border="1pt solid black"> (6)
                        <fo:block>Cohiba red dot Corona Especiale Cigars
                        </fo:block>
                  </fo:table-cell>
                  <fo:table-cell padding="2pt" border="1pt solid black">
                        <fo:block>25</fo:block>
                  </fo:table-cell>
                  <fo:table-cell padding="2pt" border="1pt solid black"
                                          text-align="right">
                        <fo:block>$226.95</fo:block>
                  </fo:table-cell>
            </fo:table-row>
            <fo:table-row>
                  <fo:table-cell padding="2pt" border="1pt solid black">
                  <fo:block>Fuente Fuente Opus X Perfecxion #4 Cigar cedar wrapped
                  </fo:block>
                  </fo:table-cell>
                  <fo:table-cell padding="2pt" border="1pt solid black">
                        <fo:block>single</fo:block>
                  </fo:table-cell>
                  <fo:table-cell padding="2pt" border="1pt solid black"
                                                text-align="right">
                        <fo:block>$28.95</fo:block>
```

```
                    </fo:table-cell>
             </fo:table-row>
             <fo:table-row>
                    <fo:table-cell padding="2pt" border="1pt solid black">
                           <fo:block>Montecristo Double Corona Cigars</fo:block>
                    </fo:table-cell>
                    <fo:table-cell padding="2pt" border="1pt solid black">
                           <fo:block>25</fo:block>
                    </fo:table-cell>
                    <fo:table-cell padding="2pt" border="1pt solid black"
                                                      text-align="right">
                           <fo:block>$157.95</fo:block>
                    </fo:table-cell>
             </fo:table-row>
       </fo:table-body>
</fo:table>
```

The result of rendering is displayed in following figure:



Name	Quantity	Price
Cohiba red dot Corona Especiale Cigars	25	$226.95
Fuente Fuente Opus X Perfecxion #4 Cigar cedar wrapped	single	$28.95

Name	Quantity	Price
Montecristo Double Corona Cigars	25	$157.95

Figure 17.

Things to notice:

(1) The **fo:table** element is defined. This table has the **border-collapse** attribute set to "collapse", which will cause cell borders to merge.
(2),(3) Columns can have either a fixed width or a percentual width.
(4),(5) We define table's header and body. If a page break will occurs, the headers and the footers are displayed on the next page as well.
(6) Each **fo:table-cell** can span multiple rows and/or columns.

The content of the cell is aligned vertically according to **display-align** property.

Note:
- By default a cell will not clip it's content. To clip the cell's content set overflow attribute to hidden.

The next sample illustrates these attributes:

```
...
<fo:table border-collapse="collapse" font-size="14pt" font-family="Arial">
      <fo:table-column column-width="50%"/>
      <fo:table-column column-width="50%"/>
      <fo:table-body>
            <fo:table-row>
                  <fo:table-cell border="1pt solid black" height="2.4cm"
                  overflow="hidden" display-align="center" text-align="center">
                        <fo:block font-size="48pt" color="red">Clipped Cell
                        </fo:block>
            </fo:table-cell>
                  <fo:table-cell border="1pt solid black" display-align="center">
                        <fo:block>Normal table cell.</fo:block>
                  </fo:table-cell>
            </fo:table-row>
      </fo:table-body>
</fo:table>
...
```

The rendering result is displayed in the next figure:

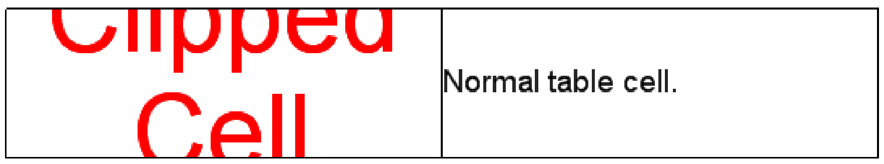


Figure 18.

This example also shows how to create fixed height rows by using **height** attribute.

Table Columns

As noted in the example above, a column can have a proportional width or a fixed width.
A fixed width includes length units (in, pt, cm; for example<**fo:table-column column-width="3in"**/>).
A proportional width is expressed via proportional-column-width function (for example <**fo:table-column column-width**= "**proportional-column-width(20)**"/>) or by using a percentage sign (<**fo:table-column column-width="20%"**/>).

There is a third way to specify a column width: by omitting the **column-width** attribute, the column will size itself automatically, depending on it's content.
A table can mix fixed, proportional and automatic columns. When a table contains only proportional columns, XF will resize them even if the sum of percentages is not 100.

For example:

```
<fo:table>
<fo:table-column column-width="50%"/>
<fo:table-column column-width="50%"/>
..
```

```
</fo:table>
```

and

```
<fo:table>
<fo:table-column column-width="proportional-column-width(1)"/>
<fo:table-column column-width="proportional-column-width(1)"/>
..
</fo:table>
```

and

```
<fo:table>
<fo:table-column column-width="proportional-column-width(60)"/>
<fo:table-column column-width="proportional-column-width(60)"/>
..
</fo:table>
```

will produce the same result.

When fixed columns are included, XF will layout first the fixed columns, then the remaining space is distributed between proportional columns, according with **column-width** attribute.

When a table includes automatic columns, XF will first layout the fixed columns, then it will calculate the width percentages of the automatic columns based on their contents. The automatic columns have now a percentage width, and the remaining space is distributed between these columns and the proportional columns that have a **column-width** attribute.

If a column is removed, if the table still have at least one proportional column, the width will be distributed. If the table had only fixed columns, the total table width will be reduced.

Indentation

Consider the following example:

```
...
<fo:block>
      <fo:table start-indent="1in" background-color="rgb(255,255,255)">
            <fo:table-column column-width="proportional-column-width(1)"
                                                column-number="1"/>
            <fo:table-column column-width="proportional-column-width(1)"
                                                column-number="2"/>
            <fo:table-body>
                  <fo:table-row>
                        <fo:table-cell>
                              <fo:block background-color="rgb(153,204,255)">
                              Cell 1 content
                        </fo:block>
                        </fo:table-cell>
                        <fo:table-cell>
                              <fo:block background-color="rgb(153,204,255)">
                              Cell 2 content
                        </fo:block>
                        </fo:table-cell>
                  </fo:table-row>
            </fo:table-body>
      </fo:table>
</fo:block>
...
```

You can notice that although **start-indent** is specified only for **fo:table**, not only the table gets indented, but the child **fo:block** are indented as well.
start-indent is the trickiest property in the XSL-FO specs because it behaves against what most developers find normal: you would expect that the block being indented to move to the left and with it, its children (in our example, the table and it's cell).

The different behaviour is however correct, because:

- **start-indent** is an inheritable property. It is like you specify the same start indent for the cell's content.
- **fo:table-cell** elements establish a "**reference view-port area**", that is, a reference point from where the indentation is calculated.

To indent only the table, you can either:

- Use **margin-left** instead of **start-indent**.
- Use **start-indent** for the table, and then use-it again for each table-cell but set it's value to 0.

1.10 Lists

XSL-FO lists are created using **fo:list-block** element. A list can contain one or more items: **fo:list- item**. Each item has a label: **fo:list-item-label** usually used to display a bullet or a number, and a body: **fo:list-item-body**.

```
...
<fo:block>To do list:</fo:block>
<fo:list-block> (1)
      <fo:list-item> (2)
            <fo:list-item-label end-indent="label-end()"> (3)
                  <fo:block>
                        1)
                  </fo:block>
            </fo:list-item-label>
            <fo:list-item-body start-indent="body-start()"> (4)
                  <fo:block>
                        Very very important stuff
                  </fo:block>
            </fo:list-item-body>
      </fo:list-item>
      <fo:list-item>
            <fo:list-item-label end-indent="label-end()">
                  <fo:block>
                        2)
                  </fo:block>
            </fo:list-item-label>
            <fo:list-item-body start-indent="body-start()">
                  <fo:block>
                        Very important stuff
                  </fo:block>
            </fo:list-item-body>
      </fo:list-item>
      <fo:list-item>
            <fo:list-item-label end-indent="label-end()">
                  <fo:block>
                        3)
                  </fo:block>
            </fo:list-item-label>
            <fo:list-item-body start-indent="body-start()">
                  <fo:block>
                        Other important items
                  </fo:block>
            </fo:list-item-body>
      </fo:list-item>
      <fo:list-item>
            <fo:list-item-label end-indent="label-end()">
                  <fo:block>
                        4)
                  </fo:block>
            </fo:list-item-label>
            <fo:list-item-body start-indent="body-start()">
                  <fo:block>
                        Don't forget to eat
                  </fo:block>
            </fo:list-item-body>
      </fo:list-item>
      <fo:list-item>
            <fo:list-item-label end-indent="label-end()">
```

```
                    <fo:block>
                          5)
                    </fo:block>
              </fo:list-item-label>
              <fo:list-item-body start-indent="body-start()">
                    <fo:block>
                          Sleep would be good
                    </fo:block>
              </fo:list-item-body>
      </fo:list-item>
</fo:list-block>
```

The rendering result is displayed in the next figure:

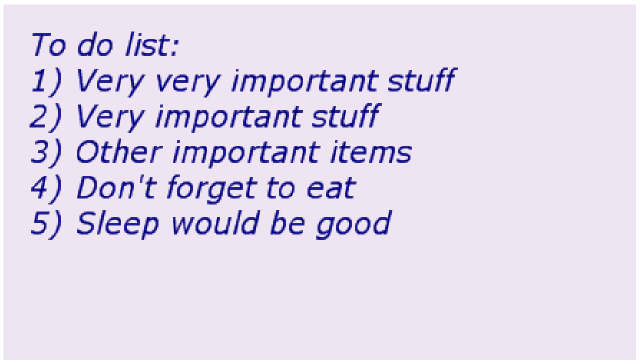

To do list:
1) Very very important stuff
2) Very important stuff
3) Other important items
4) Don't forget to eat
5) Sleep would be good

Figure 19.

The important points in this document are:
(1) The list is created using **fo:list-block**.
(2) A list can have one or multiple **items**.
(3),**(4)** Each item has a **label**, usually used to display a bullet or number, and a **body**.

Numbered Lists

XSL-FO does not provide an element to create numbered lists like HTML does; you have to generate the numbers using XSL techniques.
Considering the following source XML document:

```
<products>
<product>Fuji FinePix F700</product>
<product>Nikon CoolPix 5700</product>
<product>Cannon Powershot A310</product>
</products>
```

We can create an XSL template that uses **xsl:number** element to generate numbers for each **fo:list-item-label**:

```
<?xml version="1.0" encoding="utf-8"?>
<?xsl-test-case type="text/xml" href=".\Numbered List.xml"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
                                                     version="1.0">
      <xsl:output method="xml" encoding="utf-8" indent="yes"/>
      <xsl:template match="/">
            <fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
                  <fo:layout-master-set>
                        <fo:simple-page-master master-name="all"
                              page-width="5in" page-height="1in">
                              <fo:region-body region-name="xsl-region-body"
                                                        margin="0.2in"/>
                        </fo:simple-page-master>
                  </fo:layout-master-set>
                  <fo:page-sequence master-reference="all">
                     <fo:flow flow-name="xsl-region-body" font="10pt Arial">
                        <fo:list-block>
                            <xsl:for-each select="products/product">
                              <fo:list-item>
                                  <fo:list-item-label end-indent="label-end()">
                                     <fo:block>
                                          <xsl:number/>
                                     </fo:block>
                                  </fo:list-item-label>
                                  <fo:list-item-body start-indent="body-start()">
                                      <fo:block>
                                          <xsl:value-of select="."/>
                                      </fo:block>
                                  </fo:list-item-body>
                              </fo:list-item>
                            </xsl:for-each>
                        </fo:list-block>
                      </fo:flow>
                  </fo:page-sequence>
            </fo:root>
      </xsl:template>
</xsl:stylesheet>
```

1.11 Pagination

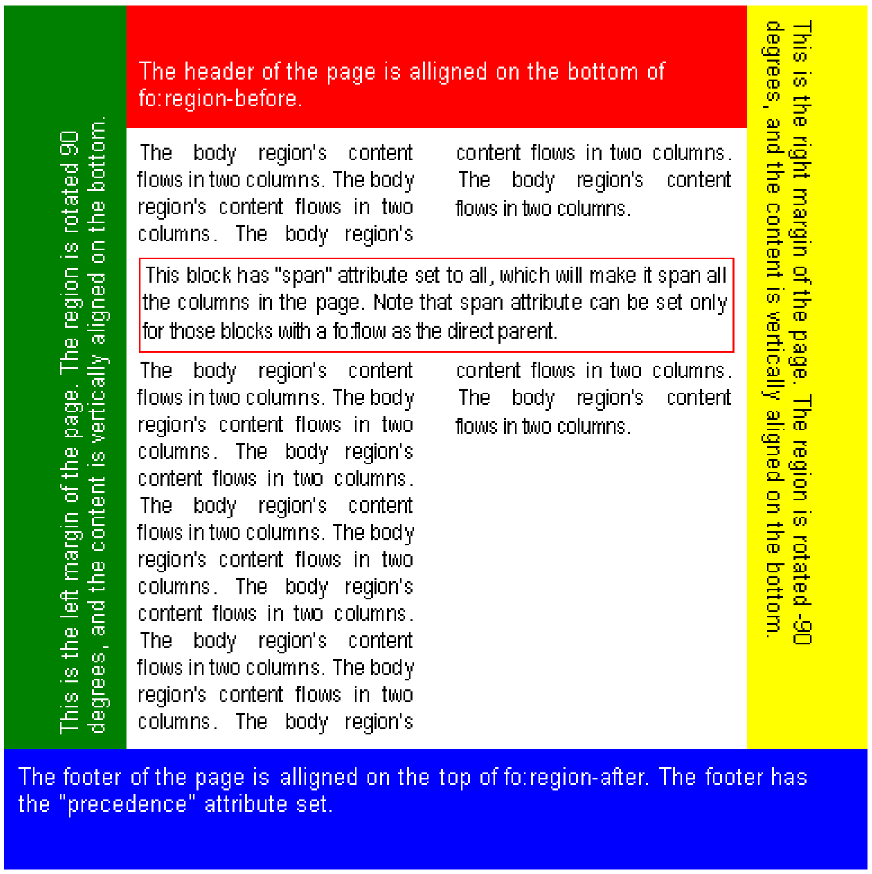
Every page has the following regions:
- **fo:region-body** which holds the main page content, that is, the content of fo:flow
- **fo:region-before**, used to display headers
- **fo:region-after**, used to display footers
- **fo:region-start** and **fo:region-end**, used to display side regions

Of all regions, **fo:region-body** can have multiple columns:

```
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
   <fo:layout-master-set>
      <fo:simple-page-master master-name="all-pages" page-width="5in"
                                                    page-height="5in">
            <fo:region-body region-name="Content" margin="0.7in"
                  column-gap="0.25in" padding="6pt" column-count="2"/>
            <fo:region-before region-name="Header" extent="0.7in"
            display-align="after" padding="6pt" background-color="red"/>
            <fo:region-after region-name="Footer" extent="0.7in"
            display-align="before" padding="6pt" background-color="blue"
            precedence="true"/>
            <fo:region-start region-name="LeftMargin" extent="0.7in"
            padding="6pt" background-color="green"reference-orientation="90"
            display-align="after"/>
            <fo:region-end region-name="RightMargin" extent="0.7in"
            padding="6pt" background-color="yellow" reference-orientation="-90"
            display-align="after"/>
      </fo:simple-page-master>
      <fo:page-sequence-master master-name="default-sequence">
            <fo:repeatable-page-master-reference master-reference="all-pages"/>
      </fo:page-sequence-master>
   </fo:layout-master-set>
   <fo:page-sequence master-reference="default-sequence">
      <fo:static-content flow-name="Header" font-family="Arial"
                          font-size="10pt" color="white">
            <fo:block>
            The header of the page is aligned on the bottom of fo:region-before.
            </fo:block>
      </fo:static-content>
      <fo:static-content flow-name="Footer" font-family="Arial"
                          font-size="10pt" color="white">
            <fo:block>
               The footer of the page is aligned on the top of fo:region-after.
               The footer has the "precedence" attribute set.
            </fo:block>
      </fo:static-content>
      <fo:static-content flow-name="LeftMargin" font-family="Arial"
                                       font-size="10pt" color="white">
            <fo:block>
               This is the left margin of the page. The region is rotated 90
               degrees, and the content is vertically aligned on the bottom.
            </fo:block>
      </fo:static-content>
      <fo:static-content flow-name="RightMargin" font-family="Arial"
                                                    font-size="10pt">
            <fo:block>
               This is the right margin of the page. The region is rotated -90
               degrees, and the content is vertically aligned on the bottom.
            </fo:block>
```

```
       </fo:static-content>
       <fo:flow flow-name="Content" font-family="Arial Narrow" font-size="10pt"
                                                      text-align="justify">
             <fo:block>
                    The body region's content flows in two columns.
                    ...............................................
                    The body region's content flows in two columns.
             </fo:block>
             <fo:block span="all" border="1pt solid red" padding="2pt"
                    space-before="3pt" space-after="3pt" start-indent="2pt"
                                                      end-indent="2pt">
                    This block has "span" attribute set to all, which will make
                    it span all the columns in the page. Note that span attribute
                    can be set only for those blocks with a fo:flow as the direct
                    parent.
             </fo:block>
             <fo:block>
                    The body region's content flows in two columns.
                    ...............................................
                    The body region's content flows in two columns.
             </fo:block>
       </fo:flow>
   </fo:page-sequence>
</fo:root>
```

The rendering result is displayed in the next figure:



The header of the page is alligned on the bottom of fo:region-before.

This is the left margin of the page. The region is rotated 90 degrees, and the content is vertically aligned on the bottom.

The body region's content flows in two columns. The body region's content flows in two columns. The body region's content flows in two columns. The body region's content flows in two columns.

This block has "span" attribute set to all, which will make it span all the columns in the page. Note that span attribute can be set only for those blocks with a fo:flow as the direct parent.

The body region's content flows in two columns. The body region's content flows in two columns. The body region's content flows in two columns. The body region's content flows in two columns. The body region's content flows in two columns. The body region's content flows in two columns. The body region's content flows in two columns. The body region's content flows in two columns. The body region's content flows in two columns.

This is the right margin of the page. The region is rotated -90 degrees, and the content is vertically aligned on the bottom.

The footer of the page is alligned on the top of fo:region-after. The footer has the "precedence" attribute set.

Figure 20.

Every sequence of pages generated by the XSL-FO engine can have one or more page layouts associated with it:

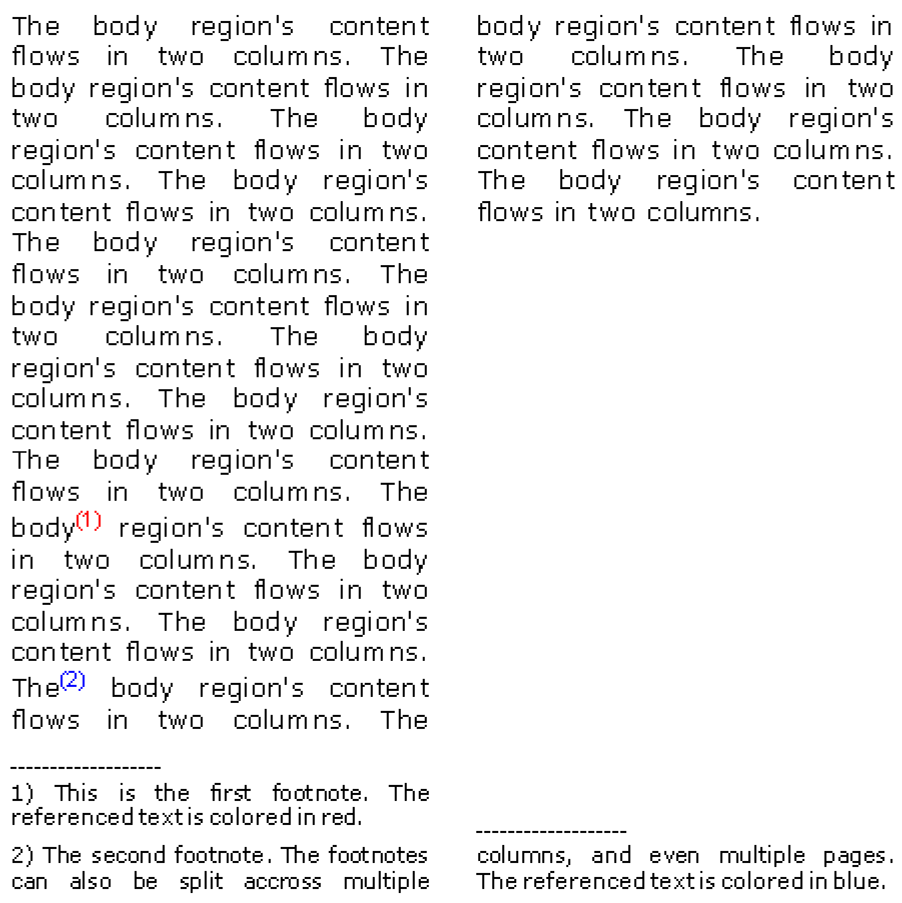
a) The simples scenario: only one page layout for the whole document. All documents described so far belong to this category.
c) Different page layouts for the first and subsequent pages, for the case when you want a cover page formatted differently than the rest of the pages.
b) Different page layouts for even and odd pages, as it happens with most printed books, where the inside margin of a page is slightly larger than the outside margin, to allow binding.

1.12 Footnotes

A **footnote** is an out-of-line object composed from two elements: a **fo:inline** used to insert a symbol in the document and a **fo:footnote-body** containing the text that the symbol is referring to.

```
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
   <fo:layout-master-set>
       <fo:simple-page-master master-name="all-pages" page-width="5in"
                                                     page-height="5in">
             <fo:region-body region-name="Content" margin="0.2in"
                                 column-gap="0.25in" column-count="2"/> (1)
       </fo:simple-page-master>
       <fo:page-sequence-master master-name="default-sequence">
             <fo:repeatable-page-master-reference master-reference="all-pages"/>
       </fo:page-sequence-master>
   </fo:layout-master-set>
   <fo:page-sequence master-reference="default-sequence">
       <fo:static-content flow-name="xsl-footnote-separator" font-family="Arial"
                                                     font-size="10pt"> (2)
             <fo:block>
                   -------------------
             </fo:block>
       </fo:static-content>
       <fo:flow flow-name="Content" font-family="Verdana" font-size="10pt"
                                                     text-align="justify">
             <fo:block>
                   The body region's content flows in two columns.
                   ..............................................
                   The body region's content flows in two columns.
                   The body<fo:footnote><fo:inline baseline-shift="super"
                                             font-size="8pt" color="red">(3)
                                             (1)</fo:inline>
                   <fo:footnote-body><fo:block font="8pt Verdana"
                                 text-align="justify" space-after="5pt">(4)
                                       1)
                                       This is the first footnote. The referenced
                                       text is colored in red.
                                 </fo:block></fo:footnote-body></fo:footnote>
                                 region's content flows in two columns.
                   The body region's content flows in two columns.
                   The body region's content flows in two columns.
                   The<fo:footnote><fo:inline baseline-shift="super"
                                 font-size="8pt" color="blue">(2)</fo:inline>
                   <fo:footnote-body>
                         <fo:block font="8pt Verdana" text-align="justify">
                                       2)
                                       The second footnote.
                                       The footnotes can also be split across
                                       multiple columns, and even multiple pages.
                                       The referenced text is colored in blue.
                         </fo:block></fo:footnote-body></fo:footnote>
                                 body region's content flows in two columns.
                   The body region's content flows in two columns.
                   ..............................................
                   The body region's content flows in two columns.
             </fo:block>
       </fo:flow>
   </fo:page-sequence>
</fo:root>
```

The rendering result is displayed in the next figure:



The body region's content flows in two columns. The body region's content flows in two columns. The body region's content flows in two columns. The body region's content flows in two columns. The body region's content flows in two columns. The body region's content flows in two columns. The body region's content flows in two columns. The body region's content flows in two columns. The body region's content flows in two columns. The body(1) region's content flows in two columns. The body region's content flows in two columns. The body region's content flows in two columns. The(2) body region's content flows in two columns. The body region's content flows in two columns. The body region's content flows in two columns. The body region's content flows in two columns. The body region's content flows in two columns.

----

1) This is the first footnote. The referenced text is colored in red.

2) The second footnote. The footnotes can also be split accross multiple

----

columns, and even multiple pages. The referenced text is colored in blue.

Figure 21.

The important points in this document are the following:

(1) We set the number of columns for the main flow to 2.
(2) A special area ("**xsl-footnote-separator**" is a reserved name) is created to hold the separator between the footnotes and the document body. This region is optional.
(3) The footnote object is declared, inline with the text. The first child element, **fo:inline** is used to format the number/symbol of the footnote.
(4) Footnote's body is defined.

1.13 Markers

You have probably noticed how in a book, the current chapter name is displayed on the header. To implement this feature, you need to understand **XSL-FO marker** elements.

First you "**mark**" (delimitate) pieces of your content as being retrievable for the purpose of displaying them in headers or footers.
For this you use **fo:marker**. The marker is usually associated with a **fo:block**, therefore, the information from the marker can be displayed in all that pages where the fo:block generates areas.

Then in the headers or footer you tell the engine to display a marker using **fo:retrieve-marker**.

There can be of course multiple markers in a certain page, so you have to use an unique name for each marker, as well as the retrieval rule: first marker with the given name present in the page, or the first that starts in the page, or the last one, etc.

The next example shows a two chapter document, with the chapter title being displayed in the header.

```
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format"
      xmlns:rx="http://www.renderx.com/XSL/Extensions">
   <fo:layout-master-set>
      <fo:simple-page-master master-name="all-pages" page-width="5in"
                             page-height="1.2in" margin="0.1in">
            <fo:region-body region-name="xsl-region-body" margin-top="0.14in"/>
            <fo:region-before region-name="xsl-region-before" extent="0.12in"
               padding-right="0.2in" border-bottom="1pt solid black"
                                             display-align="after"/>
      </fo:simple-page-master>
   </fo:layout-master-set>
   <fo:page-sequence master-reference="all-pages">
      <fo:static-content flow-name="xsl-region-before" font="italic 10pt'Verdana'">
            <fo:block>
                  The first title starting on this page:
                  <fo:retrieve-marker retrieve-class-name="title"
                  retrieve-position="first-starting-within-page" (1)
                                             retrieve-boundary="page"/>
            </fo:block>
      </fo:static-content>
      <fo:flow flow-name="xsl-region-body" font="10pt Arial">
            <fo:block font-weight="bold"><fo:marker marker-class-name="title">(2)
                        Title of Chapter 1
                  </fo:marker>
                  Chapter 1
            </fo:block>
            <fo:block widows="1"> (3)
                  Text text text text text text text text text text text text
                  text text text text text text text text text text text text
                  ...........................................................
                  text text text text text text text text text text text text
            </fo:block>
            <fo:block font-weight="bold"><fo:marker marker-class-name="title">(4)
                        Title of Chapter 2
                  </fo:marker>
                  Chapter 2
            </fo:block>
            <fo:block>Text text text, text</fo:block>
      </fo:flow>
   </fo:page-sequence>
</fo:root>
```

The rendering result is displayed in the next figure:

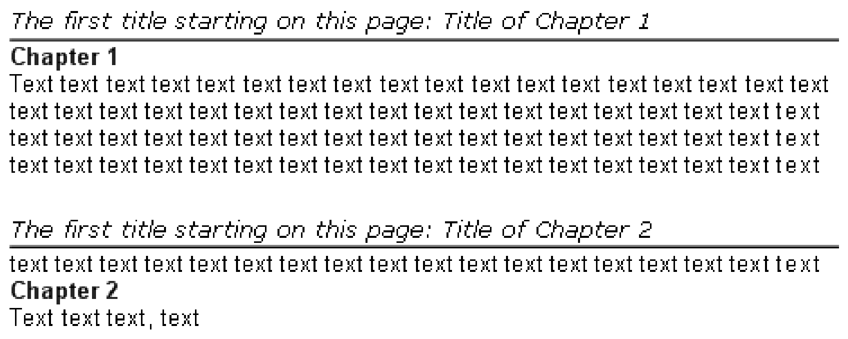

The first title starting on this page: Title of Chapter 1
Chapter 1
Text text text text text text text text text text text text text text text text text text
text text text text text text text text text text text text text text text text text text
text text text text text text text text text text text text text text text text text text
text text text text text text text text text text text text text text text text text text
The first title starting on this page: Title of Chapter 2
text text text text text text text text text text text text text text text text text text
Chapter 2
Text text text, text

Figure 22.

The important points in this document are:

(2),(4) In the body of the page we declare **a marker for each fo:block** in the flow.
(3) The second paragraph will go on the second page because it has the **widows** attribute set.
(1) In the header region, that applies for every generated page, we retrieve the markers using **fo:retrieve-marker**. The marker "**type**" is specified in **retrieve-class-name** attribute. Limit the scope of the marker retrieval to be in the same page as **retrieve-marker** element. First occurrence will be displayed.

1.14 Bookmarks

XF Rendering Server implements the bookmarks as defined by the latest W3C Working Draft for XSL-FO 1.1.

The **fo:bookmark-tree** formatting object is used to hold list of access points within the document such as a table of contents, a list of figures or tables, etc.

Each access point is called a bookmark.
The **fo:bookmark** object is used to identify an access point, and to specify where that access point is within the current document or another external document.

A given bookmark may be further subdivided into a sequence of (sub-)bookmarks to as many levels as the authors desire:

```
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
      <fo:layout-master-set>
            <fo:simple-page-master master-name="LetterPage" margin="1in">
                  <fo:region-body region-name="PageBody"/>
            </fo:simple-page-master>
      </fo:layout-master-set>
      <fo:bookmark-tree>
            <fo:bookmark internal-destination="toc">
                  <fo:bookmark-title>Bookmarks Example</fo:bookmark-title>
                  <fo:bookmark internal-destination="chapter1">
                        <fo:bookmark-title>Hello World</fo:bookmark-title>
                  </fo:bookmark>
                  <fo:bookmark internal-destination="chapter2">
                        <fo:bookmark-title>Paragraphs</fo:bookmark-title>
                  </fo:bookmark>
            </fo:bookmark>
      </fo:bookmark-tree>
      <fo:page-sequence master-reference="LetterPage" font="10pt Arial">
            <fo:flow flow-name="PageBody" font-family="Arial Narrow"
                                                  font-size="10pt">
                  <fo:block id="toc">Table Of Contents</fo:block>
                  <fo:block text-align-last="justify">
                        <fo:basic-link color="blue"
                  internal-destination="chapter1">Hello World</fo:basic-link>
                        <fo:inline keep-together.within-line="always">
                              <fo:leader leader-pattern="dots"/>
                              <fo:page-number-citation ref-id="chapter1"/>
                        </fo:inline>
                  </fo:block>
                  <fo:block text-align-last="justify">
                        <fo:basic-link color="blue"
                  internal-destination="chapter2">Paragraphs</fo:basic-link>
                        <fo:inline keep-together.within-line="always">
                              <fo:leader leader-pattern="dots"/>
                              <fo:page-number-citation ref-id="chapter2"/>
                        </fo:inline>
                  </fo:block>
                  <fo:block id="chapter1" break-before="page" font-size="18pt">
                        Hello World
                  </fo:block>
                  <fo:block>
                        Text text text text text text text text text text text
                  </fo:block>
```

```
                    <fo:block id="chapter2" break-before="page" font-size="18pt">
                          Paragraphs
                    </fo:block>
                    <fo:block>
                          Text text text text text text text text text text text
                    </fo:block>
             </fo:flow>
      </fo:page-sequence>
</fo:root>
```

The bookmarks displayed in Acrobat Reader can be used to navigate the PDF file.

1.15 Miscellaneous Inline Elements

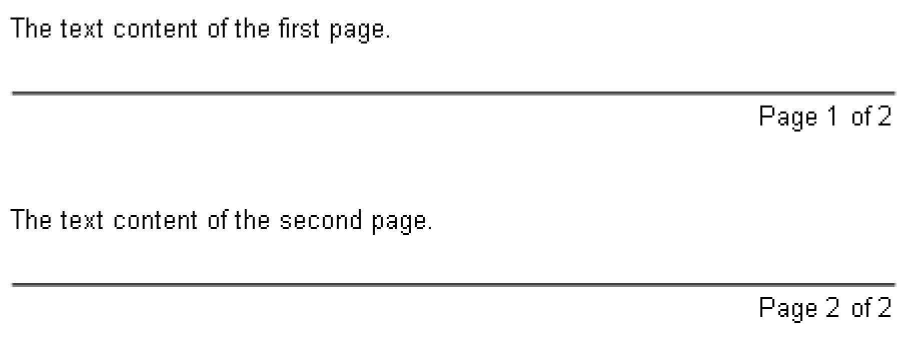
There are several inline elements left to describe:

Page Numbers

- **fo:page-number** is used to insert the current page number
- **fo:page-number-citation** is used to retrieve the page number of a give element. This element is also useful in inserting the number of pages in a document, as shown below:

```
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
   <fo:layout-master-set>
       <fo:simple-page-master master-name="LetterPage" page-width="5in"
                                        page-height="1in" margin="0.2in">
             <fo:region-body region-name="PageBody"/>
             <fo:region-after region-name="Footer" extent="0.2in"/>
       </fo:simple-page-master>
   </fo:layout-master-set>
   <fo:page-sequence master-reference="LetterPage" font="10pt Arial">
       <fo:static-content flow-name="Footer">
             <fo:block text-align="right" border-top="1pt solid black"
                                                padding-top="1mm">
                    Page
                    <fo:page-number/>
                    of
                    <fo:page-number-citation ref-id="theEnd"/>
             </fo:block>
       </fo:static-content>
       <fo:flow flow-name="PageBody">
             <fo:block>
                    The text content of the first page.
             </fo:block>
             <fo:block break-before="page">
                    The text content of the second page.
             </fo:block>
             <fo:block id="the End"/>
       </fo:flow>
   </fo:page-sequence>
</fo:root>
```

The rendering result is displayed in the next figure:


The text content of the first page.

Page 1 of 2

The text content of the second page.

Page 2 of 2

Figure 23.

To start the numbering from a different page number use **initial-page-number** attribute of **fo:page-sequence**.

Hyperlinks

fo:basic-link can be used to display hyperlinks in a document, either to an external document, or as a cross reference within the current document.

As opposed to HTML, this element, does not underline the text or sets the text the color to blue; it just simply marks the area as being active.
You must use the standard properties like color and text-decoration to simulate the aspect of HTML hyperlinks.

```
...
<fo:block>
      Hyperlink to an external resource:
      <fo:basic-link color="blue" text-decoration="underline"
                          external-destination="url(http://www.ecrion.com)">
      Ecrion Home
      </fo:basic-link>
</fo:block>
...
```

The rendering result is displayed in the next figure:

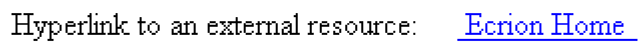


Figure 24.

Leaders

fo:leader is a more complicated version of HTML's rule element.

In the next example we will display a dotted leader in a table of contents:

```
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
   <fo:layout-master-set>
      <fo:simple-page-master master-name="LetterPage" page-width="5in"
                                        page-height="0.8in" margin="0.1in">
            <fo:region-body region-name="PageBody"/>
      </fo:simple-page-master>
   </fo:layout-master-set>
   <fo:page-sequence master-reference="LetterPage" font="10pt Arial">
      <fo:flow flow-name="PageBody" font-family="Arial Narrow" font-size="10pt">
            <fo:block>Table Of Contents</fo:block>
            <fo:block text-align-last="justify">
                  <fo:basic-link color="blue" internal-destination="chapter1">
                                          Hello World</fo:basic-link>
                  <fo:inline keep-together.within-line="always">
                        <fo:leader leader-pattern="dots"/>
                        <fo:page-number-citation ref-id="chapter1"/>
                  </fo:inline>
```

```
             </fo:block>
             <fo:block text-align-last="justify">
                    <fo:basic-link color="blue" internal-destination="chapter2">
                                               Paragraphs</fo:basic-link>
                    <fo:inline keep-together.within-line="always">
                           <fo:leader leader-pattern="dots"/>
                           <fo:page-number-citation ref-id="chapter2"/>
                    </fo:inline>
             </fo:block>
             <fo:block id="chapter1" break-before="page" font-size="18pt">
                    Hello World
             </fo:block>
             <fo:block>
                    Text text text text text text text text text text text
             </fo:block>
             <fo:block id="chapter2" break-before="page" font-size="18pt">
                    Paragraphs
             </fo:block>
             <fo:block>
                    Text text text text text text text text text text text
             </fo:block>
      </fo:flow>
   </fo:page-sequence>
</fo:root>
```

The rendering result is displayed in the next figure:

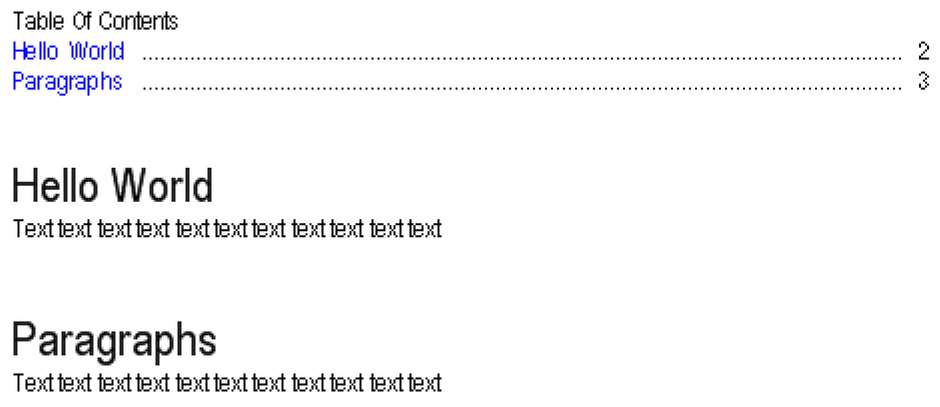


Table Of Contents
Hello World 2
Paragraphs 3

Hello World
Text text text text text text text text text text text

Paragraphs
Text text text text text text text text text text text

Figure 25.

1.16 Extensions

Apart from the features that are built in to XSL-FO, there are of course other features that are not a part of XSL-FO, but can be used in combination with XSL-FO.

For example, if documents need to be printed on paper, you might need some settings as duplex/simplex or input and output paper trays.
Some output streams, such as AFP, have even more concepts, like AFP overlays that define the 'background' of a page, but in a very specific manner.

These things are not included in XSL-FO. They are so specific for a certain application that they're hard to standardize. For this and many other needs, implementation support **XSL-FO extensions**.

Here are some extensions to XSL-FO:

1.16.1 Index Entries

There are two main steps in creating indexes:
• Mark words, phrases or whole blocks.
• Insert page indexes.

Marking

Use **xf:key** attribute for any element that can have an id. While an id should always be unique, xf:key values may not be unique. All occurences of a specific key will participate in generating the final index.

```
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format"
                          xmlns:xf="http://www.ecrion.com/xf/1.0">
      ...
      <fo:block xf:key="keywords.lion.range">
            <fo:inline xf:key="keywords.lion">Lions</fo:inline>,
            along with the other big cats such as tigers are in the genus Panthera.
      ...
      </fo:block>
      <fo:block>
            Some <fo:inline xf:key="keywords.lion">lions</fo:inline> are nomadic.
      </fo:block>
      ...
</fo:root>
```

In the example above, there are two distinct key values: **keywords.lion.range** and **keywords.lion**.

Note:
• The string value of key attribute can be anything, but for clarity purposes, we use a dotted notation in the examples presented in this chapter.

Page Indexes

To insert a list of pages corresponding to a index key, use **xf:page-index**.

```
<xf:page-index xmlns:xf="http://www.ecrion.com/xf/1.0"/
      ref-key="string"
      list-separator="string"
      range-separator="string">
```

• **list-separator** represents the separator between non consecutive page numbers; the default value is ", ".
• **range-separator** represents the separator between the first and last pages in a range; the default value is "-".

```
...
<fo:block>
      lions <xf:page-index ref-key="keywords.lion"></xf:page-index>
</fo:block>
<fo:block>
      range <xf:page-index ref-key="keywords.lion.range"></xf:page-index>
</fo:block>
```

In normal usage conditions, you would probabily generate the index in a XSL transformation.

For example, considering the following XML document:

```
<doc title="African Lion">
      <section title="Classification &amp; Range">
            <keyword>Lions</keyword>, along with the other big cats such as
            <keyword>tigers</keyword>
            ...
      </section>
</doc>
```

To automatically generate a list of keywords and their respective page index, you can use the following XSL template:

```
<xsl:stylesheet version="1.0"
   xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
   xmlns:fo="http://www.w3.org/1999/XSL/Format"
   xmlns:xf="http://www.ecrion.com/xf/1.0"
   xmlns:ms="urn:schemas-microsoft-com:xslt"
   xmlns:tt="samples-and-documentation"
   >
   <ms:script implements-prefix="tt" language="JScript">
   function toLower(str)
   {
       return str.toLowerCase();
   }
   </ms:script>
   <xsl:key name="kkey" match="//keyword" use="tt:toLower(string(text()))"/>
   <xsl:template name="GenerateIndex">
      <xsl:for-each select="//keyword">
         <xsl:sort select="." />
           <xsl:if test="generate-id(.)=
                    generate-id(key('kkey',tt:toLower(string(.)))[1])">
             <fo:block>
               <fo:inline text-transform="lowercase">
                          <xsl:value-of select="."/></fo:inline>
                  <xf:page-index>
                    <xsl:attribute name="ref-key">
                      <xsl:value-of select="tt:toLower(string(.))"/>
                        </xsl:attribute>
                  </xf:page-index>
             </fo:block>
           </xsl:if>
       </xsl:for-each>
   </xsl:template>
</xsl:stylesheet>
```

1.16.2 Encryption

PDF documents can be encrypted, and permission sets can be applied at rendering time using **xf:security** XSL-FO extension.

```
<xf:security xmlns:xf="http://www.ecrion.com/xf/1.0"/
      owner-password="password" user-password="password"
      encryption-strength="128 | 40"
      allow-printing="true | false"
      allow-modify-contents ="true | false"
      allow-copy="true | false"
      allow-modify-annotations="true | false"
      allow-fill-in="true | false"
      allow-screen-readers="true | false"
      allow-assembly="true | false"
      allow-degraded-printing="true | false" >
```

owner-password and **user-password**

There are two passwords that can be specified for a document: an owner password and a user password.

- Opening the document with the correct owner password (assuming it is not the same as the user password) allows full (owner) access to the document. This unlimited access includes the ability to change the document’s passwords and access permissions.
- Opening the document with the correct user password (or opening a document that does not have a user password) allows additional operations to be performed according to the user access permissions specified in the document’s encryption dictionary.

If **user-password** and **owner-passwords** are not specified, the document will be encrypted, with user-level access, and a random owner-password is generated; the user will not be prompted for a password, but nobody can have full access to the document.
If only user-password is specified, a random owner-password is generated, the document will be encrypted and the user will be prompted for a password. Again, nobody can have full access to the document.
If both passwords are specified, the document will be encrypted and the user will be prompted for password; depending on the password entered, the user can have full access, or only restricted access to the document.

encryption-strength
Specifies the encryption strength: 128 or 40 bits. 128 bits is the default.

allow-printing
Print the document (possibly not at the highest quality level, depending on whether allow-degraded-printing is also set).

allow-modify-contents
Modify the contents of the document by operations other than those controlled by allow-modify-annotations, allow-fill-in and allow-assembly.

allow-copy
Copy or otherwise extract text and graphics from the document by operations other than that controlled by allow-screen-readers.

allow-modify-annotations
Add or modify text annotations, fill in interactive form fields, and, if allow-modify-contents is also set, create or modify interactive form fields (including signature fields).

allow-fill-in
Fill in existing interactive form fields (including signature fields), even if allow-modify-annotations is not set.

allow-screen-readers
Extract text and graphics (in support of accessibility to disabled users or for other purposes).

allow-assembly
Assemble the document (insert, rotate, or delete pages and create bookmarks or thumbnail images), even if allow-modify-contents is not set.

allow-degraded-printing
When this is set (and allow-printing is set also), printing is limited to a low level representation of the appearance, possibly of degraded quality.

1.16.3 Metadata

There are two ways to include **metadata** in the PDF files generated by **XF Rendering Server**:

1. Using **xf:info** for generic document information including author, name, subject and keywords data:

```
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
      <xf:info xmlns:xf="http://www.ecrion.com/xf/1.0">
            <xf:title>About Metadata</xf:title>
            <xf:author>Joe Doe</xf:author>
            <xf:subject>Example Metata Document</xf:subject>
            <xf:keywords>PDF XML XMP</xf:keywords>
      </xf:info>
      ...
</fo:root>
```

2. Using **xf:meta** for metadata described in RDF/XML (Resource Description Framework) format.

xf:meta can contain one or many **rdf:RDF** nodes. For each node, the engine will create an XMP (Extensible Metadata Platform) packet in the generated PDF file.

```
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
   <xf:meta xmlns:xf="http://www.ecrion.com/xf/1.0">
      <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
          <rdf:Description about="" xmlns:my="http://www.mydomain.org/myschema/">
            <my:field>value</my:field>
            <my:my-collection>
                  <rdf:Bag>
                        <rdf:li>red</rdf:li>
                        <rdf:li>green</rdf:li>
                        <rdf:li>blue</rdf:li>
                  </rdf:Bag>
            </my:my-collection>
          </rdf:Description>
      </rdf:RDF>
   </xf:meta>
      ...
</fo:root>
```

In this example, we have defined a object containing a member (called "**field**") and a collection containing three items.

Adobe PDF promotes **XMP** as a common standard that every application that works with PDF embedded metadata must understand. XMP supports a subset of RDF/XML. In addition XMP standardizes the definition, creation and processing of metadata by providing:

- A storage model. The format in which data is serialized in PDF (and in conformance with the XMP standard) is handled by **XF Rendering Server**. This includes generating the envelope of your XML data, as well as encoding it properly.
- A set of predefined schemas. Adobe schemas provide property definitions that are relevant for a wide range of applications (including Adobe's editing and publishing products). One of the most interesting features is customization of FileInfo dialog in Adobe applications that support XMP.

1.16.4 Digital Signatures

XF Rendering Server offers support for generating digital signatures in PDF output.

There are two types of signatures:
• invisible signatures - documents are signed and their authenticity can be verified, but they don't have an associated graphic element.
• visible signatures - signatures are associated with a graphic element; usually this element displays a scanned hand signature or a office stamp.

A digitally signed document can not be altered without invalidating the signature.

To generate an invisible signature:

```
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format"
             xmlns:xf="http://www.ecrion.com/xf/1.0">
      <xf:signature name="Dr. Joe Doe"
      location="Rockville, Maryland"
      reason="Prescription"
      certificate-serial-number="58 e9 4c 55 00 00 00 00 00 0c"
      certificate-issuer="Thawte CA"/>
      ...
</fo:root>
```

To generate a visible signature:

```
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format"
                   xmlns:xf="http://www.ecrion.com/xf/1.0">
      <xf:signature id="Sig1" name="Dr. Joe Doe"
      location="Rockville, Maryland"
      reason="Prescription"
      certificate-serial-number="58 e9 4c 55 00 00 00 00 00 0c"
      certificate-issuer="Thawte CA"/>
      ...
      <fo:block ref-signature="Sig1">Signature</fo:block>
</fo:root>
```

Notes:
• "**reason**" attribute is optional.
• "**certificate-serial-number**" and "**certificate-issuer**" must identify a certificate installed in XF Management Console (see below).
• You can use any element to reference a signature, for example a **fo:inline-object** or a **fo:external-graphic.**

To generate a self signed certificate:

• Open **XF Management Console**
• Right click "**Server Certificates**" and click "**Install Certificate**"
• Click **Self-signed certificate** and then **Next**
• Fill in all fields and click **Finish**
• The certificate should be displayed in the list of certificates.
• On Windows XP you can also click **Properties** to view the certificate

To generate a certificate request to be submitted to a Certification Authority (Thawte, Verisign):

- Open **XF Management Console**
- Right click "**Server Certificates**" and click "**Install Certificate**"
- Click "**Certificate issued by a CA**" and then click **Next**
- Click "**Prepare certificate request**"; click **Next**
- Fill in all fields; click **Next**
- Enter a file name and click **Finish**

To install a certificate issued by a CA:

- Open **XF Management Console**
- Right click "**Server Certificates**" and click "**Install Certificate**"
- Click "**Certificate issued by a CA**"; click **Next**
- Click "**Process pending request**"; click **Next**
- Enter a file name; click **Finish**
- The certificate should be displayed in the list of certificates.
- On Windows XP you can also click **Properties** to view the certificate

We recommend that you test using self signed certificates. If you also have a Windows Server computer you can setup your own CA and issue certificates to be used by XF.

You will need Acrobat 6.0 and higher to validate the signatures.
You will need Acrobat 7.0 or higher to validate signatures issued by a CA because Acrobat 6.0 will display an error message when a certificate chain is embedded in the signature.

1.16.5 Barcodes

XF Rendering Server offers support for drawing **128**, **128 Raw**, **128 UCC**, **UPC-A**, **UPC-E**, **EAN-13**, **EAN-8**, **2 of 5**, **3 of 9**, **Postnet**, **Planet** and **DataMatrix** barcodes.
A barcode can be defined as following:

```
<xf:barcode xmlns:xf="http://www.ecrion.com/xf/1.0"/value="upc code"
             type="AUTO|128|128RAW|128UCC|UPC-A|UPC-E|EAN-13|EAN-8|2OF5|3OF9|
                                              DATAMATRIX|POSTNET|PLANET"
      bar-unit="length"
      include-checksum="boolean"
      draw-text="boolean"
      extended="boolean"
      encoding="ASCII|C40|TEXT|BASE256|NONE|AUTO"
      preferred-format="AUTO|C10x10|C12x12|etc."
      fo:content-width="length"
      fo:content-height="length"
      fo:content-scaling="non-uniform|uniform"
      fo:font="font"
      fo:padding="padding"
      fo:border="border"
      fo:color="color">
```

value
- The code value. For EAN and UPC values specified, the value can be specified with or without the check digit (the last digit). If the check digit is specified, then type attribute must be specified as well.
- When type is EAN, UPC, 2OF5 or Postnet, non-digit characters in the value are ignored.
- When type is 3OF9 non extended, only uppercase A-Z and -. $/+%* are acceptable.

type
- 128
- 128 Raw
- 128 UCC
- UPC-A
- UPC-E
- EAN-13
- EAN-8
- 2of5
- 3of9
- Postnet
- Planet
- DATAMATRIX

bar-unit
- Specifies the length of one unit of the barcode and applies only to UPC and EAN barcodes. Barcode lines can be between 1 and 4 such units.

include-checksum
- Specifies it a checksum must be computer for 3 of 9 codes. By default, this value is set to true

draw-text
- Specifies if the barcode value and the checksum must be displayed for 2of5 or 3of9 barcodes. Default is true.

extended
- Allows for a wider character set for 3 of 9 barcodes. Default value is false.

encoding
- ASCII
- C40

- TEXT
- BASE256
- NONE
- AUTO

Encoding for DATAMATRIX codes.

preferred-format

- AUTO
- C10X10
- C12X12
- C14X14
- C16X16
- C18X18
- C20X20
- C22X22
- C24X24
- C26X26
- C32X32
- C36X36
- C40X40
- C44X44
- C48X48
- C52X52
- C64X64
- C72X72
- C80X80
- C88X88
- C96X96
- C104X104
- C120X120
- C132X132
- C144X144
- C8X18
- C8X32
- C12X26
- C12X36
- C16X36
- C16X48

Preferred format for DATAMATRIX codes.
Normal formatting attributes (font, color) still apply, but they must be prefixed by **fo:namespace**.

Examples:

1. **UPC-A** barcode:

```
<fo:block>
      <xf:barcode value="075-67 816 4125" type="UPC-A" bar-unit="1px"
      fo:font="7.5pt Arial" fo:color="blue"
      fo:scaling="non-uniform" fo:content-height="50pt"
      fo:padding="10pt"/>
</fo:block>
```

The rendering result is displayed in the next figure:

0 75678 16412 5

Figure 26.

2. **DATAMATRIX** barcode:

```
<fo:block>
      <xf:barcode value="Hello World" type="DATAMATRIX" encoding="ASCII"
                                          preferred-format="C20X20"/>
<fo:block/>
```

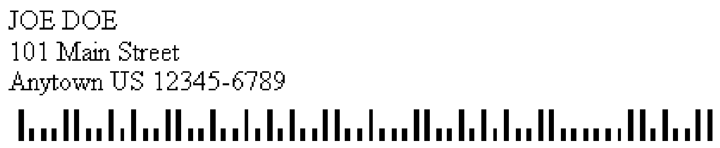
The rendering result is displayed in the next figure:

Figure 27.

3. **Postnet** barcode:

```
<fo:block>
      JOE DOE<fo:block/>
      101 Main Street<fo:block/>
      Anytown US 12345-6789<fo:block/>
      <xf:barcode xmlns:xf="http://www.ecrion.com/xf/1.0" value="12345678901"
                                                   type="Postnet"/>
</fo:block>
```

The rendering result is displayed in the next figure:



JOE DOE
101 Main Street
Anytown US 12345-6789

Figure 28.

1.16.6 Extended Layout Elements

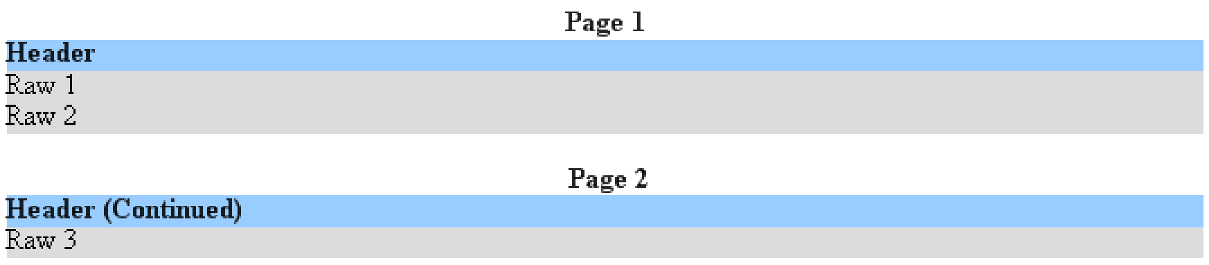
This section describes several extensions to standard XSL-FO elements like **fo:table** and **fo:list-block**. The purpose of these extensions is to provide additional control over the generated output.

Continued Labels

Sometimes is necessary to display a "**Continued from the previous page**" text whenever a page break occurs.
This behaviour can be achieved using **xf:continued-label** element:

```
...
<fo:table>
   <fo:table-column column-width="proportional-column-width(1)"/>
   <fo:table-header>
       <fo:table-row>
             <fo:table-cell background-color="rgb(153,204,255)"
                                              font-weight="bold">
                   <fo:block>Header
                   <xf:continued-label xmlns:xf="http://www.ecrion.com/xf/1.0">
                         (Continued)
                   </xf:continued-label>
                   </fo:block>
             </fo:table-cell>
       </fo:table-row>
   </fo:table-header>
   <fo:table-body>
       <fo:table-row>
             <fo:table-cell>
                   <fo:block>
                         Row 1
                   </fo:block>
             </fo:table-cell>
       </fo:table-row>
       <fo:table-row>
             <fo:table-cell>
                   <fo:block>
                         Row 2
                   </fo:block>
             </fo:table-cell>
       </fo:table-row>
       <fo:table-row>
             <fo:table-cell>
                   <fo:block>
                         Row 3
                   </fo:block>
             </fo:table-cell>
       </fo:table-row>
   </fo:table-body>
</fo:table>
...
```

The rendering result is displayed in the next figure:


Page 1

Header
Raw 1
Raw 2

Page 2

Header (Continued)
Raw 3

Figure 29.

Repeated Table Cells

Sometimes is needed to repeat a table cell whenever a page break occurs.

You can achieve this behaviour by setting the value of **xf:repeat-on-page-break** for the table cell to be repeated to true:

```
...
<fo:table>
   <fo:table-column column-width="proportional-column-width(1)" column-number="1"/>
   <fo:table-column column-width="proportional-column-width(1)" column-number="2"/>
   <fo:table-body>
       <fo:table-row>
              <fo:table-cell xmlns:xf="http://www.ecrion.com/xf/1.0"
              display-align="center" xf:repeat-on-page-break="true"
              border-style="solid" border-width="1pt" border-color="rgb(0,0,0)"
              padding="2pt" background-color="rgb(153,204,255)">
                     <fo:block font-weight="bold">
                            [table cell's content is repeated at every split]
                     </fo:block>
              </fo:table-cell>
              <fo:table-cell border-style="solid" border-width="1pt"
               border-color="rgb(0,0,0)" border-width="1pt" padding="2pt"
              background-color="rgb(255,255,255)">
                     <fo:block>
                            Normal cell spanning across multiple pages.
                            ..........................................
                            Normal cell spanning across multiple pages.
                     </fo:block>
              </fo:table-cell>
       </fo:table-row>
   </fo:table-body>
</fo:table>
...
```

As you can see in the code fragment below, the xf namespace must be declared (if not already declared on one of the parent elements).

The rendering result is displayed in the next figure:

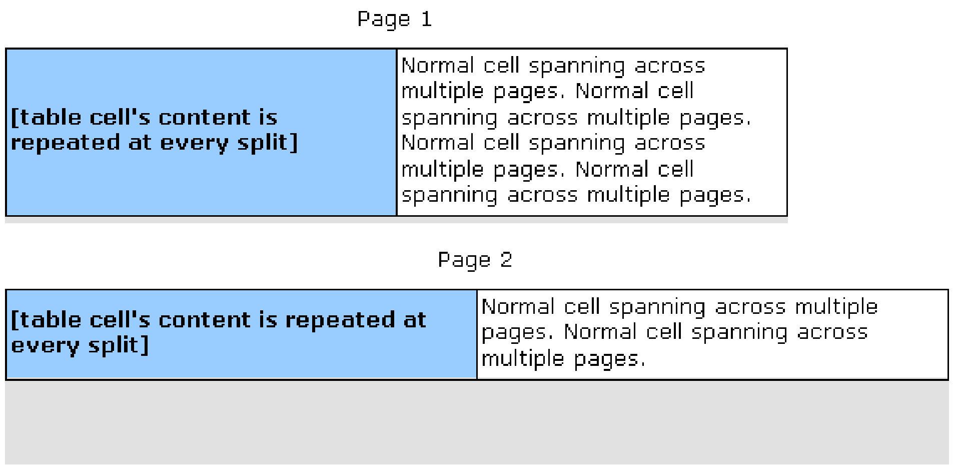

Page 1
[table cell's content is repeated at every split]
Normal cell spanning across multiple pages. Normal cell spanning across multiple pages. Normal cell spanning across multiple pages. Normal cell spanning across multiple pages.
Page 2
[table cell's content is repeated at every split]
Normal cell spanning across multiple pages. Normal cell spanning across multiple pages.

Figure 30.

This example uses two different page layouts to show that the layout of the repeated content is calculated individually every time a page break occurs.

Alternative Layout Flow

XF Rendering Server provides several attributes for **fo:list-block** elements that help users achieve an alternative layout flow. The layout is achieved by changing attributes that control the initial label placement, the sequence of items and reset points.
A common usage of this feature is the generation of catalogs with product items that swap images and textual descriptions.

Syntax:

```
<fo:list-block xf:initial-label-placement="inside | outside | left | right"
      xf:label-placement-sequence="normal | alternate"
      xf:reset-label-placement="none | column | page">
...
</fo:list-block>
```

xf:initial-label-placement
Specifies the position of the first label; this position can be absolute (left or right) or relative to the binding margin (inside or outside; 'inside' will mean right for even pages and left for odd pages).

xf:label-placement-sequence
Defines how successive items can have opposite positions or not. The values defined so far are 'normal'(all items have the same position) or 'alternate' (even items have opposite positions than odd items). In the future something more generic may be added, like (sequence(normal,opposite,opposite,normal)meaning two opposite labels are inserted between two normal labels).

xf:reset-label-placement
Defines the reset points, that is, when the sequence must be repeated (none, when a new column starts, or when a new page starts).

```
<fo:list-block xf:initial-label-placement="outside"
      xf:label-placement-sequence="alternate"
      xf:reset-label-placement="page">
   <fo:list-item keep-together="always">
      <fo:list-item-label width="1in">
            <fo:block background="rgb(135, 206, 250)" border="1px solid black">
                  Label
            </fo:block>
      </fo:list-item-label>
      <fo:list-item-body start-indent="1.1in">
            <fo:block background="white" border="1px solid black" >
                  content content content content content content content content
                  content content content content content content content content
                  content content content content content content content content
                  content content content content content content content content
                  content content content content content content content content
            </fo:block>
      </fo:list-item-body>
   </fo:list-item>
   <fo:list-item keep-together="always">
      <fo:list-item-label width="1in">
            <fo:block background="rgb(135, 206, 250)" border="1px solid black">
                  Label
            </fo:block>
      </fo:list-item-label>
      <fo:list-item-body start-indent="1.1in">
            <fo:block background="white" border="1px solid black" >
                  content content content content content content content content
                  content content content content content content content content
                  content content content content content content content content
                  content content content content content content content content
                  content content content content content content content content
            </fo:block>
      </fo:list-item-body>
   </fo:list-item>
   <fo:list-item keep-together="always">
      <fo:list-item-label width="1in">
            <fo:block background="rgb(135, 206, 250)"  border="1px solid black">
                  Label
            </fo:block>
      </fo:list-item-label>
      <fo:list-item-body start-indent="1.1in">
            <fo:block background="white" border="1px solid black" >
                  content content content content content content content content
                  content content content content content content content content
            </fo:block>
      </fo:list-item-body>
   </fo:list-item>
   <fo:list-item keep-together="always">
      <fo:list-item-label width="1in">
            <fo:block background="rgb(135, 206, 250)" border="1px solid black">
                  Label
            </fo:block>
      </fo:list-item-label>
      <fo:list-item-body start-indent="1.1in">
            <fo:block background="white" border="1px solid black">
                  content content content content content content content content
                  content content content content content content content content
            </fo:block>
      </fo:list-item-body>
   </fo:list-item>
   <fo:list-item keep-together="always">
      <fo:list-item-label width="1in">
            <fo:block background="rgb(135, 206, 250)" border="1px solid black">
                  Label
            </fo:block>
      </fo:list-item-label>
      <fo:list-item-body start-indent="1.1in">
            <fo:block background="white" border="1px solid black" >
                  content content content content content content content content
                  content content content content content content content content
            </fo:block>
      </fo:list-item-body>
   </fo:list-item>
</fo:list-block>
```

The rendering result is displayed in the figure:

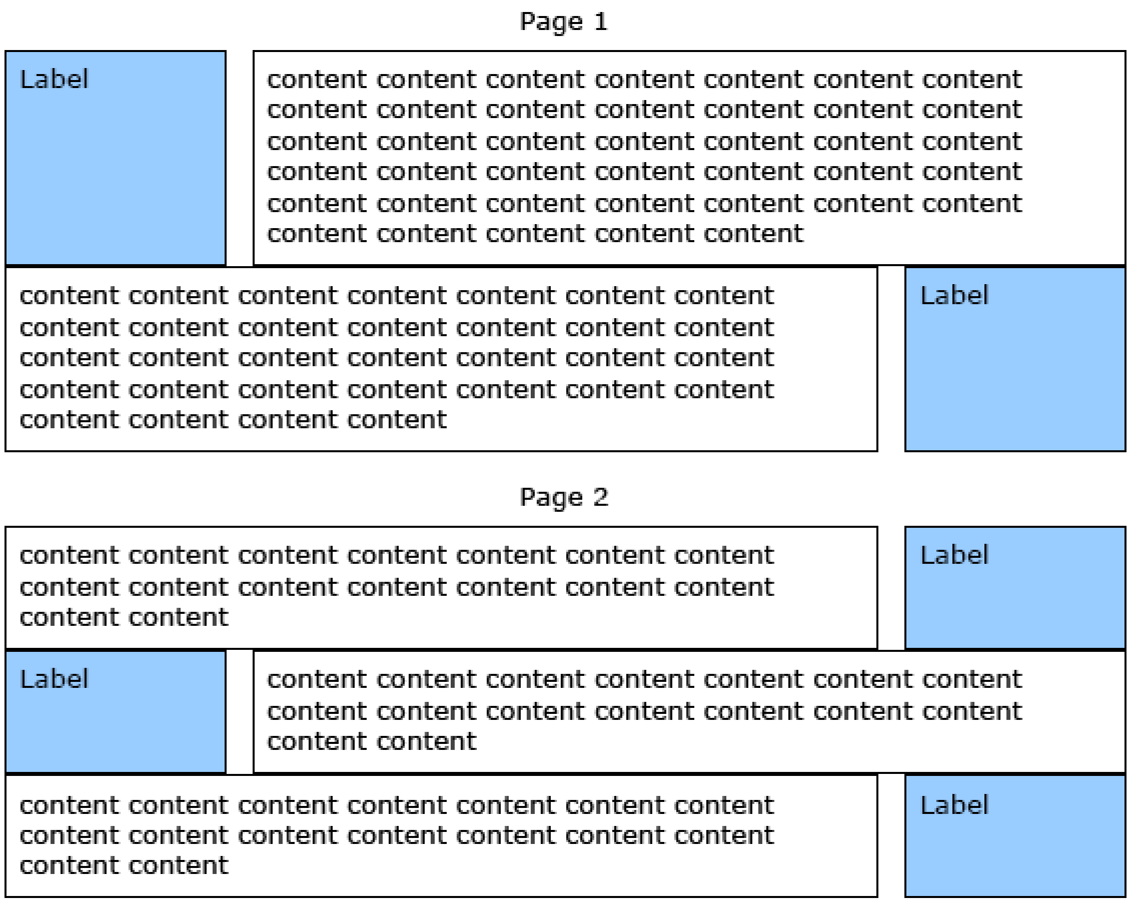

Page 1
Label
content content content content content content content
content content content content content content content
content content content content content content content
content content content content content content content
content content content content content content content
content content content content content
content content content content content content content
content content content content content content content
content content content content content content content
content content content content content content content
content content content content
Label
Page 2
content content content content content content content
content content content content content content content
content content
Label
Label
content content content content content content content
content content content content content content content
content content
content content content content content content content
content content content content content content content
content content
Label

Figure 31.

1.16.7 AFP Extensions

This section describes an extension to standard XSL-FO elements and that is **tag-afp-page** contained only by to **fo:root** and used when you work with an .afp file as output**.**

The following example shows you how it can be used:

```
<?xml version="1.0" encoding="utf-8" standalone="no"?>
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
   <xf:tag-afp-page xmlns:xf="http://www.ecrion.com/xf/1.0"
             ref-id="PAGE001" key="key-one" value="kent"/> (1)
   <xf:tag-afp-page xmlns:xf="http://www.ecrion.com/xf/1.0"
             ref-id="PAGE002" key="key-two" value="kent2"/> (2)
   <fo:layout-master-set>
       <fo:simple-page-master master-name="all-pages"
                           page-height="2in" page-width="6in">
             <fo:region-body region-name="xsl-region-body"
                    column-gap="0.25in" padding="6pt" margin="0.7in" />
             <fo:region-before region-name="xsl-region-before"
                    display-align="after" extent="0.7in" padding="6pt"/>
             <fo:region-after region-name="xsl-region-after"
                    display-align="before" extent="0.7in" padding="6pt"/>
       </fo:simple-page-master>
       <fo:page-sequence-master master-name="default-sequence">
             <fo:repeatable-page-master-reference master-reference="all-pages"/>
       </fo:page-sequence-master>
   </fo:layout-master-set>
   <fo:page-sequence master-reference="default-sequence">
       <fo:flow flow-name="xsl-region-body">
             <fo:block id="PAGE001" color="rgb(0,51,102)"
             font-family="Times New Roman" font-size="14"> (3)
                    Page 1 content<fo:block/>
             </fo:block>
             <fo:block id="PAGE002" color="rgb(0,51,102)"
             font-family="Times New Roman" font-size="14" break-before="page" >
                    Page 2 content</fo:block>
       </fo:flow>
   </fo:page-sequence>
</fo:root>
```

The important points in the example above are:

(1) In order to work with extensions you must have the corresponding xml namespace defined.
(2) xf:tag-afp-page uses three attributes: **ref-id**, **key** and **value**; ref-id is used for editing in XF Designer while the other two attributes are needed when you work with the .afp output file.
(3) If a **fo:block** has an associated reference id (**ref-id**) that means is tagged with the corresponding key and value.

Another extension used for .afp output files is **xf:afp-nop** contained by **fo:page-sequence**:

```
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format"
                           xmlns:xf="http://www.ecrion.com/xf/1.0">
   <fo:layout-master-set>
   ...
   </fo:layout-master-set>
   <fo:page-sequence master-reference="default-sequence">
       <xf:afp-nop>
```

```
      This text will be seen only by printer and in the .afp file.
      </xf:afp-nop>
      ...
   </fo:page-sequence>
</fo:root>
```

The content of this section is visible only in the .afp file at the beginning of every page and in the printer mode.

1.16.8 Filter/Crop/Transform

This section describes extensions used while working with documents that contains images and you need to make certain modification to the whole document or only to a part of it: **xf:filter**, **xf:crop** and **xf: transform.**

These can be applied only to **fo:instream-foreign-object** or **fo:external-graphic:**

```
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
   <fo:layout-master-set>
       <fo:simple-page-master master-name="all-pages"
                    page-width="6in" page-height="3in">
              <fo:region-body region-name="xsl-region-body"
              column-gap="0.25in" padding="2pt" margin="0.2in"/>
              <fo:region-before region-name="xsl-region-before"
              display-align="after" extent="0.2in" padding="2pt"/>
              <fo:region-after region-name="xsl-region-after"
              display-align="before" extent="0.2in" padding="2pt"/>
       </fo:simple-page-master>
       <fo:page-sequence-master master-name="default-sequence">
              <fo:repeatable-page-master-reference
                             master-reference="all-pages"/>
       </fo:page-sequence-master>
   </fo:layout-master-set>
   <fo:page-sequence master-reference="default-sequence">
       <fo:static-content flow-name="xsl-region-before"
              font-size="12pt" font-family="Times New Roman">
              <fo:block> </fo:block>
       </fo:static-content>
       <fo:static-content flow-name="xsl-region-after"
              font-size="12pt" font-family="Times New Roman">
              <fo:block> </fo:block>
       </fo:static-content>
       <fo:flow flow-name="xsl-region-body" font-size="12pt"
                                  font-family="Times New Roman">
              <fo:block start-indent="0px" text-indent="9.8px">
                     <fo:external-graphic src="Blue hills.jpg"/>
                     <fo:external-graphic xmlns:xf="http://www.ecrion.com/xf/1.0" (1)
                     xf:crop="rect(0pt,100pt,0pt,0pt)" (2) src="Blue hills.jpg"/>
                     <fo:external-graphic xmlns:xf="http://www.ecrion.com/xf/1.0"
                     xf:crop="rect(0pt,100pt,0pt,0pt)" xf:transform="flip-y" (3)
                     src="Blue hills.jpg"/>
              </fo:block>
       </fo:flow>
   </fo:page-sequence>
</fo:root>
```

The rendering result can be seen below:

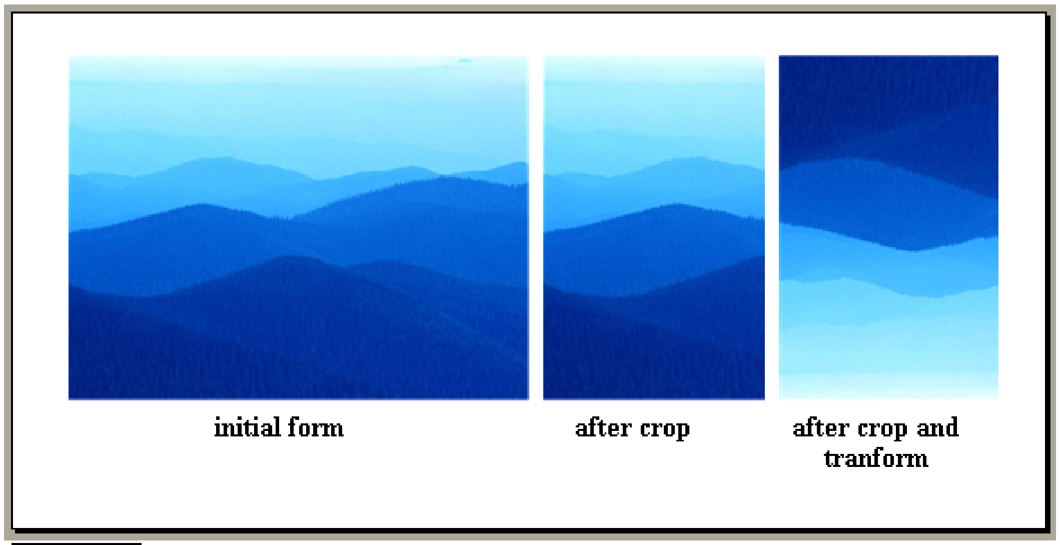


Figure 32.

Things to notice in this example:

(1) In order to work with extensions you must have the corresponding xml namespace defined.
(2) **xf:crop** is used to crop an image and its parameters represent top-length, right-length, bottom-length and left-length. Its value can be set also on **"auto"** that means it returns to the initial form.
(3) **xf:transform** takes one of the following values: **"flip-x" | "flip-y" | "flip-xy"** (the image is 180 degrees rotated on the horizontal axis | vertical axis | both axes).

xf:filter has 2 possible values: **"ConvertToGrayscale()", "IgnoreClips()" or both**. You can associate both values if you put ";" between them:

```
<fo:block>
      <fo:external-graphic scaling="non-uniform"
      xf:filter="IgnoreClips();ConvertToGrayscale()"
      src="Blue hills.jpg"/>
</fo:block>
```

1.16.9 Insert Document

xf:insert-document is an extension used to insert an external document between two **fo:page-sequence** elements; it can be contained only by **fo:root**.

This example inserts CoverPage.pdf at the beginning of the document:

```
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format"
             xmlns:xf="http://www.ecrion.com/xf/1.0"> (1)
      <fo:layout-master-set>
            <fo:simple-page-master master-name="all-pages"
                         page-width="8.5in" page-height="11in">
                  <fo:region-body region-name="xsl-region-body"
                  column-gap="0.25in" padding="6pt"  padding-bottom="6pt"
                                                    margin="0.7in"/>
                  <fo:region-before region-name="xsl-region-before"
                  display-align="after" extent="0.7in" padding="6pt"/>
                  <fo:region-after region-name="xsl-region-after"
                  display-align="before" extent="0.7in" padding="6pt"/>
            </fo:simple-page-master>
            <fo:page-sequence-master master-name="default-sequence">
                  <fo:repeatable-page-master-reference
                                    master-reference="all-pages"/>
            </fo:page-sequence-master>
      </fo:layout-master-set>
      <xf:insert-document  src="url(CoverPage.pdf)" page-width="8.5in"
      page-height="11in" scaling="non-uniform" pageOrientation="-90"/> (2)
      <fo:page-sequence master-reference="default-sequence">
            ...
            <fo:flow flow-name="xsl-region-body" font-size="12pt"
                                    font-family="Times New Roman">
                  <fo:block>...</fo:block>
            </fo:flow>
      </fo:page-sequence>
</fo:root>
```

The important points in this example above are:

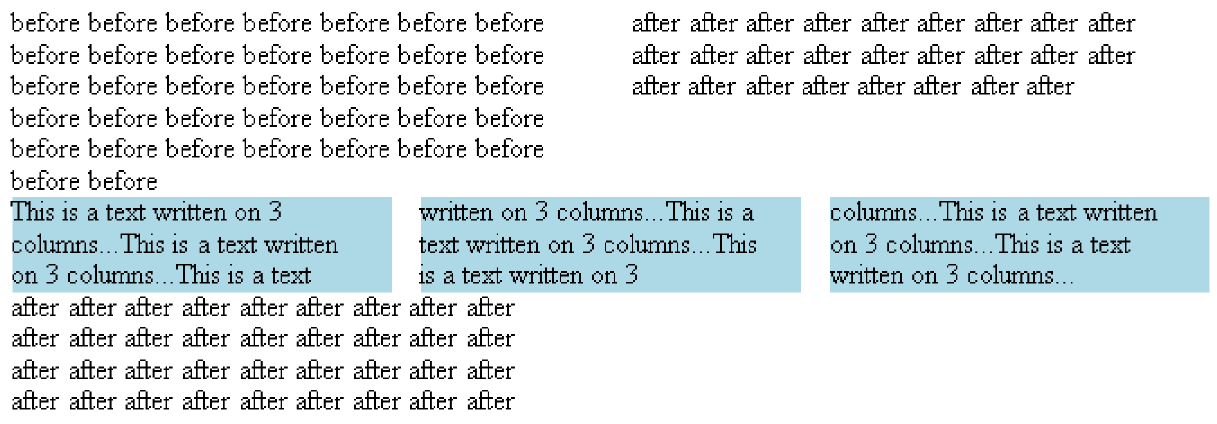
(1) In order to work with extension you must have the corresponding xml namespace defined.
(2) You can set its different attributes to the value you need: **page-height**, **page-width**, **pageOrientation** , **scaling**, **pages**, **contentType** and source from where you insert the document.

1.16.10 Multicolumn Block

This section describes an extension used to split certain blocks into columns: **xf:multicolumn-block**. It can be contained only by **fo:flow (2)** and it can be used as you can see in the following example:

```
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format"
             xmlns:xf="http://www.ecrion.com/xf/1.0"> (1)
   <fo:layout-master-set>
       <fo:simple-page-master master-name="all-pages"
                    page-width="8.5in" page-height="11in">
             <fo:region-body region-name="xsl-region-body"
             margin="0.7in" column-gap="0.25in" padding="6pt" />
             <fo:region-before region-name="xsl-region-before"
             extent="0.7in" display-align="after" padding="6pt 0.7in" />
             <fo:region-after region-name="xsl-region-after"
             extent="0.7in" display-align="before" padding="6pt 0.7in" />
       </fo:simple-page-master>
       <fo:page-sequence-master master-name="default-sequence">
             <fo:repeatable-page-master-reference
                                 master-reference="all-pages" />
       </fo:page-sequence-master>
   </fo:layout-master-set>
   <fo:page-sequence master-reference="default-sequence">
       ...
       <fo:flow flow-name="xsl-region-body"
       font-family="Times New Roman" font-size="12pt">
             <xf:multicolumn-block column-count="3"
                                 column-gap="0.5in"> (3)
                    <fo:block>
                    This is a text written on 3 columns...
                    This is a text written on 3 columns...
                    This is a text written on 3 columns...
                    This is a text written on 3 columns...
                    ....
                    </fo:block>
             </xf:multicolumn-block>
       </fo:flow>
   </fo:page-sequence>
</fo:root>
```

The rendering result is displayed in the next picture:



before before before before before before before
before before before before before before before
before before before before before before before
before before before before before before before
before before before before before before before
before before

after after after after after after after after after
after after after after after after after after after
after after after after after after after after

This is a text written on 3
columns...This is a text written
on 3 columns...This is a text

written on 3 columns...This is a
text written on 3 columns...This
is a text written on 3

columns...This is a text written
on 3 columns...This is a text
written on 3 columns...

after after after after after after after after after
after after after after after after after after after
after after after after after after after after after
after after after after after after after after after

Figure 32.

The important points in the example above are:

(1) In order to work with extension you must have the corresponding xml namespace defined.
(3) You must fix the number of columns you split the block into: **column-count** and the distance between the columns: **column-gap**.

1.16.11 Viewer Preferences

Another extension that applies on fo:root is **xf:viewer-preferences** used to set different properties regarding page layout, page mode and the visibility of bars in the exported document (toolbar, menubar, etc.).

The following example displays the way it can be used:

```
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
   <xf:viewer-preferences xmlns:xf="http://www.ecrion.com/xf/1.0" (1)
             hide-toolbar="true" hide-menubar="true" fit-window="false"
             page-layout="TwoColumnLeft" page-mode="UseThumbs"/> (2)
   <fo:layout-master-set>
       <fo:simple-page-master master-name="all-pages" page-width="8.5in"
                                              page-height="11in">
             <fo:region-body region-name="xsl-region-body"
                    margin="0.7in" column-gap="0.25in" padding="6pt" />
             <fo:region-before region-name="xsl-region-before"
             extent="0.7in" display-align="after" padding="6pt 0.7in" />
             <fo:region-after region-name="xsl-region-after" extent="0.7in"
                     display-align="before" padding="6pt 0.7in" />
       </fo:simple-page-master>
       <fo:page-sequence-master master-name="default-sequence">
             <fo:repeatable-page-master-reference master-reference="all-pages" />
       </fo:page-sequence-master>
   </fo:layout-master-set>
   <fo:page-sequence master-reference="default-sequence">
   ...
   </fo:page-sequence>
</fo:root>
```

Things to notice:

(1) In order to work with extension you must have the corresponding xml namespace defined.
(2) You can set your preferences by using the available attributes with the corresponding value:

hide-toolbar	**"true"/"false"**
hide-menubar	**"true"/"false"**
hide-window-ui	**"true"/"false"**
fit-window	**"true"/"false"**
center-window	**"true"/"false"**
display-title	**"true"/"false"**
page-layout	**"NULL"/"SinglePage"/"OneColumn"/"TwoColumnLeft"/"TwoColumnRight"/ "TwoPageLeft"/"TwoPageRight"**
page-mode	**"NULL"/"UseNone"/"Useoutlines"/"UseThumbs"/"FullScreen"/ "UseOptionalContentGroup"/"UseAtachments"**

1.16.12 Tab Extensions

This section describes extensions used to set the tab value needed while working with a document.
In the example below it can be seen the way it can be used:

```
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format"
      xmlns:xf="http://www.ecrion.com/xf/1.0" > (1)
<xf:defaultTabStop val="2in"/> (2)
   <fo:layout-master-set>
      <fo:simple-page-master master-name="all-pages"
             page-width="8.5in" page-height="3in">
             <fo:region-body region-name="xsl-region-body"
             margin="0.7in" column-gap="0.25in" padding="6pt"/>
             <fo:region-before region-name="xsl-region-before"
      extent="0.7in" display-align="after" padding="6pt 0.7in" />
             <fo:region-after region-name="xsl-region-after" extent="0.7in"
             display-align="before" padding="6pt 0.7in" />
      </fo:simple-page-master>
      <fo:page-sequence-master master-name="default-sequence">
             <fo:repeatable-page-master-reference
                            master-reference="all-pages" />
      </fo:page-sequence-master>
   </fo:layout-master-set>
   <fo:page-sequence master-reference="default-sequence">
      <fo:static-content flow-name="xsl-region-before"
                    font-family="Times New Roman" font-size="12pt">
             <fo:block>&#x00A0;</fo:block>
      </fo:static-content>
      <fo:static-content flow-name="xsl-region-after"
                    font-family="Times New Roman" font-size="12pt">
             <fo:block>&#x00A0;</fo:block>
      </fo:static-content>
      <fo:flow flow-name="xsl-region-body"
                    font-family="Times New Roman" font-size="12pt">
             <fo:block font-size="14" color="red">
             <xf:tabstops> (3)
                    <xf:tabstop pos="1in"></xf:tabstop> (4)
                    <xf:tabstop pos="3in"></xf:tabstop>
                    <xf:tabstop pos="4in"></xf:tabstop>
             </xf:tabstops>
             t<xf:tab/>o<xf:tab/>t<xf:tab/>o</fo:block> (5)
             <fo:block text-decoration="underline" font-size="14"
                                              color="blue">
             <xf:tabstops>
                    <xf:tabstop pos="96px"></xf:tabstop>
                    <xf:tabstop pos="2in"></xf:tabstop>
                    <xf:tabstop pos="5in"></xf:tabstop>
             </xf:tabstops>
             t<xf:tab/>o<xf:tab/>t<xf:tab/>o</fo:block>
             <fo:block font-size="14" color="green">
             t<xf:tab/>o<xf:tab/>t<xf:tab/>o</fo:block>
      </fo:flow>
   </fo:page-sequence>
</fo:root>
```

The rendering result is displayed in the next picture:

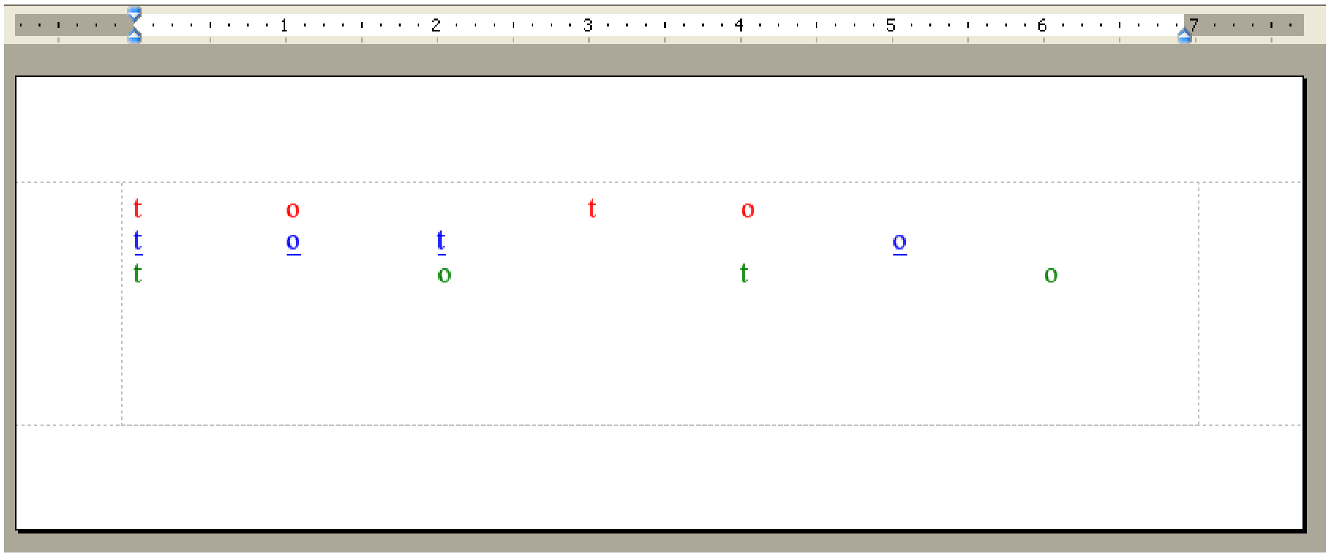


Figure 32.

Things to notice in this example:

(1) In order to work with extension you must have the corresponding xml namespace defined.
(2) **xf:defaultTabStop** defines working with tabs; **val**="2in" set the default tab value.
(3) **xf:tabstops** cancels the default tab value for the fo:block where is defined.
(4) **xf:tabstop** is used to set the new tab value using **pos** attribute; its value can be introduced in pixels or inch; that means that every time you use tab the cursor jumps to the defined values for **pos**.
(5) **xf:tab** is the extension used to insert the tab.

1.16.13 Page Settings

XF Rendering Server offers support for different page properties regarding the .pdf output file through **xf:page-settings** extension.
In the example below you can see how it can be used:

```
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format"
      xmlns:xf="http://www.ecrion.com/xf/1.0">
   <xf:page-settings blending-color-space="DeviceRGB"
      pdf-conformance="PDF/A-1b:2005" tagged-pdf="true"
      pdf-embed-true-type-fonts="true"
      pdf-overprint-black-text="false" pdf-version="1.3">
   </xf:page-settings>
   <fo:layout-master-set>
   ...
</fo:root>
```

You may notice that PDF documents that contain transparent RGB images may not print correctly on CMYK printers. This happens because the mathematics of the blending are occurring the color space of the output device. If you are sending in elements in RGB, and the calculation is occurring in CMYK (the default of most printers), you will be getting poor results.
To correct this, you set the **blending-color-space** attribute to one of the available values : **"DeviceNative" | "DeviceRGB" | "DeviceCMYK" | "DeviceGray".**

pdf-overprint-black-text is an option for printing and supports a boolean value.
If the overprint parameter is **"false"** (the default value), painting a color in any color space causes the corresponding areas of unspecified colorants to be erased (painted with a tint value of 0.0). The effect is that the color at any position on the page is whatever was painted there last, which is consistent with the normal painting behavior of the opaque imaging model.
If the overprint parameter is **"true"** and the output device supports overprinting, no such erasing actions are performed; anything previously painted in other colorants
is left undisturbed. Consequently, the color at a given position on the page may be a combined result of several painting operations in different colorants. The effect produced by such overprinting is device-dependent and is not defined by the PDF language.

pdf-embed-true-type-fonts can be set on: **"true" | "false" | "subset"**.
This attribute is used to embed document characters; difference between "true" and "subset" is that set on "subset" will be embedded only once each character.

pdf-conformance set the type of PDf file from: **"PDF/X-1a" | "PDF/A-1a" | "PDF/A-1b".**
The difference between the last two is that "PDF/A-1a" is tagged. That means that the following lines are equivalent:

```
<xf:page-settings pdf-conformance="PDF/A-1b:2005"
      tagged-pdf="true"></xf:page-settings>
<xf:page-settings pdf-conformance="PDF/A-1a:2005">
</xf:page-settings>
```

PDF/A is a standard that defines a format for the long-term archiving of electronic documents and is based on the PDF Reference Version 1.4. while the purpose of PDF/X is to facilitate graphics exchange.

tagged-pdf can have the value **"true" | "false".**
In PDF files, structure is expressed via "**tags**". Tags may be generated automatically for any PDF file using Acrobat 6.0 Professional, but unless the document is very simple indeed, automated tagging alone is unlikely to produce satisfactory results, and is certainly not a quick-fix for compliance with Section 508.
A PDF file equipped with well-formed tags may be "reflowed" to fit different page or screen widths, and will display well on handheld devices.
Tagged PDF files also work better with the screen-reader devices used by many blind and other disabled users.
In most cases, tags are necessary in order to make a PDF file comply with Section 508.

pdf-version can be set to one of the available versions: **"1.3" | "1.4".**

1.16.14 Color Profile

This section describes an extension used to assign color profiles to images and SVG documents : **xf:color-profile**.
It applies to **fo:external-graphic** and **fo:instream-foreign-object** as you can see in the following example:

```
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format"
      xmlns:xf="http://www.ecrion.com/xf/1.0"> (1)
   <fo:declarations>
       <fo:color-profile color-profile-name="RGBColorProfile"
       src="C:\Work\Documentation\Adobe ICC Profiles\RGB Profiles\AppleRGB.icc"/>
       <fo:color-profile color-profile-name="CMYKColorProfile"
       src="C:\Work\Documentation\Adobe ICC Profiles\CMYK Profiles\EuroscaleCoated.icc"/>
   </fo:declarations>
   <fo:layout-master-set>
       <fo:simple-page-master master-name="all-pages"
                   page-width="8in" page-height="11in">
              <fo:region-body region-name="xsl-region-body"
              column-gap="0.25in" padding="6pt" margin="0.7in"/>
              <fo:region-before region-name="xsl-region-before"
              display-align="after" extent="0.7in" padding="6pt"/>
              <fo:region-after region-name="xsl-region-after"
              display-align="before" extent="0.7in" padding="6pt"/>
       </fo:simple-page-master>
       <fo:page-sequence-master master-name="default-sequence">
              <fo:repeatable-page-master-reference
                                  master-reference="all-pages"/>
       </fo:page-sequence-master>
   </fo:layout-master-set>
   <fo:page-sequence master-reference="default-sequence">
       <fo:flow flow-name="xsl-region-body" font-size="12pt"
                                  font-family="Times New Roman">
              <fo:block background="rgb(255,0,0)">RGB Color</fo:block>
              <fo:block background="rgb-icc(255,0,0,#RGBColorProfile,1,0,0)">
                                               ICC RGB Color</fo:block>
              <fo:block background="cmyk(0, 100, 100, 0)">
                                               CMYK Color</fo:block>
              <fo:block background="rgb-icc(255,0,0,#CMYKColorProfile,0,1,1,0)">
                                               ICC CMYK Color</fo:block>
              <fo:block>
                     <fo:external-graphic src="rgb.tif"/>
                            <fo:external-graphic src="rgb.tif"
                            xf:color-profile="#RGBColorProfile"/> (2)
                     <fo:external-graphic src="cmyk.tif"/>
                     <fo:external-graphic src="cmyk.tif"
                            xf:color-profile="#CMYKColorProfile"/>
              </fo:block>
              <fo:block>
                     <fo:instream-foreign-object>
                            <svg width="100" height="100"
                            xmlns="http://www.w3.org/2000/svg">
                                   <rect x="0" y="0" width="100"
                                   height="100" fill="red" />
                            </svg>
                     </fo:instream-foreign-object>
                            <fo:instream-foreign-object
                                   xf:color-profile="#RGBColorProfile"> (3)
                            <svg width="100" height="100"
                                   xmlns="http://www.w3.org/2000/svg">
```

```
                                 <rect x="0" y="0" width="100"
                                        height="100" fill="red" />
                          </svg>
                    </fo:instream-foreign-object>
                    <fo:instream-foreign-object>
                          <svg width="100" height="100"
                          xmlns="http://www.w3.org/2000/svg">
                                 <rect x="0" y="0" width="100"
                                 height="100" fill="cmyk(0,100,100,0)" />
                          </svg>
                    </fo:instream-foreign-object><fo:instream-foreign-object
                                         xf:color-profile="#CMYKColorProfile">
                          <svg width="100" height="100"
                                        xmlns="http://www.w3.org/2000/svg">
                                 <rect x="0" y="0" width="100" height="100"
                                                      fill="cmyk(0,100,100,0)" />
                          </svg>
                    </fo:instream-foreign-object>
             </fo:block>
             <fo:block>
                    <fo:instream-foreign-object
                          xf:color-profile="#CMYKColorProfile">
                          <svg width="100" height="100"
                                        xmlns="http://www.w3.org/2000/svg">
                                 <rect x="0" y="0" width="100"
                                        height="100" fill="red" />
                          </svg>
                    </fo:instream-foreign-object>
                    <fo:instream-foreign-object>
                          <svg width="100" height="100"
                                        xmlns="http://www.w3.org/2000/svg">
                                 <defs>
                                        <color-profile/>
                                 </defs>
                                 <rect x="0" y="0" width="100"
                                 height="100" fill="red" />
                          </svg>
                    </fo:instream-foreign-object>
                    <fo:external-graphic src="redsquare.svg"
                                 xf:color-profile="#RGBColorProfile"/>
                    <fo:external-graphic src="redsquare.svg"
                                 xf:color-profile="#CMYKColorProfile"/>
             </fo:block>
      </fo:flow>
   </fo:page-sequence>
</fo:root>
```

The rendering result is displayed in the next figure:

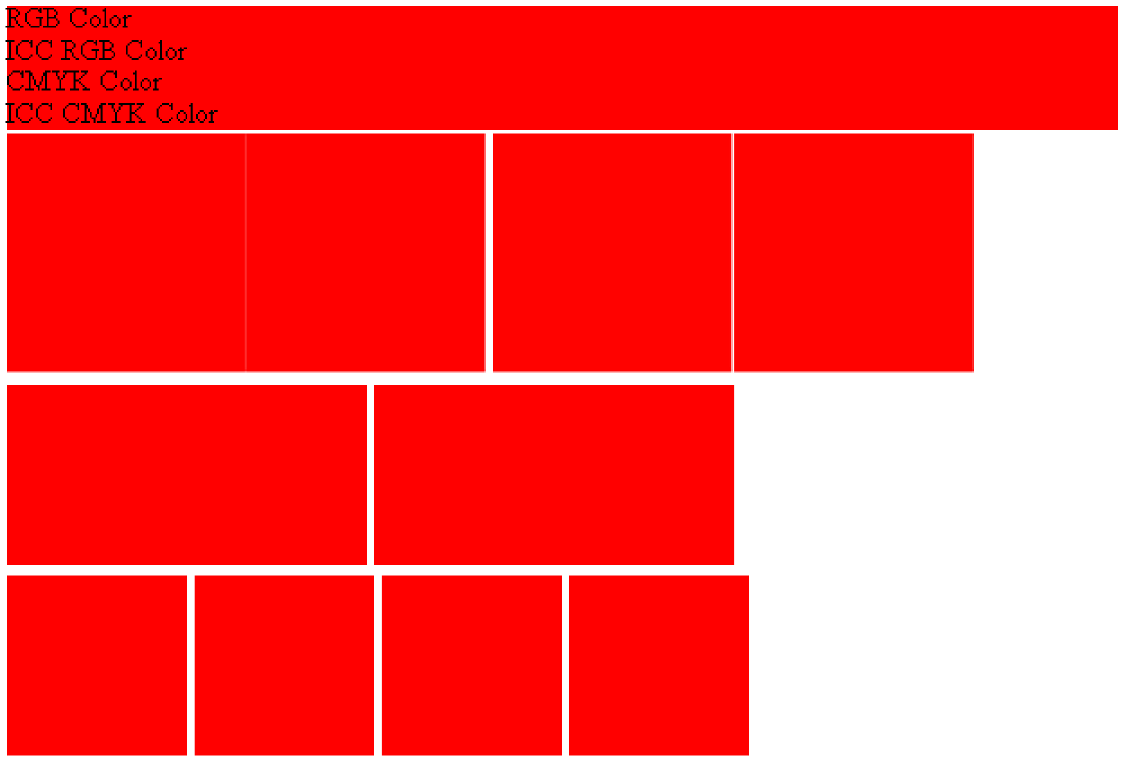


Figure 33.

The important points in this example above are:

(1) In order to work with extension you must have the corresponding xml namespace defined.
(2),(3) It applies to **fo:external-graphic** and **fo:instream-foreign-object.**

See also:
Color Management 24

1.16.15 Postscript Device Setup

XF Rendering Server 2008 offers support for paper tray selection in XSL-FO by using **xf:postscript-device-setup** extension element.
This element will have effect only when rendering Postscript output; PDF or raster formats are not affected.

xf:postscript-device-setup can only contained by **fo:simple-page-master** and it will affect only those pages where the page master is used:

```
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format"
             xmlns:xf="http://www.ecrion.com/xf/1.0"> (1)
    <fo:layout-master-set>
        <fo:simple-page-master master-name="FirstPageLayout"> (2)
            <xf:postscript-device-setup><![CDATA[
<</MediaPosition 3 /ManualFeed false>> setpagedevice
<</OutputType (LEFT OUTPUT BIN) /Staple 0>> setpagedevice ]]>
            </xf:postscript-device-setup>
            <fo:region-body region-name="xsl-region-body"
             margin="0.7in" border="5pt double black" padding="6pt" />
        </fo:simple-page-master>
        <fo:simple-page-master master-name="AllOtherPagesLayout"> (3)
            <xf:postscript-device-setup><![CDATA[
<</MediaPosition 1 /ManualFeed false>> setpagedevice
<</OutputType (LEFT OUTPUT BIN) /Staple 0>> setpagedevice]]>
            </xf:postscript-device-setup>
            <fo:region-body region-name="xsl-region-body"
             margin="0.7in" border="1pt solid black" padding="6pt" />
        </fo:simple-page-master>
        <fo:page-sequence-master master-name="default-sequence">
            <fo:single-page-master-reference
                          master-reference="FirstPageLayout"/>
            <fo:repeatable-page-master-reference
                          master-reference="OtherPagesLayout" />
        </fo:page-sequence-master>
    </fo:layout-master-set>
    <fo:page-sequence master-reference="default-sequence"> (4)
        <fo:flow flow-name="xsl-region-body">
            <fo:block>First Page</fo:block>
            <fo:block break-before="page">Second Page</fo:block>
            <fo:block break-before="page">Third Page</fo:block>
        </fo:flow>
    </fo:page-sequence>
</fo:root>
```

There are several things to notice:

(1) In order to work with extension you must have the corresponding xml namespace defined.
(2),**(3)** We have two simple-page-master elements each describing two different page layouts. The first page layout uses input tray number 3, while the second page layout uses input tray number 1.
(4) Page masters are aggregated into one page sequence. The page sequence is used to present a document with three pages.

Because the input tray numbers and output tray names are device dependent, you should obtain the PostScript Printer Description (PPD) file for your Postscript printer.

A PPD file describes printer features in such a way that a graphical user interface (GUI) can be built to present the user with a list of printer features to choose from. Examples of printer features include paper size, supported fonts, paper handling, and so on. A PPD file also contains PostScript and Job Control Language (JCL) commands to be used to invoke the selected features.

You can obtain the PPD file by searching the Internet (for example HP 4650 PPD) or by contacting the manufacturer. You may also have it already installed in one of subdirectories of **C:\WINDOWS\system32\spool\drivers**.
Once you have-it, look for "*DefaultInputSlot:" entry to find out what instructions must be sent to the printer in order to select the desired input tray, or "*DefaultOutputBin:" for the output tray.

Note:

- A PPD file may not have these sections if the printer has only one input and/or output tray.

1.16.16 Outline

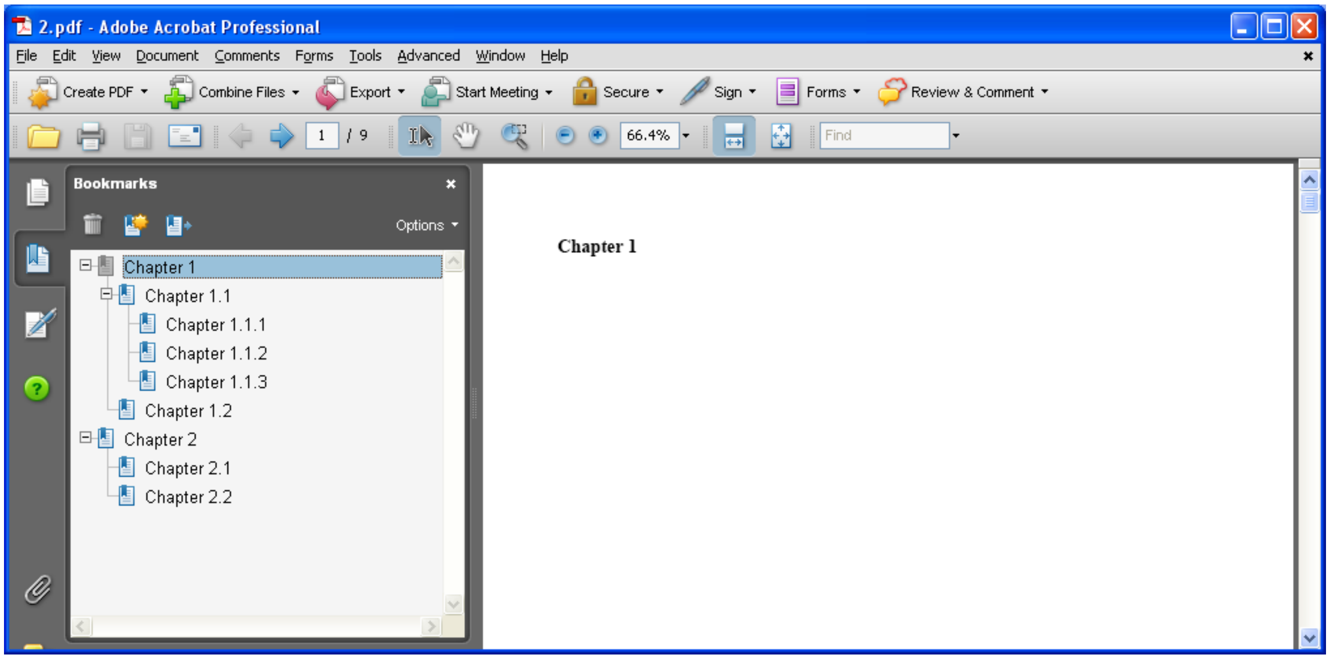
An **outline** is a hierarchical way to display related items of text to graphically depict their relationships (provide a summary showing the logical flow of a paper).

This section describes the extension used by **XF Rendering Server** in order to do that **fox:outline.**
In the following example it can be seen the way it is used:

```
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format"
                    xmlns:xf="http://www.ecrion.com/xf/1.0">
   <fox:outline xmlns:fox="http://xml.apache.org/fop/extensions" (1)
                                  internal-destination="ref1">
       <fox:label> Chapter 1 </fox:label>
       <fox:outline xmlns:fox="http://xml.apache.org/fop/extensions"
                            internal-destination="ref2"> (2)
              <fox:label> Chapter 1.1 </fox:label> (3)
              <fox:outline xmlns:fox="http://xml.apache.org/fop/extensions"
                            internal-destination="ref5"> (4)
                     <fox:label> Chapter 1.1.1 </fox:label>
              </fox:outline>
              <fox:outline xmlns:fox="http://xml.apache.org/fop/extensions"
                                  internal-destination="ref6">
                     <fox:label> Chapter 1.1.2 </fox:label>
              </fox:outline>
              <fox:outline xmlns:fox="http://xml.apache.org/fop/extensions"
                                  internal-destination="ref7">
                     <fox:label> Chapter 1.1.3 </fox:label>
              </fox:outline>
       </fox:outline>
       <fox:outline xmlns:fox="http://xml.apache.org/fop/extensions"
                                  internal-destination="ref3">
       <fox:label> Chapter 1.2 </fox:label>
       </fox:outline>
   </fox:outline>
   <fox:outline xmlns:fox="http://xml.apache.org/fop/extensions"
                                  internal-destination="ref4">
       <fox:label> Chapter 2 </fox:label>
       <fox:outline xmlns:fox="http://xml.apache.org/fop/extensions"
                                  internal-destination="ref8">
              <fox:label> Chapter 2.1 </fox:label>
       </fox:outline>
       <fox:outline xmlns:fox="http://xml.apache.org/fop/extensions"
                                  internal-destination="ref9">
              <fox:label> Chapter 2.2 </fox:label>
       </fox:outline>
   </fox:outline>
   <fo:layout-master-set>
       <fo:simple-page-master master-name="all-pages"
                            page-width="8.5in" page-height="11in">
              <fo:region-body region-name="xsl-region-body"
                     margin="0.7in" column-gap="0.25in" padding="6pt" />
              <fo:region-before region-name="xsl-region-before"
              extent="0.7in" display-align="after" padding="6pt 0.7in" />
              <fo:region-after region-name="xsl-region-after"
              extent="0.7in" display-align="before" padding="6pt 0.7in" />
       </fo:simple-page-master>
       <fo:page-sequence-master master-name="default-sequence">
              <fo:repeatable-page-master-reference
                                  master-reference="all-pages" />
```

```
      </fo:page-sequence-master>
   </fo:layout-master-set>
   <fo:page-sequence master-reference="default-sequence">
      <fo:static-content flow-name="xsl-region-before"
             font-family="Times New Roman" font-size="12pt">
             <fo:block>&#x00A0;</fo:block>
      </fo:static-content>
      <fo:static-content flow-name="xsl-region-after"
             font-family="Times New Roman" font-size="12pt">
             <fo:block>&#x00A0;</fo:block>
      </fo:static-content>
      <fo:flow flow-name="xsl-region-body"
      font-family="Times New Roman" font-size="14pt" font-weight="bold">
             <fo:block id="ref1" break-after="page">Chapter 1 </fo:block> (5)
             <fo:block id="ref2" break-after="page">Chapter 1.1</fo:block>
             <fo:block id="ref5" break-after="page">Chapter 1.1.1</fo:block>
             <fo:block id="ref6" break-after="page">Chapter 1.1.2</fo:block>
             <fo:block id="ref7" break-after="page">Chapter 1.1.3</fo:block>
             <fo:block id="ref3" break-after="page">Chapter 1.2</fo:block>
             <fo:block id="ref4" break-after="page">Chapter 2</fo:block>
             <fo:block id="ref8" break-after="page">Chapter 2.1</fo:block>
             <fo:block id="ref9" >Chapter 2.2</fo:block>
      </fo:flow>
   </fo:page-sequence>
</fo:root>
```

The result after exporting is the following:


2.pdf - Adobe Acrobat Professional
File Edit View Document Comments Forms Tools Advanced Window Help
Create PDF Combine Files Export Start Meeting Secure Sign Forms Review & Comment
1 / 9
66.4%
Find
Bookmarks
Options

Chapter 1
- Chapter 1.1
  - Chapter 1.1.1
  - Chapter 1.1.2
  - Chapter 1.1.3
- Chapter 1.2

Chapter 2
- Chapter 2.1
- Chapter 2.2

Chapter 1

Figure 34.

Things to notice in the example above:

(1) xmlns:fox="http://xml.apache.org/fop/extensions" must be defined in order to work with this extension.
(2) There is an **internal-destination** for every **fox:outline** in order to have a reference to the items from summary.

(3) A **fox:outline** contains a **fox:label** extension used to name the items from summary.
(4) A **fox:outline** can contain more fox:outline elements in order to make the summary more complex.
(5) After you set the outlines, in body region you made the references to the blocks corresponding for each item from summary.

1.16.17 Auto Adjust Regions

This section describes an extension used to auto-adjust regions from page (region-before and region-body): **xf:auto-adjust-regions** .
It can be applied only to **fo:simple-page-master** and supports boolean values.

```
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format"
             xmlns:xf="http://www.ecrion.com/xf/1.0"> (1)
   <fo:layout-master-set>
       <fo:simple-page-master master-name="all-pages"
                    page-width="8.5in" page-height="3.5in"
                           xf:auto-adjust-regions="true"> (2)
             <fo:region-body region-name="xsl-region-body"
             column-gap="0.25in" padding="6pt" margin="0.7in"/>
             <fo:region-before region-name="xsl-region-before"
             display-align="after" extent="0.7in" padding="6pt"/>
             <fo:region-after region-name="xsl-region-after"
             display-align="before" extent="0.7in" padding="6pt"/>
       </fo:simple-page-master>
       <fo:page-sequence-master master-name="default-sequence">
             <fo:repeatable-page-master-reference
                                   master-reference="all-pages"/>
       </fo:page-sequence-master>
   </fo:layout-master-set>
   <fo:page-sequence master-reference="default-sequence">
       <fo:static-content flow-name="xsl-region-before"
                    font-size="12pt" font-family="Times New Roman">
             <fo:block>header</fo:block>
             <fo:block>header</fo:block>
             <fo:block>header</fo:block>
             <fo:block>header</fo:block>
             <fo:block>header</fo:block>
             <fo:block>header</fo:block>
             <fo:block>header</fo:block>
             <fo:block>header</fo:block>
             <fo:block>header</fo:block>
       </fo:static-content>
       <fo:static-content flow-name="xsl-region-after"
                    font-size="12pt" font-family="Times New Roman">
             <fo:block>footer</fo:block>
             <fo:block>footer</fo:block>
             <fo:block>footer</fo:block>
       </fo:static-content>
       <fo:flow flow-name="xsl-region-body" font-size="12pt"
                           font-family="Times New Roman">
             <fo:block>body</fo:block>
             <fo:block>body</fo:block>
             <fo:block>body</fo:block>
       </fo:flow>
   </fo:page-sequence>
</fo:root>
```

The rendering result is displayed in the next figure:

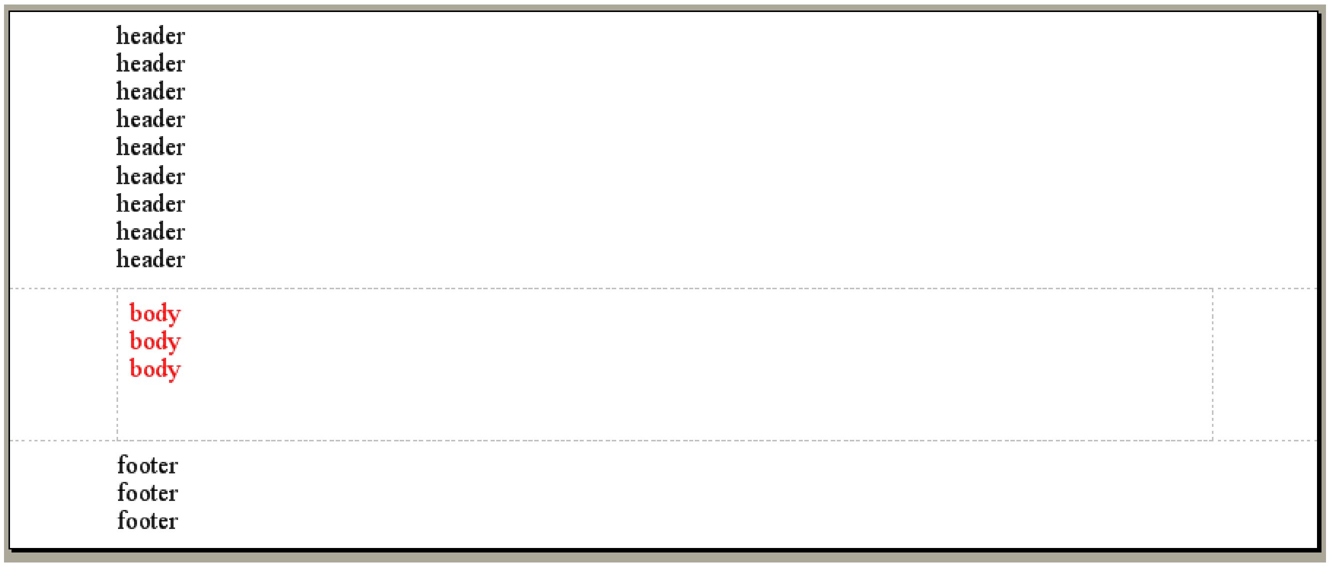

header
header
header
header
header
header
header
header
header
body
body
body
footer
footer
footer

Figure 35.

The important points in this example are:

(1) In order to work with extension you must have the corresponding xml namespace defined.
(2) **xf:auto-adjust-regions** applies to **fo:simple-page-master** and can take one of the values: **"true" | "false"**. Set on **"true"** is auto-adjusting page regions by their content.

1.16.18 Tag Group

This section describes **xf:tag-group**, an extension used while working with **.inx** output files.
It applies only on **fo:block** and assures that when you export your document this block is marked(tagged) even you have more inlines.

```
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format"
             xmlns:xf="http://www.ecrion.com/xf/1.0"> (1)
      <fo:layout-master-set>
             <fo:simple-page-master master-name="all-pages"
                          page-height="11in">
                   <fo:region-body region-name="xsl-region-body"
                   column-gap="0.25in" padding="6pt" margin="0.7in"/>
             </fo:simple-page-master>
             <fo:page-sequence-master master-name="default-sequence">
                   <fo:repeatable-page-master-reference
                                 master-reference="all-pages"/>
             </fo:page-sequence-master>
      </fo:layout-master-set>
      <fo:page-sequence master-reference="default-sequence">
             <fo:flow flow-name="xsl-region-body" font-size="12pt"
                          font-family="Times New Roman">
                   <fo:block xf:tag-group="true" start-indent="40.8px"
                   margin-right="367.8px">This is a paragraph that (2)
                          <fo:inline font-style="italic">will
                          </fo:inline>generate a group.</fo:block>
             </fo:flow>
      </fo:page-sequence>
</fo:root>
```

Things to notice:

(1) In order to work with extension you must have the corresponding xml namespace defined.
(2) **xf-tag-group** can take a boolean value; if its value is **"true"** all fo:inlines from the block will be seen like a group.

1.16.19 Page Number Extensions

You can set different options for page numbers using extensions like: **xf:skip-blank-pages**, **xf:initial-page-number**, **xf:use-sheet-number**, **xf:increment-by**.
The example below display the way these can be used:

```
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format"
                    xmlns:xf="http://www.ecrion.com/xf/1.0"> (1)
      <fo:layout-master-set>
            <fo:simple-page-master master-name="FirstPage"
            page-width="5in" page-height="1in" margin="0.2in">
                  <fo:region-body region-name="PageBody"/>
                  <fo:region-after region-name="Footer"
                                          extent="0.2in"/>
            </fo:simple-page-master>
            <fo:simple-page-master master-name="LetterPage"
            page-width="5in" page-height="1in" margin="0.2in">
                  <fo:region-body region-name="PageBody"/>
                  <fo:region-after region-name="Footer"
                                          extent="0.2in"/>
            </fo:simple-page-master>
            <fo:simple-page-master master-name="BlankPage"
            page-width="5in" page-height="1in" margin="0.2in">
                  <fo:region-body region-name="PageBody"/>
                  <fo:region-after region-name="Footer"
                                          extent="0.2in"/>
            </fo:simple-page-master>
            <fo:page-sequence-master master-name="alternate">
                  <fo:repeatable-page-master-alternatives
                                    maximum-repeats="no-limit">
                        <fo:conditional-page-master-reference
                                    master-reference="FirstPage"/>
                        <fo:conditional-page-master-reference
                        master-reference="BlankPage"
                        blank-or-not-blank="blank"
                        odd-or-even="even"/>
                        <fo:conditional-page-master-reference
                        master-reference="LetterPage"
                        odd-or-even="odd"/>
                  </fo:repeatable-page-master-alternatives>
            </fo:page-sequence-master>
      </fo:layout-master-set>
      <fo:page-sequence master-reference="alternate" font="10pt Arial">
            <fo:static-content flow-name="Footer">
                  <fo:block text-align="right" border-top="1pt
                              solid black" padding-top="1mm">
                        Page
                        <fo:page-number xf:skip-blank-pages="true" (2)
                        xf:initial-page-number="1" (3)
                        xf:use-sheet-number="false" (4)
                        xf:increment-by="1"/> (5)
                        of
                        <fo:page-number-citation ref-id="theEnd"
                        xf:skip-blank-pages="true"/>
                  </fo:block>
            </fo:static-content>
            <fo:flow flow-name="PageBody">
                  <fo:block>
                        The text content of the first page.
                  </fo:block>
```

```
                    <fo:block break-before="page">
                          The text content of the second page.
                    </fo:block>
                    <fo:block break-after="page"></fo:block>
                    <fo:block break-after="page"></fo:block>
                    <fo:block break-after="page"></fo:block>
                    <fo:block break-after="page"></fo:block>
                    <fo:block break-after="page"></fo:block>
                    <fo:block break-after="page"></fo:block>
                    <fo:block break-after="page"></fo:block>
                    <fo:block id="theEnd">
                          The text content of the last page.
                    </fo:block>
             </fo:flow>
      </fo:page-sequence>
</fo:root>
```

Things to notice:

(1) In order to work with extension you must have the corresponding xml namespace defined.
(2) **xf:skip-blank-pages** can be applied to **fo:page-number** and **fo:page-number-citation** also; it takes boolean values and if is set on "true" that means that blank pages won't have page-numbers.
(3) **xf:initial-page-number** can be applied only to **fo:page-number.** Can take one of the following values: **"auto" | "auto-odd" | "auto-even" or a integer value.**
This extension set the page number for the first page that will be incremented for every page that follows with the value set by **xf:increment-by (5)** which has the default value **"1"**.
(4) **xf:use-sheet-numbers** is used when you need to number sheets, not every page.

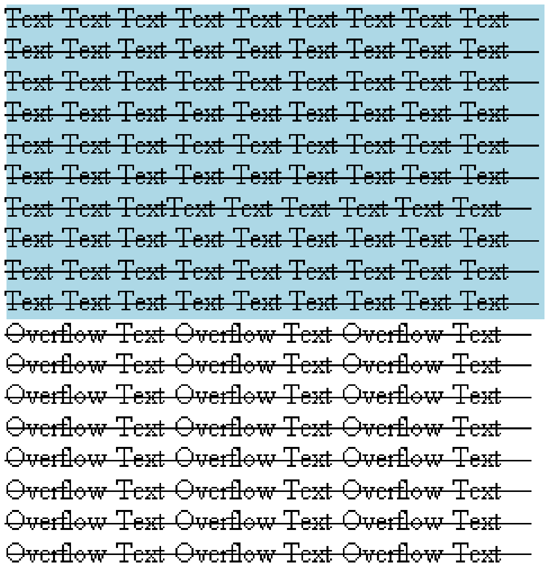
1.16.20 Overflow Errors

xf:show-overflow-errors is used to mark the text that doesn't match in his corresponding field or space.

Consider the following example: you insert a floating region into a document. Start typing text. You will notice that at a certain moment the text doesn't fill anymore in the region and it is written overflow. It can be seen if you have the **overflow-hidden** attribute set on **"false"**.

```
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format"
             xmlns:xf="http://www.ecrion.com/xf/1.0"> (1)
   <fo:layout-master-set>
      <fo:simple-page-master master-name="all-pages"
                   page-width="8.5in" page-height="5in">
             <fo:region-body region-name="xsl-region-body"
             column-gap="0.25in" padding="6pt" margin="0.7in"/>
             <fo:region-before region-name="xsl-region-before"
             display-align="after" extent="0.7in" padding="6pt"/>
             <fo:region-after region-name="xsl-region-after"
             display-align="before" extent="0.7in" padding="6pt"/>
      </fo:simple-page-master>
      <fo:page-sequence-master master-name="default-sequence">
             <fo:repeatable-page-master-reference
                          master-reference="all-pages"/>
      </fo:page-sequence-master>
   </fo:layout-master-set>
   <fo:page-sequence master-reference="default-sequence">
      <fo:static-content flow-name="xsl-region-before"
             font-size="12pt" font-family="Times New Roman">
             <fo:block> </fo:block>
      </fo:static-content>
      <fo:static-content flow-name="xsl-region-after"
             font-size="12pt" font-family="Times New Roman">
             <fo:block> </fo:block>
      </fo:static-content>
      <fo:flow flow-name="xsl-region-body" font-size="12pt"
                                 font-family="Times New Roman">
             <fo:block-container z-index="1" position="absolute"
             left="14px" top="4.45468px" width="299px" height="175px"
             xf:show-overflow-errors="true"
             background-color="rgb(173,216,230)"> (2)
                   <fo:block>Text Text Text Text Text Text Text Text
                    ...
                    Overflow Text Overflow Text Overflow Text...
                   </fo:block>
             </fo:block-container>
      </fo:flow>
   </fo:page-sequence>
</fo:root>
```

The rendering result is displayed in the picture below:



~~Text Text Text Text Text Text Text Text Text~~
~~Text Text Text Text Text Text Text Text Text~~
~~Text Text Text Text Text Text Text Text Text~~
~~Text Text Text Text Text Text Text Text Text~~
~~Text Text Text Text Text Text Text Text Text~~
~~Text Text Text Text Text Text Text Text Text~~
~~Text Text TextText Text Text Text Text Text~~
~~Text Text Text Text Text Text Text Text Text~~
~~Text Text Text Text Text Text Text Text Text~~
~~Text Text Text Text Text Text Text Text Text~~
~~Overflow Text Overflow Text Overflow Text~~
~~Overflow Text Overflow Text Overflow Text~~
~~Overflow Text Overflow Text Overflow Text~~
~~Overflow Text Overflow Text Overflow Text~~
~~Overflow Text Overflow Text Overflow Text~~
~~Overflow Text Overflow Text Overflow Text~~
~~Overflow Text Overflow Text Overflow Text~~
~~Overflow Text Overflow Text Overflow Text~~

Figure 36.

Things to notice:

(1) In order to work with extension you must have the corresponding xml namespace defined.
(2) **xf:show-overflow-errors** supports boolean values; if set on **"true"**, if the text doesn't mach in the available space it is totally marked by a linethrough.

1.16.21 Keep Together Extension

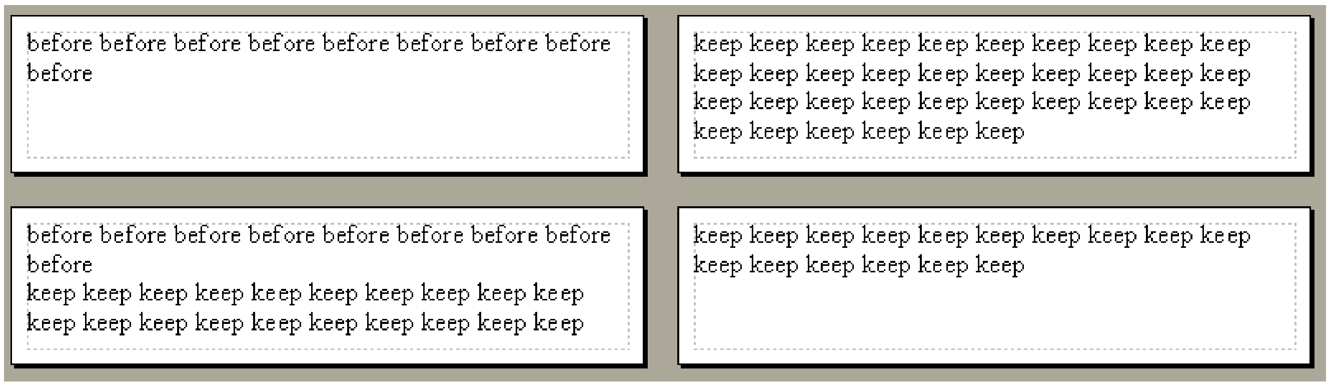
xf:keep-together-parity.within-page is an extension that can be applied on **fo:block**, **fo:list-block**, **fo:list-item-part**, **fo:table**, **fo:table-and-caption**, **fo:table-row-group** and **fo:block-container**.

This extension controls the flow of the text from a page to another. It can take one of the following values: **"odd" | "even" | "any"**.

Below is a short example for you to see the way it can be used:

```
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format"
             xmlns:xf="http://www.ecrion.com/xf/1.0"> (1)
      <fo:layout-master-set>
            <fo:simple-page-master master-name="all-pages"
                                    page-height="1in" page-width="4in">
                  <fo:region-body region-name="xsl-region-body"/>
            </fo:simple-page-master>
            <fo:page-sequence-master master-name="default-sequence">
                  <fo:repeatable-page-master-reference
                                          master-reference="all-pages" />
            </fo:page-sequence-master>
      </fo:layout-master-set>
      <fo:page-sequence master-reference="default-sequence">
            <fo:flow flow-name="xsl-region-body"
                  font-family="Times New Roman" font-size="12pt">
                  <fo:block>
                        before before before before before
                        before before before before before
                  </fo:block>
      <fo:block keep-together.within-page="always"
                        xf:keep-together-parity.within-page="Odd"> (2)
                              keep keep keep keep keep keep
                        keep keep keep keep keep keep
                        keep keep keep keep keep keep
                        keep keep keep keep keep keep
                        keep keep keep keep keep keep
                        keep keep keep keep keep keep
                  </fo:block>
                  <fo:block>
                        before before before before before
                        before before before before before
                  </fo:block>
      <fo:block keep-together.within-page="always"
                        xf:keep-together-parity.within-page="Even"> (3)
                              keep keep keep keep keep keep
                        keep keep keep keep keep keep
                        keep keep keep keep keep keep
                        keep keep keep keep keep keep
                        keep keep keep keep keep keep
                        keep keep keep keep keep keep
                  </fo:block>
            </fo:flow>
      </fo:page-sequence>
</fo:root>
```

The rendering result is displayed in the picture below:


before before before before before before before before before

keep keep keep keep keep keep keep keep keep keep keep keep keep keep keep keep keep keep keep keep keep keep keep keep keep keep keep keep keep keep keep keep keep keep keep keep

before before before before before before before before before
keep keep keep keep keep keep keep keep keep keep keep keep keep keep keep keep keep keep keep keep

keep keep keep keep keep keep keep keep keep keep keep keep keep keep keep keep

Figure 39.

Things to notice:

(1) In order to work with extensions you must have the corresponding xml namespace defined.
(2) This extension is used to keep a block within page only if the page is odd or even. In this example first block "keep keep...." has the extension set to "odd" that means it begins only on odd pages. The second one has **keep-together-parity.within-page** set on "even" that means it begins only on even pages and because the previous block is on an even page appears on the same page, right after it.
To understand better imagine an open book with left pages - even pages and right ones - odd pages.

Note:
- It is very important to know that if you add new pages, the text is moved on the third page as a result of "keep-together" attribute properties and "even" value set as parity.

1.17 Appendix A - Colors

A color value may either be a hexadecimal number (prefixed by a hash mark) or one of the following color names.
The color names are case-insensitive.

⬜	aliceblue	rgb(240, 248, 255)
⬜	antiquewhite	rgb(250, 235, 215)
🟦	aqua	rgb(0, 255, 255)
🟩	aquamarine	rgb(127, 255, 212)
⬜	azure	rgb(240, 255, 255)
⬜	beige	rgb(245, 245, 220)
⬜	bisque	rgb(255, 228, 196)
⬛	black	rgb(0, 0, 0)
⬜	blanchedalmond	rgb(255, 235, 205)
🟦	blue	rgb(0, 0, 255)
🟪	blueviolet	rgb(138, 43, 226)
🟥	brown	rgb(165, 42, 42)
🟫	burlywood	rgb(222, 184, 135)
🟦	cadetblue	rgb(95, 158, 160)
🟩	chartreuse	rgb(127, 255, 0)
🟧	chocolate	rgb(210, 105, 30)
🟧	coral	rgb(255, 127, 80)
🟦	cornflowerblue	rgb(100, 149, 237)
⬜	cornsilk	rgb(255, 248, 220)
🟥	crimson	rgb(220, 20, 60)
🟦	cyan	rgb(0, 255, 255)
🟦	darkblue	rgb(0, 0, 139)
🟦	darkcyan	rgb(0, 139, 139)
🟨	darkgoldenrod	rgb(184, 134, 11)
⬛	darkgray	rgb(169, 169, 169)
🟩	darkgreen	rgb(0, 100, 0)
⬛	darkgrey	rgb(169, 169, 169)
🟨	darkkhaki	rgb(189, 183, 107)
🟪	darkmagenta	rgb(139, 0, 139)
🟩	darkolivegreen	rgb(85, 107, 47)
🟧	darkorange	rgb(255, 140, 0)
🟪	darkorchid	rgb(153, 50, 204)
🟥	darkred	rgb(139, 0, 0)
🟥	lightpink	rgb(255, 182, 193)
🟧	lightsalmon	rgb(255, 160, 122)
🟦	lightseagreen	rgb(32, 178, 170)
🟦	lightskyblue	rgb(135, 206, 250)
⬛	lightslategray	rgb(119, 136, 153)
⬛	lightslategrey	rgb(119, 136, 153)
🟦	lightsteelblue	rgb(176, 196, 222)
⬜	lightyellow	rgb(255, 255, 224)
🟩	lime	rgb(0, 255, 0)
🟩	limegreen	rgb(50, 205, 50)
⬜	linen	rgb(250, 240, 230)
🟪	magenta	rgb(255, 0, 255)
🟥	maroon	rgb(128, 0, 0)
🟩	mediumaquamarine	rgb(102, 205, 170)
🟦	mediumblue	rgb(0, 0, 205)
🟪	mediumorchid	rgb(186, 85, 211)
🟪	mediumpurple	rgb(147, 112, 219)
🟩	mediumseagreen	rgb(60, 179, 113)
🟪	mediumslateblue	rgb(123, 104, 238)
🟩	mediumspringgreen	rgb(0, 250, 154)
🟦	mediumturquoise	rgb(72, 209, 204)
🟪	mediumvioletred	rgb(199, 21, 133)
🟦	midnightblue	rgb(25, 25, 112)
⬜	mintcream	rgb(245, 255, 250)
⬜	mistyrose	rgb(255, 228, 225)
🟨	moccasin	rgb(255, 228, 181)
🟨	navajowhite	rgb(255, 222, 173)
🟦	navy	rgb(0, 0, 128)
⬜	oldlace	rgb(253, 245, 230)
🟩	olive	rgb(128, 128, 0)
🟩	olivedrab	rgb(107, 142, 35)
🟧	orange	rgb(255, 165, 0)
🟥	orangered	rgb(255, 69, 0)

🟧	darksalmon	rgb(233, 150, 122)
🟩	darkseagreen	rgb(143, 188, 143)
🟪	darkslateblue	rgb(72, 61, 139)
⬛	darkslategray	rgb(47, 79, 79)
⬛	darkslategrey	rgb(47, 79, 79)
🟦	darkturquoise	rgb(0, 206, 209)
🟪	darkviolet	rgb(148, 0, 211)
🟥	deeppink	rgb(255, 20, 147)
🟦	deepskyblue	rgb(0, 191, 255)
⬛	dimgray	rgb(105, 105, 105)
⬛	dimgrey	rgb(105, 105, 105)
🟦	dodgerblue	rgb(30, 144, 255)
🟥	firebrick	rgb(178, 34, 34)
⬜	floralwhite	rgb(255, 250, 240)
🟩	forestgreen	rgb(34, 139, 34)
🟪	fuchsia	rgb(255, 0, 255)
⬜	gainsboro	rgb(220, 220, 220)
⬜	ghostwhite	rgb(248, 248, 255)
🟨	gold	rgb(255, 215, 0)
🟨	goldenrod	rgb(218, 165, 32)
⬛	gray	rgb(128, 128, 128)
⬛	grey	rgb(128, 128, 128)
🟩	green	rgb(0, 128, 0)
🟩	greenyellow	rgb(173, 255, 47)
⬜	honeydew	rgb(240, 255, 240)
🟥	hotpink	rgb(255, 105, 180)
🟥	indianred	rgb(205, 92, 92)
🟪	indigo	rgb(75, 0, 130)
⬜	ivory	rgb(255, 255, 240)
🟨	khaki	rgb(240, 230, 140)
⬜	lavender	rgb(230, 230, 250)
⬜	lavenderblush	rgb(255, 240, 245)
🟩	lawngreen	rgb(124, 252, 0)
⬜	lemonchiffon	rgb(255, 250, 205)
🟦	lightblue	rgb(173, 216, 230)
🟥	lightcoral	rgb(240, 128, 128)
⬜	lightcyan	rgb(224, 255, 255)
⬜	lightgoldenrodyellow	rgb(250, 250, 210)
⬜	lightgray	rgb(211, 211, 211)
🟩	lightgreen	rgb(144, 238, 144)
🟪	orchid	rgb(218, 112, 214)
🟨	palegoldenrod	rgb(238, 232, 170)
🟩	palegreen	rgb(152, 251, 152)
🟦	paleturquoise	rgb(175, 238, 238)
🟥	palevioletred	rgb(219, 112, 147)
⬜	papayawhip	rgb(255, 239, 213)
🟧	peachpuff	rgb(255, 218, 185)
🟫	peru	rgb(205, 133, 63)
🟥	pink	rgb(255, 192, 203)
🟪	plum	rgb(221, 160, 221)
🟦	powderblue	rgb(176, 224, 230)
🟪	purple	rgb(128, 0, 128)
🟥	red	rgb(255, 0, 0)
🟫	rosybrown	rgb(188, 143, 143)
🟦	royalblue	rgb(65, 105, 225)
🟫	saddlebrown	rgb(139, 69, 19)
🟧	salmon	rgb(250, 128, 114)
🟧	sandybrown	rgb(244, 164, 96)
🟩	seagreen	rgb(46, 139, 87)
⬜	seashell	rgb(255, 245, 238)
🟫	sienna	rgb(160, 82, 45)
⬜	silver	rgb(192, 192, 192)
🟦	skyblue	rgb(135, 206, 235)
🟪	slateblue	rgb(106, 90, 205)
⬛	slategray	rgb(112, 128, 144)
⬛	slategrey	rgb(112, 128, 144)
⬜	snow	rgb(255, 250, 250)
🟩	springgreen	rgb(0, 255, 127)
🟦	steelblue	rgb(70, 130, 180)
🟫	tan	rgb(210, 180, 140)
🟦	teal	rgb(0, 128, 128)
🟪	thistle	rgb(216, 191, 216)
🟥	tomato	rgb(255, 99, 71)
🟦	turquoise	rgb(64, 224, 208)
🟪	violet	rgb(238, 130, 238)
🟨	wheat	rgb(245, 222, 179)
⬜	white	rgb(255, 255, 255)
⬜	whitesmoke	rgb(245, 245, 245)
🟨	yellow	rgb(255, 255, 0)
🟩	yellowgreen	rgb(154, 205, 50)