Mastering Antenna House XSL Formatter Extensions

Antenna House, Inc.

Preface

Extensible Stylesheet Language (XSL) is a specification for converting Extensible Markup Language (XML) documents into a format for formatting. XSL is divided into XSLT, which is a specification for converting XML documents, and XSL Formatting Objects (XSL-FO), which is an XML specification for formatted objects. The “Cascading Style Sheet (CSS)” is globally recognized as a “style sheet”, but XSL-FO is a specification to express XML documents as page media, based on CSS 2 and DSSSL. For more information, see W3C's XSL specifications, etc.

The XSL-FO specification, XSL 1.1, became a W3C Recommendation in December 2006. Antenna House XSL Formatter (AH XSL Formatter) faithfully implements most of the specifications of XSL 1.1. However, many users who actually use XSL-FO in their business, etc. require extra features to more finely control formatting than is defined in XSL 1.1. AH XSL Formatter has implemented a variety of extensions in response to user requirements. This book features the kinds of documents that can be formatted with XSL-FO using AH XSL Formatter extensions, and what kind of formatting requirements can be met.

The AH XSL Formatter manual is updated as needed with version upgrades. If there are any discrepancies between this book and the manual, the descriptions and explanations in the manual take precedence.

This book has been formatted using AH XSL Formatter V7.2 MR4. The AH XSL Formatter features shown here are based on V7.1.

Audience

For the explanation of the extended specifications and usage, we assume that you have some knowledge of the XSL-FO specifications, including page sequences and block containers. In addition, we have tried to make this book work as a showcase to introduce the kinds of formatting that can be achieved with AH XSL Formatter for those who want to “output XML documents in accordance with their requirements”, “realize high quality automatic formatting”, and “know the output example of how it is possible to format based on XML”.

Most examples show PDF output.

Notation

In this book, literal FOs, `properties`, and `values` are shown in a `monospace` font. An FO name is written as `<fo:page-sequence>`. When an FO or property name contains “*”, the “*” is a wildcard to indicate all matching names.

Italic text enclosed with LESS-THAN SIGN “<” (U+003C) and GREATER-THAN SIGN “>” (U+003E), such as *<number>*, is a placeholder that indicates the type of the value rather than a specific value.

When referring to a specific character, the name in Unicode, “example of glyph image (excluding characters that indicate white spaces)”, and Unicode code point is indicated as “U+” and a hexadecimal value. For example, the letter “*あ*” is described as HIRAGANA LETTER A “あ” (U+3042).

For improved readability, some FO examples have more line breaks and indentation than should be used in a production document.

“✓” in tables indicates that an item can be specified, and “-” indicates that an item cannot be specified.

Terms

before, after, start, end: top, bottom, left, and right are absolute positions or directions. However, in formatting, it is sometimes inconvenient to specify absolute position. before, after, start, and end are relative positions that change depending on the writing mode. With `writing-mode="lr-tb"`, left corresponds to start and top corresponds to before.

Inline progression direction: The direction in which characters are placed inline. In Japanese formatting, this corresponds to “inline direction” in JIS X 4051 “Formatting rules for Japanese documents” and JIS X 4052 “Exchange format for Japanese documents with composition markup”. In this book, it refers to the start-to-end direction.

Block progression direction: The direction in which the number of rows increases. In Japanese formatting, it corresponds to “block direction” in JIS X 4051 “Formatting rules for Japanese documents” and JIS X 4052 “Exchange format for Japanese documents with composition markup”. In this book, it refers to the before-to-after direction.

Columns: A multi-column format of two or more columns. (Contexts about tables, column refers to the column in the table.)

Tab: The formatting object used for tab alignment is called “tab” and the tab character is called TAB(U+0009).

Language code: A language code conforming to ISO 639 or ISO 639-2.

Language tag : A language code and an optional country code conforming to ISO 3166, joined by a hyphen.

Script: A script code conforming to ISO 15924, or the characters and set of notation symbols that correspond to the script code.

European: The scripts that are used mainly in Europe. Scripts classified as European include Latn (Latin scripts) and Cyrl (Cyrillic scripts), etc. Latin scripts include the LATIN CAPITAL LETTER A “A” (U+0041). Cyrillic scripts include the CYRILLIC CAPITAL LETTER DE “Д” (U+0414).

CJK: The Chinese, Japanese and Korean languages or scripts are collectively referred to as “CJK”.

Japanese: Scripts such as hiragana, katakana, kanji, full-width alphabets, and punctuation characters appearing in Japanese texts are collectively described as “Japanese”. The non-European examples mainly apply to Japanese text, but some also apply to all CJK.

Unicode ‘General_Category’ property: ‘General_Category’ in the Unicode Character Database (http://www.unicode.org/reports/tr44/#General_Category_Values) classifies characters as upper case and lower case, numbers and so on. For example, the General Category of LATIN CAPITAL LETTER A “A” (U+0041) is Uppercase_Letter (Lu). General Category values are shown using their two-letter abbreviations.

Manual: The AH XSL Formatter manual. The English version of the manual is available at https://www.antenna.co.jp/AHF/help/en/.

Option setting: AH XSL Formatter operational setting. Options can be read from an XML file when running AH XSL Formatter, or, using an AH XSL Formatter extension, they can be specified in `<fo:declarations>`.

Extended specification items

There are two types of extended specifications: property extensions and element extensions. For specifications that are based on the XSL 1.1 specification or the CSS3 specification, extended coverage or extended specifiable values are shown in **bold**. The values that have not been requested and are not provided by AH XSL Formatter are indicated by strikethrough.

Some of the operational settings of AH XSL Formatter can only be controlled as option settings. These are also shown in the examples. The namespace of the option setting is included only when necessary.

Extended elements and elements of option settings

Element	Name of the extended element.
Property	Property that needed to be specified or useful.
Parent	Parent element that can describe the extended element.

Option Element is used for option setting elements.

Parent is used when the element can only occur in a limited range of parent elements.

Extended properties and option setting properties

Property	Name of the extended propert
Value	Value that can be specified to the property.
Initial	Initial value when no property is specified.
Applies to	Element that can specify the property.
Inherited	Whether to inherit the value from the ancestor's setting. Specify "yes" only when inheriting.
Percentages	Target to be used as a reference when specifying a value with *<percentage>*.

Option Property is used for option setting properties.

In this book, Inherited is shown only when the property is inherited, and Percentages is shown only when it can be specified by *<percentage>* and the target of the reference exists in FO.

The following list shows how to read Value tables. Some symbols specify combinations and some specify repetition.

- The simply ordered values must all appear in order.
- The values separated by "&&" must all appear, in any order.
- One or more of the values separated by "||" must appear, in any order.
- Exactly one of the values separated by "|" must appear.
- The contents enclosed by "[" and "]" are treated as a group.
- "*" indicates that the value or group occurs zero or more times.
- "+" indicates that the value or group occurs one or more times.
- "?" indicates that the value or group occurs zero or one times.
- "#" indicates that the value or group occurs one or more times, each separated by a COMMA "," (U+002C)
- "{N}" indicates that the value or group occurs N times.

- “{N,}” indicates that the value or group occurs N or more times.
- “{N,M}” indicates that the value or group occurs N or more times but not more than M times.
- “!” after a group indicates that the group is required, and at least one of them must appear.

Value

Some of the values specified for properties are explained in the body of the text, while others are explained in tables.

<property>	Definition
Value specified for the *<property>*.	The behavior when Value is specified.

Complex Value

There are some properties and functions that require complex specification. In this case, some property tables denote a possible Value, and the details are shown in another table.

Value	Value Composition
<Complex Value>	*<Complex Value Composition>*

Function (e.g. math, programming)

Some property values can include functions.

Function	Name of the function.
<Function arg>	Possible values for function arguments

Trademarks

Contents

Chapter 1

Showcases

Section 1.1 Document types and extension use cases

AH XSL Formatter implements most of the XSL 1.1 specification. It also implements the draft XSL 2.0 specification, the specifications developed for CSS3 and many other extensions based on user requests. This section introduces common document types with examples of extensions that are useful when formatting them.

Magazines and Brochures

These require control of photographic image resolution, complex columns and flexible styling of text and blocks.

- Spot color specification
- Multiple types of columns
- Fine control of text overflow

Literary books

Biographies and novels require detailed arrangement of the text, supplementary elements such as ruby and notes, and layout of illustrations in specific positions.

- Fine control of character spacing, line spacing and paragraphs
- Initial letter
- Ruby and emphasis marks
- Fine float layout

Academic journals

Academic journals require as much text as possible to be fitted into a fixed number of pages, as well as a variety of symbols and elegant mathematical formatting.

- Fine control of character spacing, line spacing and paragraphs
- Control of variant characters
- Support for a large number of scripts
- Math expressions based on MathML
- Insert and output vector images such as SVG and CGM
- Fine control over floated figures and tables

Forms

It is possible to output editable forms that can be input from a PDF viewer in a flexible layout.

- Various decorations of blocks and text
- PDF interactive form

Hardware and software manuals

Manuals require formatting that can cope with enormous number of pages and a variety of figures and tables.

- Insert and output vector images such as SVG and CGM
- Fine float layout
- Tables that don't fit on one page
- Charts or page inserts of differing sizes
- Generation of table of contents and index
- PDF output as multiple separate volumes
- Embed and attach videos and files

The remainder of this chapter shows example documents formatted using AH XSL Formatter extensions.

Section 1.2　Brochure-style

You can create pages that use a large background image but still have readable text.

The example below includes: one background image across two facing pages; a heading with separate stroke color; body regions with semi-transparent backgrounds; and images floated within columns [Figure 1.2-1] (p.22).

Figure 1.2-1　Brochure-style page design

Two facing pages

```
    <fo:layout-master-set>
        <!-- simple-page-master LEFT -->
        <fo:simple-page-master master-name="left"
            page-width="210mm" page-height="295mm"
            margin-left="0mm">
            <fo:region-before region-name="leftSection"
                extent="20mm"/>
            <fo:region-body margin-top="65mm" margin-bottom="20mm"
                margin-left="30mm" margin-right="15mm" padding="4m
m"
                column-count="2" column-gap="24pt"
                background-color="rgba(200, 200, 180,0.8)"/>
```

```
        </fo:simple-page-master>
        <!-- simple-page-master RIGHT -->
        <fo:simple-page-master master-name="right"
            page-width="210mm" page-height="295mm"
            margin-right="0mm">
            <fo:region-before region-name="rightSection"
                extent="20mm"/>
            <fo:region-body margin-top="65mm" margin-bottom="20mm"
                margin-left="15mm" margin-right="30mm" padding="4m
m"
                column-count="2" column-gap="24pt"
                background-color="rgba(200, 200, 180,0.8)"/>
        </fo:simple-page-master>
        <!-- facing page SPREADPM -->
        <axf:spread-page-master master-name="spreadpm"
            left-page-master-reference="left"
            right-page-master-reference="right"
            background-image="showcase_brochure_2.jpeg"
            axf:background-image-resolution="normal"
            axf:background-content-height="scale-to-fit"
            background-position-vertical="bottom"
            background-repeat="no-repeat">
            <axf:spread-region region-name="chapter" margin="20mm"
                margin-bottom="250mm"/>
        </axf:spread-page-master>
        <!-- page-sequence-master PSM -->
        <fo:page-sequence-master master-name="psm">
            <fo:repeatable-page-master-reference
                master-reference="spreadpm"/>
            <fo:repeatable-page-master-alternatives>
                <fo:conditional-page-master-reference
                    master-reference="left" odd-or-even="even"/>
                <fo:conditional-page-master-reference
                    master-reference="right" odd-or-even="odd"/>
            </fo:repeatable-page-master-alternatives>
        </fo:page-sequence-master>
    </fo:layout-master-set>
```

To use different page headers on left and right pages, use the `left-page-master-reference` and `right-page-master-reference` properties to specify different page masters for the left and right pages.

To avoid overlapping the body region and the heading section, the top margin of the body region is specified to be wide.

It is not always possible to compose all pages as sets of two facing pages, so specify two facing pages and left and right page masters respectively in the page sequence master. Page sequences refer to these page sequence masters.

To display a spread page when opened in a PDF viewer, specify an `<axf:document-info>` element with `name="pagelayout"` and `value="TwoPageLeft"`.

Background image across facing pages

```
<!--  Spread page master - -->
<fo:layout-master-set>
  ...
  <axf:spread-page-master master-name="spreadpm"
    left-page-master-reference="left"
    right-page-master-reference="right"
    background-image="showcase_brochure_2.jpeg"
    background-content-height="scale-to-fit"
    background-position-vertical="bottom"
    axf:background-image-resolution="normal"
    background-repeat="no-repeat">
    ...
  </axf:spread-page-master>
  ...
<fo:layout-master-set>
```

You can generate a background image over two facing pages by specifying the image in the `background-image` of an `<axf:spread-page-master>`. This is useful for brochures that use expansive landscape photography.

Also, `axf:background-image-resolution` specifies the resolution of the background image.

Outline color on heading that spans facing pages

```
<!-- Spread region -->
<fo:layout-master-set>
  ...
  <axf:spread-page-master master-name="spreadpm" ...>
    <axf:spread-region region-name="chapter"
      margin="20mm" margin-bottom="250mm" />
    ...
  </axf:spread-page-master>
  ...
</fo:layout-master-set>
<fo:page-sequence master-reference="psm"
  <!-- Flow of the Spread region -->
  <fo:flow flow-name="chapter">
    <fo:block-container height="100%"
      axf:text-stroke="2pt white"
      color="cmyk(80%, 30%, 10%, 70%)"
      font-size="86pt" font-weight="bold"
      text-align="center"
      font-family="sans-serif">
```

```
        <fo:block padding="3mm"
          >Go to the Sea!!</fo:block>
    </fo:block-container>
  </fo:flow>
  ...
</fo:page-sequence>
```

The heading is placed in a spread region that is higher on the page than the body text. `flow-name` matches the spread region name.

Because the text would be hard to see against the background, `axf:text-stroke` adds a white outline to the text.

Body region with semi-transparent background

```
<!-- Set a transparent color for the background of the body regio
n -->
<fo:simple-page-master master-name="left"
  ...>
  ...
  <fo:region-body ...
    background-color="rgba(200, 200, 180,0.8)" />
</fo:simple-page-master>
```

The idea is to ensure the readability of the text while showing the background image. It is possible to place a light color under the text as an alternative to specifying either a text outline or a text color that is easily distinguished from the background. The semi-transparent color is specified by using `rgba()` for the `background-color` property of `<fo:region-body>`.

Bottom-aligned images

```
<!-- Float including the image -->
<fo:float axf:float-y="bottom">
  <fo:block text-indent="0" text-align="center"
    space-before="1.0em" space-after="1.0em">
    <fo:external-graphic
      src="showcase_brochure_1.png"
      width="48mm" content-width="scale-to-fit"/>
  </fo:block>
  <fo:block text-indent="0" text-align="center"
    space-before="1.0em" space-after="1.0em">
    <fo:external-graphic src="showcase_brochure_3.png"
      width="48mm" content-width="scale-to-fit" />
```

```
    </fo:block>
</fo:float>
```

By specifying bottom for `axf:float-y`, the float is aligned to the bottom.

In this example, the images are grouped in a single float, and the space between blocks is specified with the `space-*` properties. If you generate each image as a separate float, you can adjust the space between images with `axf:float-float-margin-*` or other.

Section 1.3 Japanese novel with vertical text

You can create pages that are mainly vertical writing in Japanese.

Combine multiple writing directions, line grids, ruby, sidenotes, punctuation characters, number formatting and more [Figure 1.3-1] (p.27).

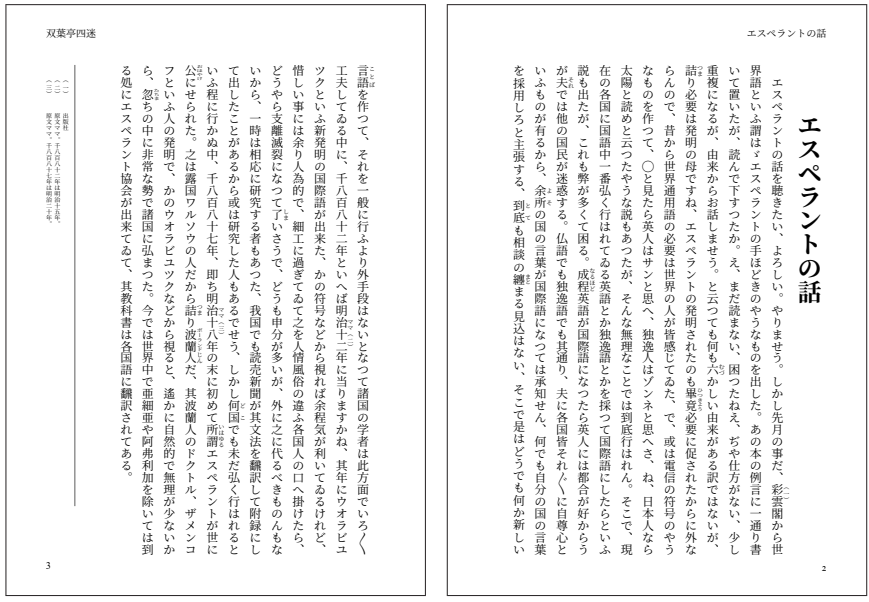


Figure 1.3-1 Japanese novel with vertical text

A6 page for vertical text

```
<!--  Right and left page master -->
<fo:root
  font-size="12q" line-height="1.8" ... >
  <fo:layout-master-set>
    <fo:simple-page-master master-name="left"
      size="A6 portrait" writing-mode="tb-rl">
      <fo:region-start region-name="left-header"
        margin-left="5mm" extent="12mm" />
      <fo:region-end region-name="left-footer"
        extent="10mm"/>
      <fo:region-body margin-top="15mm" margin-bottom="10mm"
        margin-left="10mm" margin-right="14mm"
        axf:footnote-position="odd-page"/>
```

```
    </fo:simple-page-master>
    <fo:simple-page-master master-name="right"
      size="A6 portrait" writing-mode="tb-rl">
      <fo:region-start region-name="right-header"
        margin-right="5mm" extent="12mm" />
      <fo:region-end region-name="right-footer"
        extent="9mm"/>
      <fo:region-body margin-top="15mm" margin-bottom="10mm"
        margin-left="14mm" margin-right="10mm"
        axf:footnote-position="odd-page"/>
    </fo:simple-page-master>
    <fo:page-sequence-master master-name="main">
      <fo:repeatable-page-master-alternatives>
        <fo:conditional-page-master-reference
          master-reference="left" odd-or-even="odd" />
        <fo:conditional-page-master-reference
          master-reference="right" odd-or-even="even" />
      </fo:repeatable-page-master-alternatives>
    </fo:page-sequence-master>
  </fo:layout-master-set>
...
</fo:root>
```

`size` is a shorthand for specifying the page size. For a Japanese paperback book with vertical text, this is specified as A6.

Font size and line height for the body text are specified on `<fo:root>`. In Japanese, the use of a quarter-millimeter (*q*) as the unit of font size may make the calculation of the printing area easier. The line spacing is enlarged because ruby and sidenote markers frequently appear between lines in this document.

In order to alternate the margins and position of the headers between left and right pages, two page masters are defined and the page sequence master determines when each is selected.

Sidenote marker

```
...
しかし先月の事だ、<fo:inline-container
  alignment-baseline="central"
  block-progression-dimension="0rem"
  text-indent="0" line-stacking-strategy="font-height"
  start-indent="0" end-indent="0" text-align="center" ...>
  <fo:block
        text-indent="0pt"
    font-size="0.5rem" line-height="1"
    space-before.conditionality="retain"
    space-before="-1rem" >
```

```
    <fo:footnote>
      <fo:inline alignment-baseline="before-edge"
        font-size="0.5rem"> ( <axf:footnote-number xml:id="saiunkak
u" />)
      </fo:inline>...
    </fo:footnote>
  </fo:block>
</fo:inline-container>...
```

Because the sidenote marker takes up the inline area, even if moving the baseline, a space of the width (height) of the sidenote marker will be generated between the preceding and following text. To display the sidenote marker on the side of the body text without spaces, the inline container is used.

Keep sidenote marker within the printing area

```
<!-- Adjust to not protrude sidenote markers from the printing imag
e -->
しかし先月の事だ、<fo:inline-container ...
  axf:inline-overflow-align="end">
  ...
  </fo:inline-container>...
```

The inline container positions the sidenote marker to the side of the body text, but if a line break occurs at the point of the inline container, the inline container could overflow the body region.

`axf:inline-overflow-align="end"` adjusts this so that the sidenote marker fits within the body region at the bottom of the page where the sidenote marker could otherwise overflow due to a line break.

Convert text to different styles

```
<!-- convert text and number format -->
<fo:flow flow-name="xsl-region-body"
    axf:number-transform="kansuji" axf:kansuji-style="readable"
    text-transform="fullsize-kana" font-variant="jis04" >
  ...
</fo:flow>
```

`text-transform="fullsize-kana"` changes small hiragana and katakana in the original text to full size.

`axf:number-transform="kansuji"` specifies to format numbers in the body text as Japanese numerals. `axf:kansuji-style="readable"` specifies to format the Japanese numerals as "〆" and "〇", etc.

Replace specific text

```
<!--  Replace specific text -->
<fo:block ...
 axf:text-replace="はば は&#x309E; それぞれ それ&#x3032; いろいろ いろ&#x3031;"
 >エスペラントの話を聴きたい、よろしい。やりませう。...</fo:block>
```

`axf:text-replace` makes it is possible to replace text that is difficult to input directly from the keyboard, such as hiragana repetition symbols, during formatting without using them in the original text.

Spacing of Japanese punctuation characters

```
<!-- Adjust the spacing of Japanese punctuation characters -->
<fo:wrapper ... axf:punctuation-trim="adjacent">
  <fo:block ...>エスペラントの話...</fo:block>
  </fo:wrapper>
```

Specify `axf:punctuation-trim` to reduce the width of some Japanese punctuation characters. This example specifies `adjacent` to trim adjacent full-width parentheses.

Sidenote rule side

```
<!--  Specify the ruled line of the sidenote -->
<fo:static-content flow-name="xsl-footnote-separator">
  <fo:block text-align="left"
  line-height="1">
    <fo:leader leader-pattern="rule" leader-length="90%"
    leader-alignment="start"
    rule-thickness="0.3pt" rule-style="solid" />
  </fo:block>
</fo:static-content>
```

In this FO, the sidenote is placed on the after side of the odd-numbered page body area. The xsl-footnote-separator region defines the line ruled between the body text and the first sidenote.

When drawing a line with `<fo:leader>`, `leader-length` specifies the line length. 90 % means 90 % of the width of the body text area. `leader-alignment="start"` aligns the end of the leader to the start side.

One-third sized ruby when ruby extends beyond its base

```
<!-- Make ruby that would extend beyond the base character into one
-third sized ruby  -->
...発明されたのも<axf:ruby
  ruby-condense="66.6%"
  ruby-font-size="0.5em"
  ruby-minimum-font-size="0.3rem">
  <axf:ruby-base>畢竟</axf:ruby-base>
  <axf:ruby-text
    font-variant="ruby">ひっきょう</axf:ruby-text>
</axf:ruby>...
```

If you always want narrow ruby, specify `axf:ruby-font-stretch`. If you want a narrow width only when ruby extends beyond its base, specify `axf:ruby-condense`. A one-third sized ruby is one-third the height (width) of the base character, but `axf:ruby-condense` is specified as a proportion of ruby size, so this takes the font size of the ruby into account.

Although it doesn't appear in this FO, when you specify `axf:ruby-condense`, you can use `axf:ruby-overhang` and `axf:ruby-limit-overhang` to control whether and how ruby can overhand its adjacent character.

Section 1.4　Magazine-style with freeform layout

You can combine absolutely positioned block containers and decorations to create a design like a magazine produced with DTP software, as shown in [Figure 1.4-1] (p.32).

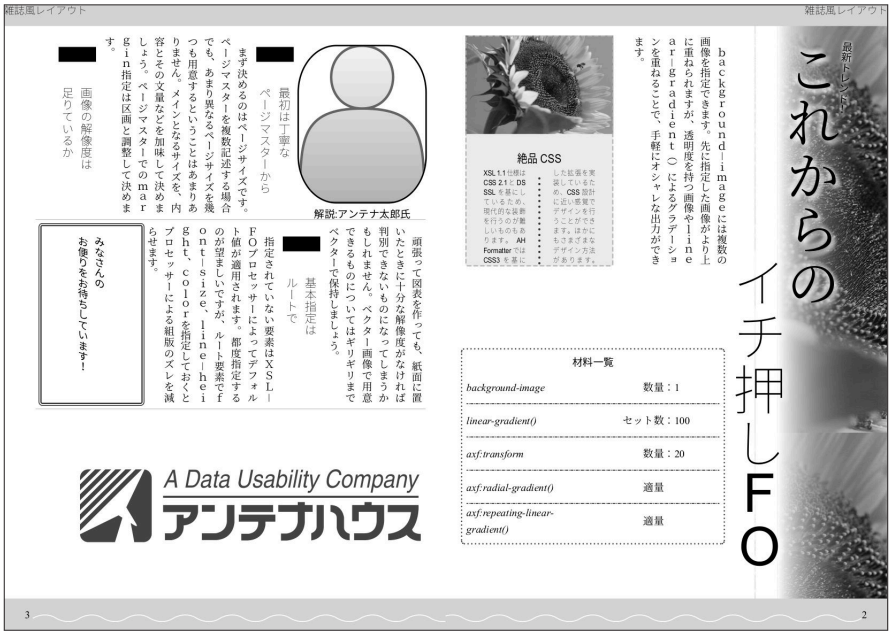


Figure 1.4-1　Magazine-style page design with freeform layout

Background gradient over an image

```
<!-- Specify the background image to the page -->
  size="A4 portrait"
  background-image="linear-gradient(to right,cmyka(0,0,0,0, 1.0)  7
5% ,
    rgba(0,0,0,.0) 100%), url(./background.jpg) ">
  ...
</fo:simple-page-master>
```

`background-image` on the page master specifies the background image for the entire page. `background-image` can specify multiple images, where earlier images are drawn

over later. The gradient from `linear-gradient()` fades from white to transparent. 75 % is completely white, and then the fade to transparent displays the image below.

Different layouts on the right and left

```
<!-- Layout on the left page  -->
<fo:simple-page-master master-name="left"
  size="A4 portrait">
  <fo:region-start extent="10mm" />
  <fo:region-end extent="15mm" />
  <fo:region-body margin-top="15mm"
    margin-bottom="18mm" margin-left="15mm" margin-right="10mm"
    column-count="3" column-gap="6mm" axf:column-rule-style="soli
d"
    axf:column-rule-color="cmyk(0.3,0.3,0,0.0)"
    axf:column-rule-width="0.3pt"/>
</fo:simple-page-master>
```

In this example, the content in the right page is in absolutely positioned block containers, and the content in the left page flows to fill the region.

Fit overflowing text into a defined area

```
<!-- Horizontal writing in the upper left of the right page -->
<fo:block-container  width="14em" height="22rem">
  <fo:block-container height="10em" ...>
    ...
  </fo:block-container>
  ...
  <fo:block-container  overflow="condense"
     axf:overflow-condense="font-stretch"
    height="22rem - 10em"
    axf:overflow-limit-block="22rem - 10em - 16pt - 2pt - 0.5pt"
    ...
    margin="4mm">...

    <fo:block ...>絶品 CSS<fo:block>
    <fo:block ...>XSL 1.1 仕様は CSS 2.1 と DSSSL を基にして...
    </fo:block>
  </fo:block-container>
</fo:block-container>
```

When a block container has specified height and width, the contents of child elements may exceed its size. There are several ways to control this overflow, but in this case, the font width is condensed until the text fits into the area.

Heading style

```
<!-- Decoration of the heading -->
<fo:block-container margin-top="0.5em"
  absolute-position="absolute" top="0mm" right="0"
  height="100vh">
  <fo:wrapper color="cmyk(80, 10,10, 0)"
    font-family="sans-serif">
    <fo:block
      axf:transform="rotate(5)"
      axf:transform-origin="top"
      text-shadow="1pt 1pt 3pt cmyk(0,0,0,0.0),
        -1pt -1pt 3pt cmyk(0,0,0,0)"

      >最新トレンド！</fo:block>
    <fo:block font-size="72pt"
      line-height="1" font-weight="100"
      text-shadow="4pt 4pt 5pt cmyk(0,0,0,0.0),
        -4pt -4pt 5pt cmyk(0,0,0,0)"
      axf:tab-stops="eol"
      axf:text-orientation="upright"
      >
        <fo:inline
          font-family="serif" font-weight="normal"
          >これからの</fo:inline
          ><fo:inline axf:indent-here="-1em"
          /><axf:tab/>
            イチ押し FO
    </fo:block>
  </fo:wrapper>
</fo:block-container>
```

The smaller text is tilted and displayed next to the main heading. `axf:transform` specifies the transform, and `axf:transform-origin` specifies its origin. In this example, the origin is changed to top. `text-shadow` adds emphasis to the heading text. The blur around the characters is the same white color as the background.

Background color under dotted border

```
<!-- Change the range where the background is applied -->
<fo:block-container axf:background-clip="border-box"
  writing-mode="lr-tb"
  axf:border-radius="3pt"
  border="1.5pt dashed cmyk(10, 0.4, 0.4, 0)"
```

```
  background-color="cmyk(0.0, 0.071, 0.298,0.0)"

  absolute-position="absolute"
  width="14em" height="22rem">
</fo:block-container>
```

`axf:backfground-clip="border-box"` extends the block background to below the dotted border line.

Full-width European text in vertical writing

```
<!-- writing-mode="tb-rl"-->
<fo:wrapper ...
  text-transform="fullwidth-if-vertical">
  ...
  <fo:block>background-image には複数の画像を指定できます。...
  </fo:block>
</fo:wrapper>
```

`text-transform="fullwidth-if-vertical"` converts European characters into full-width characters when the text is vertical writing.

Section 1.5 Magazine-style

You can create a page with a combination of multiple columns and column-spanning floats.

The pages shown in [Figure 1.5-1] (p.36) combine column rules, fine control of float placement and column spanning, line grids, drop initials and column settings by block containers.

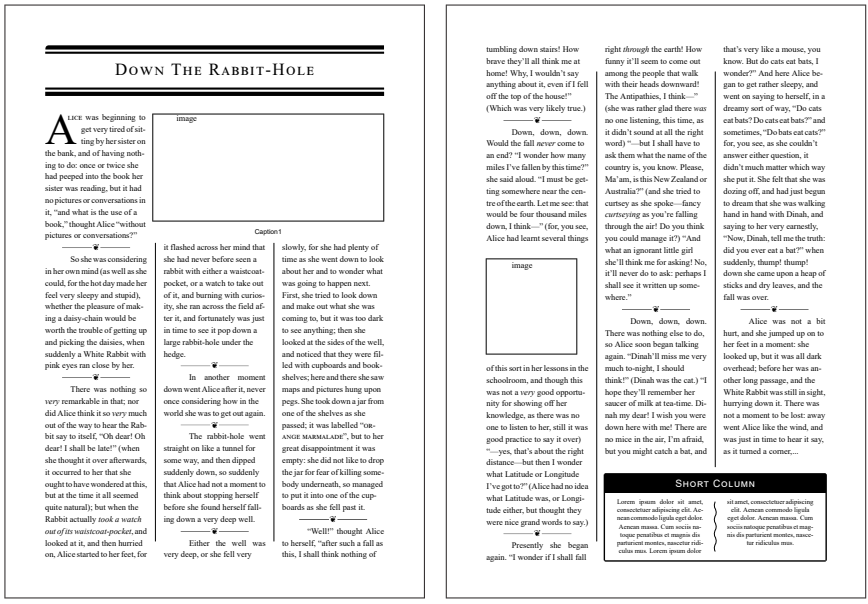


Figure 1.5-1 European magazine-style page design

Columns and column rules

```
<!-- Columns and column rules in the body region - -->
<fo:root xml:lang="en" ... axf:line-break="line">
  ...
  <fo:layout-master-set>
    <fo:simple-page-master ...>
      <fo:region-body ...
          column-count="3" column-gap="8mm"
        axf:column-rule-style="solid"
        axf:column-rule-color="rgb-icc(0, 0, 0, #CMYK,
                                       0.0, 0.0, 0.0, 1.0)"
        axf:column-rule-width="1pt" axf:column-rule-length="90%"
```

```
        axf:column-rule-align="center" axf:column-rule-display="all
"/>
        ...
    </fo:simple-page-master ...>
  </fo:layout-master-set>
  ...
</fo:root>
```

In order to maximize line length while keeping three columns, `column-gap="8mm"` specifies a narrow column-gap, while the column rule specified with `axf:column-rule-*` emphasizes the space between columns. `axf:column-rule-length="90%"` leaves openings at the top and bottom. `axf:column-rule-align="center"` centers the column rule. `axf:column-rule-display="all"` specifies that the column rule is also drawn on columns that have no body text.

Decorative heading

```
<fo:block span="all"
  border-top-style="double" border-bottom-style="double"
  border-color="black" border-width="10pt"
  axf:border-top-left-radius="5pt"
  axf:border-double-thickness="0.5rem 0.25rem 0.25rem""
  space-after="1rem" space-after.conditionality="retain"
  letter-spacing="2pt" font-size="2rem"  dominant-baseline="centra
l"
  font-variant="small-caps"
  text-align="center">Down The Rabbit-Hole</fo:block>
```

The heading spans all of the columns. This example specifies the block decoration, space to the next element and the style of the heading text.

`border-style="double"` specifies a double line border. `axf:border-double-thickness` refines the appearance of the double line border. `axf:border-top-left-radius` specifies a rounded corner only in the upper left.

`space-after` prevents the heading and the body text from being too close together. `font-variant="small-caps"` specifies that the heading string is formatted as small capitals. You may instead specify `titling-caps`, but only if the font has the required glyphs. The text is center aligned, the font size is set to three times the size of the body text and letter spacing is specified as `letter-spacing="1.2pt"`.

Body paragraphs

```
<!-- - Layout of the body paragraphs - -->
<fo:root ...
```

```
  font-size="12pt" line-height="1.4">
  ...
  <fo:page-sequence  ...
    ...
      <fo:flow axf:baseline-grid="root" ...>
      <fo:wrapper text-indent="3rem" text-align="left"
        axf:text-align-first="end"
      axf:avoid-widow-words="true"
      hyphenate="true"
      hyphenation-push-character-count="auto"
            hyphenation-remain-character-count="auto"
      axf:hyphenate-hyphenated-word="false"
        line-stacking-strategy="font-height"
        axf:ligature-mode="latin">
        <fo:block>...</fo:block>
        ...
      </fo:wrapper>
    </fo:flow>
  <fo:page-sequence ...>
</fo:root>
```

The style common to the body paragraphs is defined on the `<fo:wrapper>`. The body text comprises most of the document, the font size and line height of the body text are specified on the root element, and `axf:baseline-grid="root"` on `<fo:flow>` aligns the line grid to that.

The paragraphs are left-aligned by specifying `text-align="left"`. `text-indent="3rem"` specifies the indent of the first line of each paragraph. To emphasize this indent, the first line is justified from after the indent to the end of the line by `axf:text-align-first="justify"`. The `axf:text-align-first="justify"` specified here is inherited, and when there is a single line of text, such as in a child float, it will take precedence over `text-align="center"`, etc.

Because of the short line length, `hyphenate="true"` is specified to allow line breaking in the middle of a word. By specifying `axf:hyphenate-hyphenated-word="false"`, words that originally contain hyphens are not hyphenated. The `hyphenation-push-character-count` and the `hyphenation-remain-character-count` properties specify auto, which is an AH XSL Formatter extension. When `auto`, AH XSL Formatter uses the character count specified in the hyphenation exception dictionary, or two characters if it is unspecified.

`axf:ligature-mode="latin"` specifies that ligatures should be used when wherever possible in European text.

Note

Lines in a multi-column layout are short, so the paragraph-by-paragraph optimization of line breaks specified by `axf:line-break="bpil"` will have limited effect.

Drop capital

```
<!-- Drop capital on the first paragraph - -->
<fo:wrapper text-indent="3rem" ...>
  <fo:block axf:baseline-block-snap="center"
    axf:initial-letters="3 drop"
    text-indent="0rem"

    axf:initial-letters-end-indent="-1rem 0.5rem"
    axf:text-align-first="justify">
    <fo:inline
      font-variant="small-caps">Alice</fo:inline> was beginning to
...
  </fo:block>...
</fo:wrapper>
```

`axf:initial-letters="3 drop"` specifies a three-line drop capital. This is the only paragraph with a drop capital, and it specifies `text-indent="0rem"` to override the `text-indent="3rem"` specified on `<fo:wrapper>`.

Because the space between the "*A*" drop capital and the first line appears wide, the first line is shifted 1 rem to the left, and the spaces from second and subsequent lines are widened by 0.5 rem with `axf:initial-letters-end-indent="-1rem 0.5rem"`. Because this adjustment causes extra white-space at the end of the first line, `axf:text-align-first="justify"` makes it justified.

Decoration between body paragraphs

```
<!-- Insert decorative block between paragraphs -->
<fo:block>...large rabbit-hole under the hedge.
</fo:block>
<fo:block keep-with-previous="always" text-align="center"
  text-indent="0" axf:text-align-first="center"
  font-family="fantasy">
  <fo:leader leader-pattern="rule"
    leader-length="50% - 0.5em - 2em"
        rule-thickness="0.3pt"
        baseline-shift="0.3em"
    rule-style="solid" color="k100"/>
  &#x2766;
  <fo:leader ... />
</fo:block>
<fo:block>In another...
<fo:block>
```

You can put decorative lines between paragraphs. Because this is not normal text, `text-indent` is reset for this `<fo:block>`. `keep-with-previous="always"` ensures that it is always placed at the end of its preceding paragraph and so is not displayed at the top of a page or column.

Note

To suppress a decoration at the top of a page or column, you can specify `axf:suppress-if-first-on-page="true"` and remove `keep-with-previous="always"` on the block.

Float positioned below heading

```
<!--  Adjust float position  -->
<fo:float
  axf:float-x="inside" axf:float-y="before"
  axf:float-wrap="wrap" axf:float-margin-x="0.8rem 0"
  axf:float-margin-y="0.5rlh"
  axf:float-offset-y="6rem + 1.5rlh"
  axf:float-reference="multicol">
  ...
  <fo:block-container
    axf:baseline-block-snap="before" axf:baseline-grid="none"
    ... width="4gr - 0.8rem">
    <fo:block text-indent="0" ...
      text-align="center"
      font-family="sans-serif">Caption1</fo:block>
  </fo:block-container>
</fo:float>
```

Because the heading is at the top of the first page, the first float, which has `axf:float-y="before"`, would ordinarily appear above the heading. The `axf:float-offset-y` shifts the float downwards and aligns it with the top of the body text.

The heading height is 6 rem (border + glyph + pad) and the space after the heading is 1.5 rlh, so this value is specified as `6rem + 1.5rlh`.

`axf:float-reference="multicol"` allows the float to span fewer than all columns. gr is a specific unit for column spans where each column and each column gap is 1 gr. As shown with the block container, gr lengths can be combined with other lengths in calculations, such as `4gr - 0.8rem`.

If `axf:float-wrap="skip"` is specified along with `axf:float-reference="multicol"`, there will be no text adjacent to the float, not even in the columns that the float does not span.

Float placed in body text

```
<!--  Float placed in the column  -->
<fo:float
  axf:float-x="inside" axf:float-y="center"
  axf:float-margin-y="0.7rlh" axf:float-margin-x="6mm"
  axf:float-reference="multicol">
  <fo:block-container width="1gr - 6mm"
    height="8rlh" border="1pt solid" ...>
      ...
  </fo:block-container>
</fo:float>
```

`axf:float-reference="multicol"` is specified because if it was `axf:float-reference="column"`, this float would not be placed in the 'inside' column, even when `axf:float-x="inside"` is specified.

`axf:float-wrap="wrap"` is specified for the same reason as for the float positioned below the heading. `axf:float-margin-x` is specified to avoid text wrapping inside the column beside the float.

Float with different number of columns

```
<!--  Float for the "SHORT COLUMN" -->
<fo:float axf:float-margin-y="0.5rlh"
  axf:baseline-block-snap="before"
  axf:float-x="outside" axf:float-y="bottom"
  axf:float-reference="multicol"
  axf:float-offset-y="3mm">
  <fo:block-container text-indent="0"
    width="3gr" border="2pt solid"
    axf:border-radius="3pt"
        line-height="1"
        line-stacking-strategy="max-height"
        axf:baseline-grid="none">
      <fo:block
      padding-top="0.35rlh" padding-bottom="0.35rlh"
      font-family="sans-serif"
      background-color="cmyk(0, 0, 0, 1.0)"
      color="cmyk(0, 0, 0, 0)"
      text-align="center" font-size="1.2rem">
       <fo:inline letter-spacing="1pt"
          font-variant="small-caps"
          >Short Column</fo:inline>
    </fo:block>
      <fo:block-container margin="3mm">
```

```
      column-count="2" axf:column-gap="5mm"
      axf:column-rule-style="wave"
      axf:column-rule-width="3pt"
      axf:column-rule-length="90%"
      axf:column-rule-align="center"
      axf:column-fill="balance"
          <fo:wrapper font-size="0.8rem" line-height="1.2">
        <fo:block text-indent="0" axf:baseline-grid="new"
          line-stacking-strategy="max-height">...
        </fo:block>
      </fo:wrapper>
    </fo:block-container>
  </fo:block-container>
</fo:float>
```

This specifies a block container with two columns as the child of a float that has `axf:float-y="bottom"`.

Its heading block is defined as a sibling of the two-column block container. The heading could instead be a child `<fo:block-container>` with `span="all"` to make it span both columns. Then block-height effects on `axf:column-rule-length` value.

In order to clearly distinguish the float from the body text, its font size is made smaller and the column rule style is wave. Also, `axf:baseline-grid="new"` specifies a new line grid.

Section 1.6 Academic journal

When you create a page, you control the layout of columns and footnotes and the display position of floats.

See [Figure 1.6-1] (p.43).

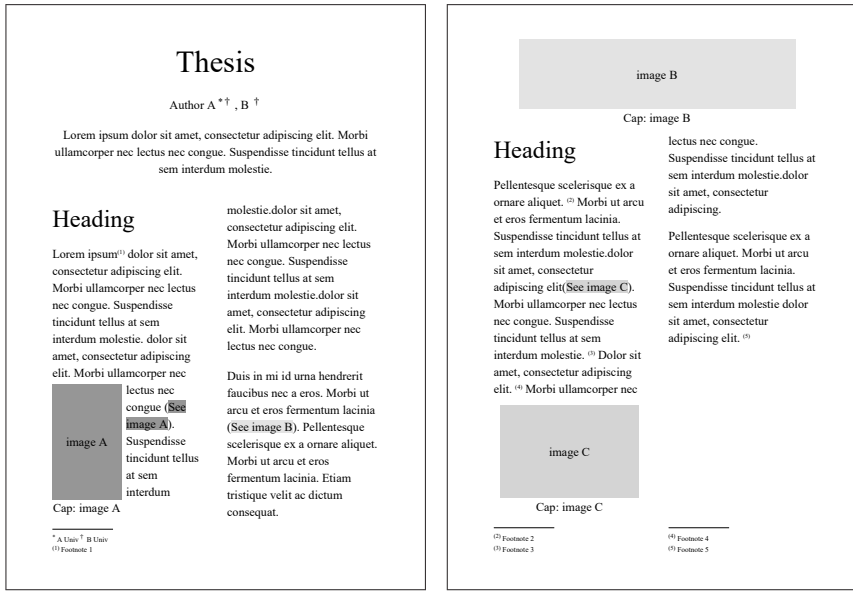


Figure 1.6-1 Academic journal style page design

Footnotes for each column

```
<!-- Footnotes for each column -->
<fo:region-body ...
  column-count="2" column-gap="10mm"
  axf:footnote-position="column" />
  ...
</fo:region-body>
```

Footnotes ordinarily span all columns. `axf:footnote-position="column"` places footnotes in their current column.

Change numbers to symbols

```
<!--  Format numbers as symbols - -->
<fo:declarations>
  <axf:counter-style name="cs-01" system="symbolic" symbols="* †">
</fo:declarations>
...
<fo:page-sequence ...>
  ...
  <fo:block ...>
    ...
    <fo:footnote ...>
      ...
      <fo:footnote-body>
        <fo:block>
        <!-- The cs-01 counter style changes "1" to "*" -->
          <fo:inline baseline-shift="super"
            axf:number-transform="cs-01"
            font-size="0.8em">1</fo:inline>
          ...
        </fo:block>
      </fo:footnote-body>
    </fo:footonote>
  ...
  </fo:block>
  ...
</fo:page-sequence>
```

Some footnotes use a symbol as their marker. Instead of requiring the symbol in the input XSL-FO, AH XSL Formatter transforms the footnote number to a symbol. `<axf:counter-style>` defines a number format conversion rule, and the `axf:number-transform` property on the author's note refers to that format.

Preprocessing by XSLT or other means only needs to count the number. `system="symbolic"` cycles repeatedly through the provided symbols. After the last symbol, the first symbol is doubled up, and so on. As this example uses `symbols="* †"`, the sequence is "*", "†", "**", "††", "***" ....

Title footnotes and body footnotes in same area

```
<!--  Title part  -->
<fo:block ... span="all">
  <fo:block ...>Thesis</fo:block>
  <fo:block ...>Author A
    <fo:inline baseline-shift="super"
```

```
        font-size="0.8em">1</fo:inline>
    Author B
    <fo:inline ...>2</fo:inline>
  </fo:block>
</fo:block>
...
<!-- Body text  -->
<fo:block line-height="0">
...
</fo:block>
<!-- Heading block -->
<fo:block>
...
</fo:block>
```

When the authors' information is in the title part that spans both columns, if their affiliations are present as `<fo:footnote>`, the footnotes will be displayed with the title part.

In order to display the affiliations as footnotes within a column, the footnote symbols are present in the title part just as superscript characters, and the actual footnotes are in a zero-height block at the start of the column-wide body text. These footnotes have an empty element for the footnote marker, and have an author's affiliation in their `<fo:footnote-body>`.

⚠ Notice

Because this structure is just for display, it is recommended when outputting as a tagged PDF to adjust the logical structure using `axf:tag property`, `<fo:basic-link>` to link the footnote-symbol and the footnote, etc.

Inline footnotes

```
<!-- Authors' affiliations footnote block -->
<fo:footnote axf:footnote-stacking="inline">
...
</fo:footnote>
```

Unless specified otherwise, `<fo:footnote-body>` is treated as a block, with each footnote displayed on a separate line. If `axf:footnote-stacking="inline"` is specified, the `<fo:footnote-body>` will generate an inline area. Specify `axf:footnote-stacking` on individual `<fo:footnote>` to mix inline and block footnotes in the same footnote area.

Image at bottom–left of column

```
<!-- eft-aligned float in the column -->
<fo:float axf:float-x="start"
  axf:float-reference="column"
  axf:float-y="after"
  axf:float-move="auto-next"
  ...
  >...</fo:float>
```

`axf:float-x="start"` floats the image to the left side of the current column, and `axf:float-y="after"` floats it to the bottom. Because the image is aligned to one side, the text can wrap around the image.

Centered image without wrapping text when narrow margins

```
<!-- Specify the minimum value of the margin -->
<fo:float ...
  axf:float-wrap="wrap" axf:float-min-wrap-x="4em"
  axf:float-margin-x="0.3em" axf:float-margin-y="0.1em"
  axf:float-centering-x="auto" >
  ...
</fo:float>
```

`axf:float-wrap="wrap"` allows the text to wrap around the image. However, depending on the image size, the wrapped text might be hard to read, and could be as little as one character per line. To avoid this, `axf:float-min-wrap-x` specifies the minimum width for text to be able to wrap. The `axf:float-centering-x="auto"` centers the image when there is not enough width to allow text to wrap. If you know that you want to center the image when text cannot wrap, you can specify a length or percentage for `axf:float-centering-x` instead of specifying `axf:float-min-wrap-x`.

Spanning image in top-center of page

```
<!-- Lay out a large image in the upper-center spanning over column
s -->
<fo:float
  axf:float-reference="page"
  axf:float-y="before" axf:float-x="center"
  axf:float-wrap="skip"
  axf:float-move="next"
  axf:float-margin-y="1em">
```

```
  ...
  <fo:block ...>Cap: Image B</fo:block>
</fo:float>
```

`axf:float-reference="page"` places the large image relative to the page, not to a single column. `center` for `axf:float-x` property and `before` for `axf:float-y` places it in the center and top.

`axf:float-move` is `next` instead of `auto-next` to place the image after its anchor.

`axf:float-wrap="skip"` ensures that the text will not wrap around this image.

Section 1.7 Mathematics reference book

You can write mathematical formulas using MathML 3.0 and format a mathematics reference book, etc. See [Figure 1.7-1] (p.49).

問題 1　　因数分解

次の整式の同類項をまとめなさい。

① $3x - x^2 + 4x + 5 + 2x^2$

② $2x^2 + 3xy + 2x + 2xy + 3y^2 + 2y + 2x + 3y + 2$

回答

① $3x - x^2 + 4x + 5 + 2x^2$

$= (-1 + 2)x^2 + (3 + 4)x + 5$

$= x^2 + 7x + 5$

② $2x^2 + 3xy + 2x + 2xy + 3y^2 + 2y + 2x + 3y + 2$

$= 2x^2 + (3 + 2)xy + 3y^2 + (2 + 2)x + (2 + 3)y + 2$

$= 2x^2 + 5xy + 3y^2 + 4x + 5y + 2$

$2x$ や $-3ay^2$ のように，文字や数の積として表される式を単項式といいます。単項式の和としてあらわされる式を多項式といいます。単項式と多項式を合わせて整式といいます。

ポイント

・整式において，文字の部分が同じ項を同類項といいます。

・同類項をまとめた整式において，各項の次数のうちで最も高いものをその整式の次数といい，次数が m である整式を m 次式 といいます。2 種類以上の文字を含む整式では，特定の文字だけに注目して次数を考えることもあります。注目した文字を含まない項を定数項といいます。

・整式は，特定の文字について，次のように整理することがあります。

降べきの順	多項式を，次数が高いものから順に並べること
昇べきの順	整式の項を次数の低い順に並べること

Figure 1.7-1　Mathematics reference book style page design

Block formula using MathML

```
<!-- Block mathematical formula -->
<fo:block>
  <fo:instream-foreign-object>
    <math xmlns="http://www.w3.org/1998/Math/MathML" display="block
">
      <mn>3</mn><mi>x</mi><mo>-</mo>
      <msup><mrow><mi>x</mi></mrow>
      <mrow><mn>2</mn></mrow></msup>
      ...
    </math>
  </fo:instream-foreign-object>
</fo:block>
```

MathML is included directly as the child of `<fo:instream-foreign-object>`.

If you do not use `display` to specify whether it is a block or inline formula, AH XSL Formatter will use the `mathDisplay` option setting.

When `display` is not specified and `mathmlDisplay` is `context`, AH XSL Formatter will decide based on the surrounding elements, but try to specify `display` whenever possible.

Inline formula using MathML

```
<!-- Inline mathematical formula  -->
<fo:block>
  <fo:inline space-start="0.25em" space-end="0.25em">
    <fo:instream-foreign-object>
      <math xmlns="http://www.w3.org/1998/Math/MathML" display="inl
ine">
        <mn>2</mn><mi>x</mi>
      </math>
    </fo:instream-foreign-object>
  </fo:inline>や<fo:inline>
    <fo:instream-foreign-object>
      <math xmlns="http://www.w3.org/1998/Math/MathML">
      ...
      </math>
    </fo:instream-foreign-object>
  </fo:inline>のように，文字や数の積として表される式を単項式といいます。...
</fo:block>
```

When you include MathML inline in Japanese text, you must generate the spaces between formulas and surrounding text. This example avoids inserting literal spaces,

etc., and instead specifies spaces using the `space-start` and `space-end` properties of `<fo:inline>`.

For an explanation of MathML itself, please refer to the webpage of the W3C Math Working Group and "*An Introduction to MathML Typesetting*" published by Antenna House (Author: Yuji Michihiro. Publisher: Antenna House, Inc. 2019). For details on the option settings related to MathML, refer to the manual.

Overlap heading and border

```
<!-- A block container that specifies only the distance to the prev
ious object - -->
<fo:block-container space-before="10mm">
  <!-- Heading block container -->
  <fo:block-container absolute-position="absolute"
    top="-9pt" left="2.35pt"
    z-index="2">
    inline-progresson-dimension="18pt * 4"
    block-progression-dimension="18pt + 2pt"
    <fo:block font-size="18pt" ... >
    回答
    </fo:block>
  </fo:block-container>
  <!-- 見出し以外の内容を含むブロックコンテナ -->
  <fo:block-container space-before="10pt"
    z-index="1">
    ...
  </fo:block-container>
</fo:block-container>
```

The "*回答*" heading and the content are generated using three block containers.

The outermost block container just specifies the distance from the previous object.

The heading is an absolutely positioned block container, and the borders are on the before and start sides of the block container containing other contents. If the heading block container is a child of the border block container, the borders are drawn over the heading. For the heading to overlap the border, the heading is a sibling with a higher `z-index` value.

Another way to get a similar appearance is for the content block container to not display the top border and to generate the heading plus a leader in a child block. In this way, even if the number of characters in the heading fluctuates a little, you do not have to change the properties of other parts.

```
<!-- - Heading and upper border using fo:leader - -->
<fo:block>
  <fo:inline background-color="gray" ...>回答</fo:inline>
  <fo:inline>
```

```
    <fo:leader leader-pattern="rule"  color="gray"
      rule-thickness="1pt" rule-style="solid" .../>
    </fo:inline>
</fo:block>
<fo:block
  border-start-width="1pt"
  border-start-color="gray"
  border-start-style="solid">
...
</fo:block>
```

Sidenote as float

Because the sidenote only applies to the math formula part, the note is generated as a side float. Because it is desirable to keep the note beside the math formula, specify `axf:float-move="keep"` if there is no room to place the float, the math formula will be moved to the next page.

Note

If you want to place the sidenote in either `<fo:region-start>` or `<fo:region-end>`, the `axf:footnote-position` property for specifying footnote placement is useful.

Section 1.8 Form design

You can design a form by using absolutely positioned block containers that hold tables and PDF interactive forms.

See [Figure 1.8-1] (p.53).

Figure 1.8-1 Form displayed in Acrobat Reader

Absolutely positioned block container

```
<!-- Place item in a fixed position -->
<fo:block-container absolute-position="absolute"
  top="0mm" left="90mm" width="80mm" height="10mm"
  ... >
  <fo:block text-align="right" ...>
    <fo:inline>...年...月...日...<fo:inline>
  </fo:block>
</fo:block-container>
```

Specify the vertical and horizontal offsets for a block container that has `absolute-position="absolute"`. The origin is the current reference area, not the edge of the page.

Text format

```
<!-- unit price input field - -->
<axf:form-field axf:field-name="Tanka001"
  axf:field-type="text"  axf:field-default-text=""
  axf:field-multiline="false"
  axf:field-scroll="false"
  axf:field-maxlen="5"
  ...
  axf:field-text-align="right"
  axf:field-format="0 0 0 ' ' true"
  axf:field-format-category="number"/>
```

Input fields for numerical values are defined using text fields. Specifying the format makes it easier to process the input as a numerical value because you can avoid mixing formats such as "*百円*" (100 yen in kanji) and "*100*".

`axf:field-format-category="number"` limits the input value to a numerical value. `axf:field-format="0 0 0 ' ' true"` indicates: no decimal point display; separate every three digits with ',' ; no special formatting for negative numbers; currency symbol is blank; currency symbol as a prefix. Since it is not necessary to consider decimal point alignment, right alignment is specified with `axf:field text-align="right"`.

Only the display format is specified here. Rounding off the display of the fractional part and not displaying the currency symbol does not affect the calculation of values.

Automatically display subtotals and totals

```
<!-- Quantity -->
<axf:form-field  axf:field-name="Count001"
  axf:field-type="text"
  axf:field-default-text=""
  axf:field-maxlen="4"
  axf:field-text-align="right"
  axf:field-format="0 1 0 ' ' true"
  axf:field-format-category="number"
  width="2em" height="1.2em" ...>
  <axf:form-field-event name="OnBlur" axf:action-type="javascript">
    g=11;
    n = this.getField("Count001").value;
    t = this.getField("Tanka001").value;
    if( n != "" &amp;&amp; t != ""){
    this.getField("syoukei1").value = n * t;
    var t=0;
    for ( i = 1; i &lt; g; i++) {
```

```
      var fieldName="syoukei"+i;
      k=this.getField(fieldName).value;
      if( k != ""){
        t+=k;
       }
     }
    this.getField("goukei").value = t;
   }
  </axf:form-field-event>
</axf:form-field>
```

PDF supports a subset of JavaScript. You can specify `action-type="javascript"` on `<axf:form-field-event>` and include the JavaScript as the element content. Here, the subtotal is calculated and the total is displayed.

Because the JavaScript is defined as an event in the 'quantity' text field, it will not work unless the quantity is entered. The value of `name` on `<axf:form-field-event>` determines the event timing. `OnBlur` specifies that the JavaScript executes when this field, `field-name="Count001"`, loses focus, such as when another item is clicked. Some JavaScript syntax is written as entities for inclusion in the FO.

`g=11` is set because there are 11 items that can be input.

Calculation result as read-only field

```
<!--  Subtotal -->
<axf:form-field
axf:field-type="text" axf:field-name="syoukei1"
  axf:field-format-category="number"
  axf:field-format="0 0 0 ' ' true"
  ...
  axf:field-maxlen="7"
  axf:field-readonly="true"/>
```

Subtotals and total amounts are values obtained from the quantity and price inputs. Fields for displaying the calculation result should not be edited directly in the PDF viewer. Specify `axf:field-readonly="true"` or `axf:field-flags="ReadOnly"` to make a read-only field.

Lines in multi-line text input field

`axf:field-multiline="true"` specifies a `multi-line` text field. However, the lines of the multi-line text field do not count as XSL-FO lines, so trying to underline the lines will only draw an underline on the end of the field.

```
<!-- Background image for multi-line text field -->
<axf:form-field axf:field-type="text"
  axf:field-name="bikou"
  axf:field-multiline="true"
  height="30mm" width="80vw"
  background-image="repeating-linear-gradient(to bottom,
                            white 0mm, white 4.8mm,
                            black 4.8mm, black 5.0mm)" />
```

A background image can be specified for `<axf:form-field>`. `repeating-linear-gradient()` is used to repeat a black color that resembles a ruled line sized to match the font size and spacing of text entered in the field.

Digital signature field to lock the document

```
<!--  Digital signature field -->
<axf:form-field axf:field-name="sig"
  axf:field-type="signature"
  axf:field-lock-document="true" .../>
```

For a digital signature field with `axf:field-lock-document="true"`, signing the field in the output PDF will lock the PDF from further editing.

Chapter 2

AH XSL Formatter Extensions

Section 2.1 Unit

A variety of units can be specified in addition to the units available in the XSL 1.1 specification.

2.1.1 Use the absolute value of the length

Units used in formatting, such as q (quarter) can be described as they are.

Unit Name	Definition	Equivalent length
jpt	Defined by the JIS X 8305.	$1\,\mathrm{jpt} = 0.3514\,\mathrm{mm}$
q	Defined by the JIS X 4052 and the JIS Z 8125. Quarter.	$1\mathrm{q} = 0.25\,\mathrm{mm}$
emu	English Metric Unit.	$1\,\mathrm{emu} = \frac{1}{914400}\,\mathrm{in} = \frac{1}{360000}\,\mathrm{cm}$

2.1.2 Relative Units

The types of relative units that can be used as units of length have also been extended.

Unit Name	Definition	Fallback
lh	Line height as 1.	-
rem	Root element em as 1.	-
rlh	Root element lh as 1.	-
vw	Viewport width as 100.	-
vh	Viewport height as 100.	-
pvw	Page width as 100.	-
pvh	Page height as 100.	-
pvmax	Equal to the larger length of pvw or pvh.	-
pvmin	Equal to the smaller length of pvw or pvh.	-
cap	cap-height of the font as 1.	altitude of the font.
ex	x-height of the font as 1.	`0.5em`

Unit Name	Definition	Fallback
ch	Width of the DIGIT ZERO “0” (U+0030) as 1.	`0.5em`
ic	Width of the “水” (U+6C34) as 1.	`1em`
dcem	Font size of the dropped initial as 1.	`1em`
gr	Unit to specify spanning columns.	-

If the <*length*> is set to `1lh` in places where `font-size="12pt"` and `line-height="1.2"`, it will be internally converted to 14.4 pt and processed.

Since it is based on the value specified in other places, the places that can be specified are limited. For example, the unit “lh” based on the line height cannot be used to specify the line-height property. The fallback value is used for where it cannot be specified or when the value of the reference cannot be determined.

“gr” is a special unit that counts column and column gap as 1 in columns. It can be specified only in columns and in the child elements of the float.

Specify the viewport basis

The viewport is the current area that can be seen on a fixed screen such as paper (without considering scrolling in the screen) as “vw” or “vh”. Interpretation is different betweenfor XSL and CSS3. Determine the basis in the option settings.

Option Property	`viewport-length-units-mode`
Value	`5` | `6`
Initial	`6`
Applies to	`<formatter-settings>`

Control the “vw” and “vh” basis by the `viewport-length-units-mode` property in the option setting.

When it is `5`, the page size is the basis, and when it is `6`, the area size excluding the page margin is the basis.

When `6` is specified, “pvw” and “pvh” are available instead in units where the area including the page margin is 100.

2.1.3 Specify “0” without unit

Only when the length is 0, “no units” is accepted.

The XSL 1.1 specification always requires units for length values. AH XSL Formatter accepts “no units” like `text-indent="0"` only when it is 0. It is incorrect for unitless 0 to appear in a formula calculating the length.

2.1.4 Use dpi to specify the resolution

There are properties that can be specified by dpi.

Unit Name	Definition
<*dpi*>	Represents the resolution. Describe as the numerical value with `dpi` added.

In the `axf:image-resolution` property etc., the resolution can be specified with dpi. Specify as `axf:image-resolusion="300dpi"`.

2.1.5 px per inch

A length in “px” must be converted to an absolute length. The number of pixels per inch can be specified as an option setting.

Option Property	`pxpi`
Value	<*number*>
Initial	`96`
Applies to	`<formatter-settings>`

`pxpi` is a positive real number that specifies the conversion factor. Images that have vertical and horizontal lengths in pixels and no resolution are also converted based on the value of the `pxpi` property.

2.1.6 Specify the page size in paper size

When setting the page size, you can specify the standardized paper size by its name.

The XSL 1.1 specification allows you to specify the width and height of page by length. However, in many cases it will be specified as a standard size for printing, such as 210×297mm. AH XSL Formatter extends the value that can be specified for `size`, and the paper size (paper name) can be specified for the `size` property. It is not case sensitive.

size	Horizontal × Vertical	Paper size
A3	297 × 420 mm	ISO 216
A4	210 × 297 mm	ISO 216
A5	148 × 210 mm	ISO 216
A6	105 × 148 mm	ISO 216
B4	250 × 353 mm	ISO 216
ISO-B4	250 × 353 mm	ISO 216
JIS-B4	257 × 364 mm	JIS P 0138
B5	176 × 250 mm	ISO 216
ISO-B5	176 × 250 mm	ISO 216
JIS-B5	182 × 257 mm	JIS P 0138
B6	125 × 176 mm	ISO 216
ISO-B6	125 × 176 mm	ISO 216
JIS-B6	128 × 182 mm	JIS P 0138
Letter	8.5 × 11 in	North American Paper Sizes
Government-Letter	8 × 10.5 in	North american Paper Sizes
Legal	8.5 × 14 in	North American Paper Sizes
Ledger	17 × 11 in	North American Paper Sizes
Tabloid	11 × 17 in	North American Paper Sizes
Statement	5.5 × 8.5 in	-
Executive	7.25 × 10.5 in	-
Folio	8.5×11in	-
C	17 × 22 in	ANSI Paper Sizes
D	22 × 34 in	ANSI Paper Sizes
E	34 × 44 in	ANSI Paper Sizes
ISO-DL	110 × 220 mm	ISO 269
ISO-C3	324 × 458 mm	ISO 269

size	Horizontal × Vertical	Paper size
ISO-C4	229 × 324 mm	ISO 269
ISO-C5	162 × 229 mm	ISO 269
ISO-C6	114 × 162 mm	ISO 269
Hagaki	100 × 148 mm	Japan Post Service

For these settings, `portrait` and `landscape` can be added to specify which is the width or height. If not specified or in case of the portrait, the long side will be the height. In case of the `landscape`, the short side will be the height.

```
<!-- Specify the A4 lansdscape -->
<fo:simple-page-master master-name="a4page" size="A4 landscape" >
  ...
</fo:simple-page-master>
```

2.1.7 Examples

Layout based on multiples of line height

rlh is the line height specified for the `<fo:root>` element. By specifying the body text size on the `<fo:root>` element, you can easily specify, for example, a heading block the same size as n lines of body text, and so on. A two-line height in Japanese is specified as `1rlh + 1rem`.

Text block based on the number of characters per line

For example, in an A6 portrait page (width 105 mm, height 148 mm) with vertical text, it is possible to specify a body region that is 40 characters high and 20 lines wide and with symmetric margins:

```
<!--  Specify on the basis of the height of the line -->
<fo:region-body
  margin-top="74mm - 20rem" margin-bottom="74mm - 20rem"
  margin-left="52.5mm - 10rlh" margin-right="52.5mm - 10rlh"/>
```

The actual number of characters can be rendered is affected by other settings, such as the `line-stacking-strategy`.

Section 2.2 Page control

You can control the order and size of pages.

2.2.1 Element across two pages in a spread

To generate different layouts on left and right pages using just the features of XSL 1.1, you need to define two simple page masters and to use them in the page sequence master as the masters for even and odd pages.

This works for facing pages with different layouts on the right and left pages, but it does not handle elements that are placed across both the left and right pages of the spread. If a float or an absolutely positioned block container is placed across the pages, the rest is not displayed on the opposite page [Figure 2.2-1] (p.65).

Element	`<axf:spread-page-master>`
Property	`axf:left-page-master-reference`, `axf:right-page-master-reference`, `reference-orientation`, `wrirting-mode`
Parent	fo:layout-master-set

Element	`<axf:spread-region>`
Property	`region-name`, `height`, `overflow`, `reference-orienatation`, `width`, `writing-mode`
Parent	axf:spread-page-master

Property	`axf:left-page-master-reference`, `axf:right-page-master-reference`
Value	*<name>*
Initial	None (Value is required.)
Applies to	axf:spread-page-master

You can use `<axf:spread-page-master>` to create a page master for a two-page spread. `axf:left-page-master-reference` specifies the simple page master for the left page and `axf:right-page-master-reference` specifies the right page master.

Use `<axf:spread-region>` to define regions that can span across the left and right pages. Specify the position and size of each region similarly to the body region. The default size of a spread region is the combined size of the left and right pages. `margin-left` of a

<axf:spread-region> is the distance from the left edge of the left page, and margin-right is the distance from the right edge of the right page.

A spread region cannot have <fo:footnote> as descendant. The spread region takes precedence over the body region flow where they overlap.

In practice, <axf:spread-page-master> is frequently used with a page sequence master that also uses left-only or right-only page masters.

```
<!--  Combining the two-page spread with the normal page -->
<fo:layout-master-set>
  <fo:simple-page-master master-name="left" ... >
  ...
  </fo:simple-page-master>
  <fo:simple-page-master master-name="right" ... >
  ...
  </fo:simple-page-master>
  <axf:spread-page-master master-name="spread"
    left-page-master-refeence="left"... >
    right-page-master-reference="right"
    <axf:spread-region region-name="spreadRegion" .../>
  </axf:simple-page-master>
  <!-- Page sequence master that specifies the appearance conditio
n of the page
                master -->
  <fo:page-sequence-master master-name="main" >
    <fo:repeatable-page-sequence-master-alternatives>
      <fo:conditional-page-sequence-master-reference
        master-reference="right" page-position="first"/>
      <fo:conditional-page-sequence-master-reference
        master-reference="spread" page-position="rest"/>
      <fo:conditional-page-sequence-master-reference
          master-reference="left" page-position="last"/>
    </fo:repeatable-page-sequence-master-alternatives>
  </fo:page-sequence-master>
</fo:layout-master-set>
...
<page-sequence master-reference="main"
  force-page-count="end-on-odd" ...>
  <!-- Region to display two- page spread element -->
  <fo:flow flow-name="spreadRegion">
    <fo:block-container ...>
        ...
    </fo:block-container>
  </fo:flow>
  <!-- Normal body region -->
  <flow flow-name="xsl-region-body">
  ...
  </flow>
</page-sequence>
```

```
<page-sequence master-reference="spread">
  <flow flow-name="xsl-region-body">
  ...
  </flow>
</page-sequence>
```

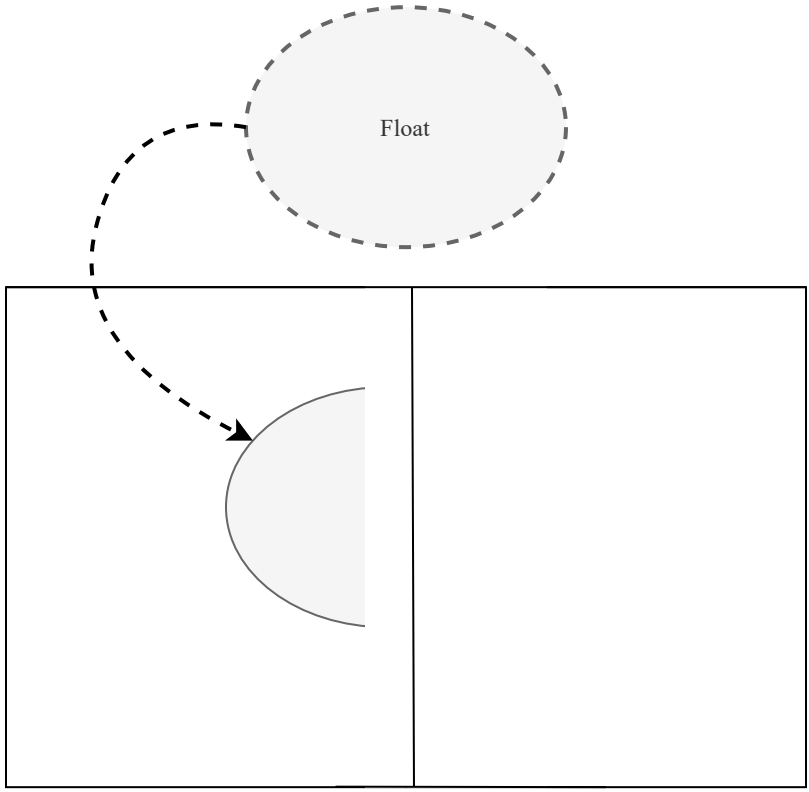


Figure 2.2-1 Object cannot cross two pages

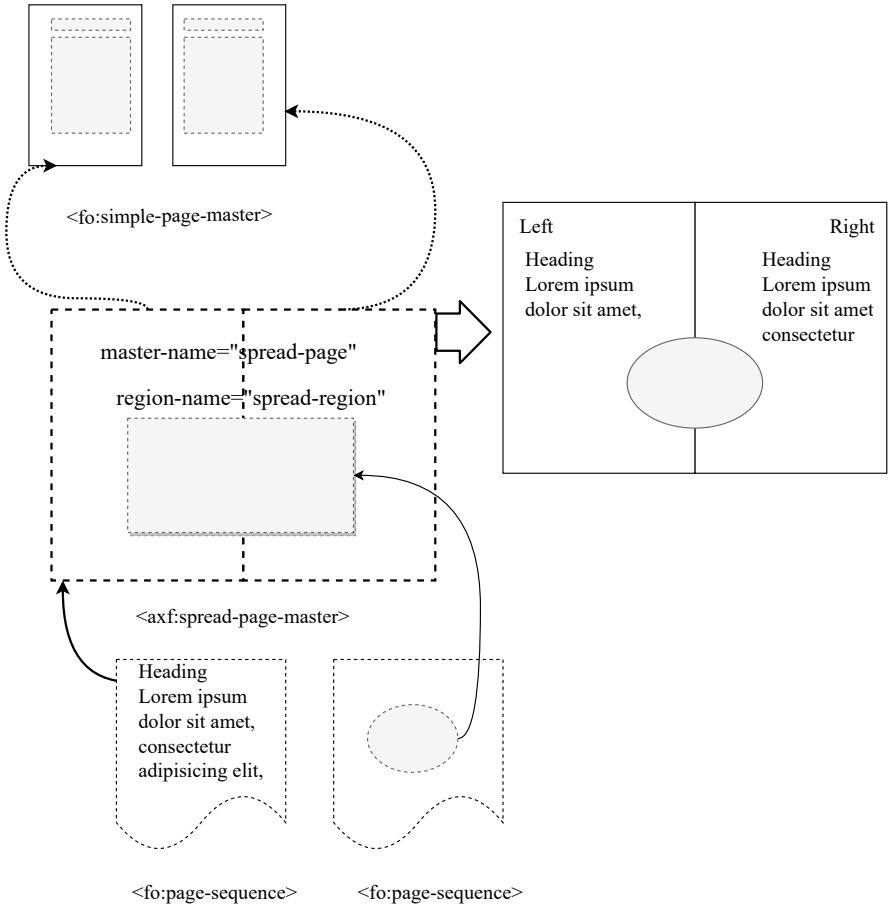


Figure 2.2-2 Two-page spread from `<axf:spread-page-master>` and spread region from `<axf:spread-region>`

2.2.2 Nested page sequences

With XSL 1.1, a temporary switch in the page size requires you to end the current page sequence, start and end a page sequence that uses a different page master with the different page size, then start a new page sequence that resumes using the same page master as the first page sequence. It can sometimes be difficult to specify all of this in your XSL-FO.

```
<!-- End and switch the page sequence -->
<fo:layout-master-set>
  <fo:simple-page-master size="A4 portrait" master-name="main">
    ...
  </fo:simple-page-master>
  <fo:simple-page-master size="A3 landscape" master-name="illust">
    ...
  </fo:simple-page-master>
</fo:layout-master-set>
  ...
<fo:page-sequence master-reference="main">
  ...
</fo:page-sequence>
<fo:page-sequence master-reference="illust">...
</fo:page-sequence>
<fo:page-sequence master-reference="main">...
<fo:page-sequence
```

In AH XSL Formatter, `<fo:page-sequence>` has been extended so that it can be nested.

```
<!-- Nesting page sequences -->
<fo:page-sequence master-reference="main">
  ...
  <fo:page-sequence master-reference="illust">...
  </fo:page-sequence>
  ...
</fo:page-sequence>
```

This is useful, for example, when you want to change the page size and insert a large figure or table.

2.2.3 Page count control

AH XSL Formatter extends the `force-page-count` property compared to XSL 1.1 to give you more control over the number of pages in the page sequence than simply requiring an odd or even number of pages.

Property	`force-page-count`
Value	`auto` | `even` | `odd` | **`doubly-even`** | `end-on-even` | `end-on-odd` | **`end-on-doubly-even`** | **`even-document`** | **`odd-document`** | **`doubly-even-document`** | [**`end-on`** | **`document`**]] ? <***number***> <***number***>] ? | `no-force`
Initial	`auto`
Applies to	fo:page-sequence

force-page-count	Definition
doubly-even	The page sequence will be a multiple of 4 pages.
end-on-doubly-even	The page number of the last page of the page sequence will be a multiple of four.
even-document	Counting from the beginning of the document, the page number of the last page of the page sequence will be an even number.
odd-document	Counting from the beginning of the document, the page number of the last page of the page sequence will be an odd number.
doubly-even-document	Counting from the beginning of the document, the page number of the last page of the page sequence will be a multiple of four.
[**end-on** | **document**]] ? <*number*> <*number*>] ?	The number of pages will be a multiple of the first <*number*> plus the second <*number*>, if present.

`doubly-even` and `end-on-doubly-even` constrain the current page sequence based on multiples of four. They are extensions beyond the standard `even` and `end-on-even`.

`*-document` constrain the number of pages from the beginning of the document. They are useful when combining multiple page sequences.

When the common constraints for a number of pages that is odd, even, or a multiple of four are not sufficient, the page count can be specified as a multiple of <*number*> plus a second <*number*>. An initial `end-on` constrains the number of pages within the page sequence, or if document is added, it constrains the number of pages from the beginning of the document.

Any added pages are output as blank pages.

2.2.4 Insert arbitrary empty page

In AH XSL Formatter, empty `<fo:page-sequence>`, one that omits any `<fo:flow>` children, is allowed.

This is useful for adjusting the page count by adding blank pages.

However, when `force-page-count="no-force"` is specified, `<fo:flow>` must be present.

2.2.5 Reverse page order

You can output the pages in a page sequence in reverse order.

Property	axf:reverse-page
Value	true | false
Initial	false
Applies to	fo:page-sequence

The pages in an <fo:page-sequence> with axf:reverse-page="true" are output in reverse order.

The page numbering remains the same. For example, in a twenty-page document, the first output page will be page 20 with page number "20".

Only PDF output is supported. Also, this is not supported in multiple-volume output.

2.2.6 Example: Change the page order of PDF output

In AH XSL Formatter, a specific page of a PDF can be embedded using either <fo:external-graphic> or background-image.

It is possible to embed a PDF in a new FO and output it again in the page order for optimal printing.

In this example, an 8-page A4 PDF booklet is embedded in A3 size and output as a set of "1-8", "2-7", "3-6", and "4-5".

```
<!--  Embedding PDF in page master -->
<fo:layout-master-set>
  <fo:simple-page-master master-name="embed-1"
    background-image="embedded.pdf#page=1"
    background-repeat="no-repeat" >...</fo:simple-page-master>
  <fo:simple-page-master master-name="embed-2"
    background-image="embedded.pdf#page=8"
    ... >...</fo:simple-page-master>
  <fo:simple-page-master master-name="embed-3"
    background-image="embedded.pdf#page=2"
    ... >...</fo:simple-page-master>
  <fo:simple-page-master master-name="embed-4"
    background-image="embedded.pdf#page=7"
    ... >...</fo:simple-page-master>
  <fo:simple-page-master master-name="embed-5"
    background-image="embedded.pdf#page=3"
    ... >...</fo:simple-page-master>
  <fo:simple-page-master master-name="embed-6"
    background-image="embedded.pdf#page=7"
    ... >...</fo:simple-page-master>
  <fo:simple-page-master master-name="embed-7"
    background-image="embedded.pdf#page=4"... >
    ...</fo:simple-page-master>
  <fo:simple-page-master master-name="embed-8"
    background-image="embedded.pdf#page=5"... >
    ...</fo:simple-page-master>
  <fo:page-sequence-master master-name="main" ...>
    <fo:single-page-master-reference master-reference="embed-1" />
    <fo:single-page-master-reference master-reference="embed-2" />
    <fo:single-page-master-reference master-reference="embed-3" />
    <fo:single-page-master-reference master-reference="embed-4" />
    <fo:single-page-master-reference master-reference="embed-5" />
```

```
    <fo:single-page-master-reference master-reference="embed-6" />
    <fo:single-page-master-reference master-reference="embed-7" />
    <fo:single-page-master-reference master-reference="embed-8" />
  </fo:page-sequence-master>
</fo:layout-master-set>
    ...
<fo:page-sequence master-reference="main">
      <fo:flow flow-name="xsl-region-body"></fo:flow>
</fo:page-sequence>
```

2.2.7 Example: Images and tables with different page sizes

It is simple to include pages with different sizes in the middle of the formatting.

In [Figure 2.2-3] (p.71), the first and third pages and the second and forth pages have the same page size.

Page for images and tables using nested page sequences

```
<!-- - Page masters with different sizes -->
<fo:simple-page-master master-name="NormalPage"
  page-height="257mm" page-width="182mm">...
</fo:simple-page-master>
<fo:simple-page-master master-name="BigPage"
  page-height="257mm" page-width="364mm">...
</fo:simple-page-master>
...
<!-- Nested page sequence -->
<fo:page-sequence master-reference="NormalPage">
...
  <fo:block ...>出典:...</fo:block>
    <fo:page-sequence master-reference="BigPage">
    ...<fo:external-graphic ...>
    </fo:page-sequence>
  <fo:block ...>...</fo:block>
  ...
</fo:page-sequence>
```

The page sequence that refers to *BigPage* is included as a descendant, without terminating the page sequence that refers to *NormalPage*.

Note

Mixing pages of different sizes is not a problem when outputting on digital media such as PDF. But for printing purposes, the printed paper will use a blank page on the back because two pages are printed as one sheet of paper, even if only one page is inserted in the middle. When multiple pages of different sizes are included, it is better to collect the parts that appear in the document in advance.

DITA 入門 しくみと概念

DITA とは、ドキュメント記述・管理のための「しくみ」です。この「しくみ」の目的と、基本的な概念を紹介します。

DITA（Darwin Information Typing Architecture）は元は IBM が策定した、「技術文書」（テクニカルドキュメント）を記述・管理するための仕様[1]で、その後、この仕様は構造化情報の標準化団体である OASIS（Organization for the Advancement of Structured Information Standards）に寄贈され世界標準の仕様となりました。

DITA の目的は次の 3 つです。「データ再利用性の向上」「ワンソースマルチユース」「翻訳の効率化」これらはどれも絡み合った話ですので、あえて 3 つに分けることもないかと思いますが、DITA はこれらを実現するために考えられた仕様です。

よって、DITA は次の特徴を持つ文書、または文書の一部に対し効果的です。

「複数の箇所で共通する説明がある」

「複数の製品で共通する説明がある」

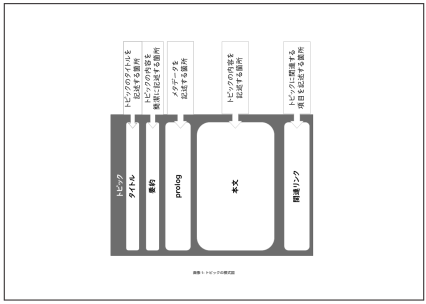
「多言語で用意する必要がある」

「様々な対象向けに少しずつ異なった出力を用意する必要がある」

「様々な媒体向けに出力を変える必要がある」

「微細な変更が広範に、頻繁に行われる」

Page 1

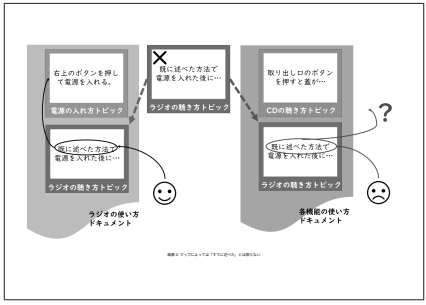


Page 2

DITA は XML（Extensible Markup Language）文書として記述します。 XML は、HTML のように文書にタグ付けをする言語ですが、HTML との大きな違いとして「文書の決まりを自分で作れる」という特徴があります。作った決まりをみんなで共有し、使えるようにすることを「標準化」といいます。つまり、DITA は実際の文書としては「DITA という決まりに従った XML で書かれたファイル群」となります。

DITA ではひとつのトピックファイルにはひとつのトピック（話題）しか書かない、ということが推奨されています。ひとつのトピックファイルの中でだらだらといくつもの話題に触れるのはよしましょう、ということですね。そのため、ひとつのドキュメントを制作するにあたり必然的に複数のトピックファイルを用意することになります。

Page 3



Page 4

Figure 2.2-3　Pages sized for image and tables inserted in the middle

Section 2.3 Styling formatting objects

AH XSL Formatter has multiple extensions for styling formatting objects.

Stylistic extensions include: borders for both blocks and inline areas; column rules; and change bars. In addition, the types of borders that can be specified for leaders have been extended.

2.3.1 Border styles

The border style has been extended with `dot-dash`, `dot-dot-dash` and `wave` which were defined in an old CSS3 Border Working Draft.

<border-style>	**Priority when multiple settings are specified**
`double`	1
`solid`	2
`dashed`	3
`dot-dash`	4
`dot-dot-dash`	5
`dotted`	6
`ridge`	7
`outset`	8
`groove`	9
`inset`	10
`wave`	11

Border style priority: Priority of border styles is defined by the system. This order is used when there is no other setting specifying which border to be prioritized when borders overlap.

2.3.2 Wave border shape

You can control both the length and width of the wave border style.

Property	axf:border-wave-form
Value	<*value*>[<*value*>]
Initial	auto
Applies to	all elements
Inherited	yes

axf:border-wave-form specifies the shape of a wave border. Each <*value*> is a <*length*> or <*number*> or auto. The first one sets the length of the wave, and the second one sets the width of the wave border. auto is equivalent to 6 and 0.125. <*length*> is a literal value, and <*number*> is proportional to the border width.

[Figure 2.3-1] (p.74) shows the effect of setting axf:border-wave-form when border-width="3pt".

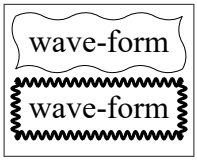


Figure 2.3-1 Default ("6 0.125") and "3pt 0.4"

2.3.3 Double border style

You can control the outer line width, line–line spacing and inner line width of a double border.

When border-style="double", the border is drawn as a double line. AH XSL Formatter extends the XSL 1.1 specification to allow control of the width of the outer and inner lines of the border.

Property	axf:border-double-thickness
Value	<*value*>[<*value*>[<*value*>]]
Initial	1
Applies to	all elements
Inherited	yes

The axf:border-double-thickness property allows you to specify the width of each double line.

The outer line width, line–line spacing and inner line width are specified in sequence. Each <*value*> is a <*number*> or <*length*>. Any omitted value is considered equal to the

previous value. The combined width of the outer line, line–line spacing and inner line should equal the value specified for the `border-width` property.

2.3.4 Rounded corners

You can generate borders with rounded corners.

Because every area is a rectangle, the corners of every border or background color are right angles. AH XSL Formatter supports rounded corners using properties based on [CSS3-Background] Curve Radii: the 'border-radius' properties.

Property	`axf:border-radius`, `axf:border-top-left-radius`, `axf:border-top-right-radius`, `axf:border-bottom-left-radius`, `axf:border-bottom-right-radius`
Value	[<*length*> | <*percentage*>][<*length*> | <*percentage*>]] ?
Initial	`0`
Applies to	all elements which can have borders
Percentages	border box

`axf:border-radius` is a shorthand that applies to all corners. The other `axf:border-*-radius` properties specify the curve at one corner.

Assuming a circle tangential to the border, the first value is the radius in the start–end direction and the second value is the radius in the before–after direction [Figure 2.3-2] (p.76). If only one value is specified, that value is used for both.

The border radius affects box shadows as well as borders.

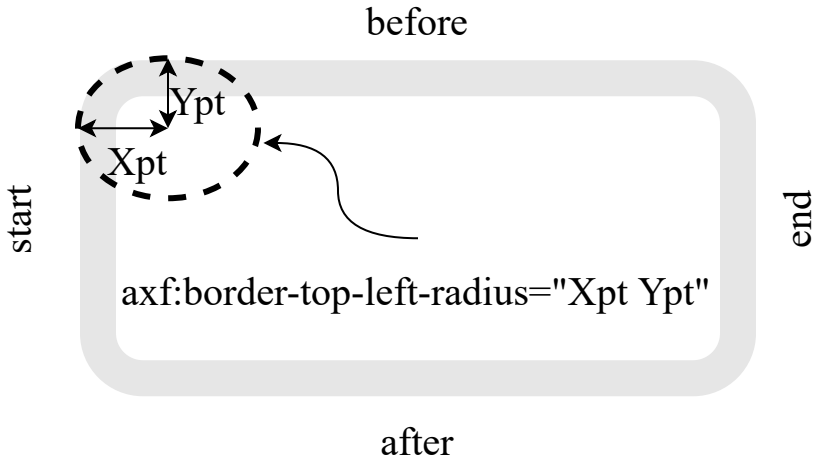


Figure 2.3-2 Setting of the rounded corner

Note

This is not applied when the border style is `wave`.

2.3.5 Border connection shape

Specifies the shape of where the border lines intersect and connect.

Property	`axf:border-connection-form`
Value	`wedge` | `mixed` | `precedence`
Initial	`mixed`
Applies to	all elements
Inherited	yes

`axf:border-connection-form` applies when borders meet at the four corners of an area (upper right, lower right, upper left and lower left corners), and also where intersecting borders make corners, such as in a table.

When `mixed` is specified, a T-shaped intersection is rectangular and other intersections is wedge-shaped. `wedge` connections are X-shaped where borders cross and Y-shaped at T-shaped intersections. By specifying the `precedence`, when the styles of adjacent edges are different, the one with the higher priority of the border is prioritized and drawn to the end.

[Figure 2.3-3] (p.77) shows an example.

In the table with `border-collapse="collapse"` specified, the value of the `axf:border-connection-form` property in `<fo:table>` is adopted and the value specified by `<fo:table-row>` or `<fo:table-cell>` is invalid.

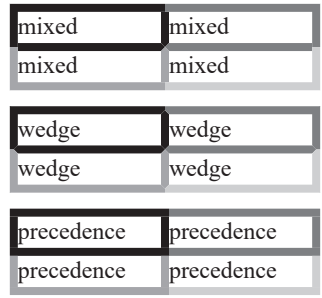


Figure 2.3-3 `axf:border-connection-form` on `<fo:table>`

2.3.6 Diagonal border

You can generate diagonal borders through both block and inline areas.

Property	`axf:diagonal-border-color`, `axf:reverse-diagonal-border-color`
Value	*<color>*
Initial	the value of the 'color' property
Applies to	all elements which can have borders
Inherited	yes

Property	`axf:diagonal-border-style`, `axf:reverse-diagonal-border-style`
Value	*<border-style>*
Initial	`none`
Applies to	all elements which can have borders

Property	axf:diagonal-border-width, axf:reverse-diagonal-border-width
Value	*<border-width>*
Initial	medium
Applies to	all elements which can have borders
Inherited	yes

axf:diagonal-border-* and axf:reverse-diagonal-border-* allow you to draw diagonal borders from corner to opposite corner of an area. This is useful for drawing a diagonal border when you do not know how big the area will be or on things that can be split, such as table cells.

axf:diagonal-border-* apply to a border drawn from the before-start corner to the after-end corner, and axf:reverse-diagonal-border-* apply to a border drawn from the before-end corner to the after-start corner.

2.3.7 Region border

You can specify borders on the body region, etc.

XSL 1.1 defines the border width of region-* as 0, meaning that borders cannot be drawn. AH XSL Formatter supports border widths greater than 0 on region-*, and those borders can be styled.

2.3.8 Box shadow

You can add shadows to block and inline areas.

Property	axf:box-shadow
Value	none | [inset]? && [<*length*>{2,4} && <*color*>]?]]#
Initial	none
Applies to	all elements which can have borders

axf:box-shadow can add a box shadow to any object that can have borders. <*Horizontal offset*>, <*Vertical offset*>, <*Blur radius*>, <*Shadow extent*> and <*Shadow color*> are specified as a set, and multiple shadows are separated by a COMMA “,” (U+002C). When inset is specified, the box shadow is drawn inward.

Blur is ignored when transparency is not available, such as PDF/X output or when transparency="false" is specified in the output settings.

Note

The `axf:border-radius*` properties also affect the box shadow.

When `border-style="wave"` is specified, the shape of the wave is not reflected in the shadow. If you want a wave-shaped shadow, create a wavy line frame with a shadow in advance using SVG.

2.3.9 Block transformation

You can scale, rotate, move, and skew blocks.

Property	`axf:transform`
Value	`none` | <*transform-function*>+
Initial	`none`
Applies to	transformable objects

axf:transform contains the function and arguments for transforming a block. The content of the area is transformed, and not the block area itself [Figure 2.3-4] (p.80).

Value	Value Composition
<*transform-function*>	`matrix()` | `translate()` | `translateX()` | `translateY()` | `scale()` | `scaleX()` | `scaleY()` | `rotate()` | `skew()` | `skewX()` | `skewY()`

Apart from `matrix()` and `rotate()`, if a function has one argument, the same value is applied in the X and Y directions, and if there are two arguments, the first argument is applied in the X direction and the second is applied in the Y direction. In addition, functions with names ending in X or Y transform only in that direction.

Function Name	`matrix()`
Argument Value	<*number*> , <*number*> , <*number*> , <*number*> , <*number*> , <*number*>

Transformations equivalent to `translate()`, `skew()` and `scale()` can be specified as determinant.

Function Name	`translate()`
Argument Value	<*translation-value*>[, <*translation-value*>]

`translate()` moves the block. `translateX()` moves in the X direction, and `translateY()` moves in the Y direction.

Function Name	`scale()`
Argument Value	<*number*>[, <*number*>]

`scale()` scales the block.

Function Name	`rotate()`
Argument Value	<*angle*>

`rotate()` rotates the block.

Function Name	`skew()`
Argument Value	<*angle*>[, <*angle*>]

`skew()` skews the block, `skewX()` skews in the X direction, and `skewY()` skews in the Y direction.

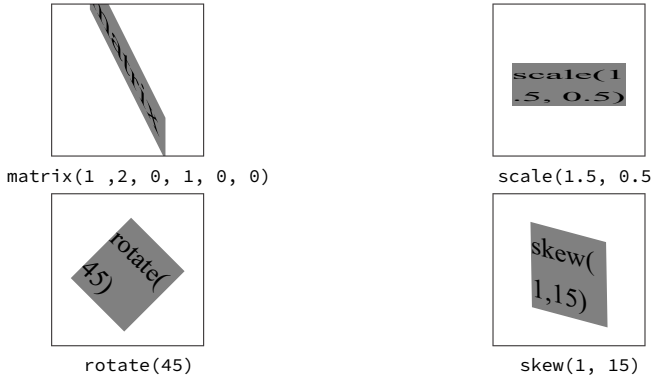


Figure 2.3-4 Block transformation

Property	axf:transform-origin
Value	[<*percentage*> | <*length*> | left | center | right | top | bottom] | [[<*percentage*> | <*length*> | left | center | right] && [<*percentage*> | <*length*> | top | center | bottom]]
Initial	center center
Applies to	transformable objects
Percentages	refer to the size of the bounding box

The axf:transform-origin property specifies the origin of the transformation. <*percentage*> and <*length*> are offsets in the X and Y directions.

If, for example, you want to make a diamond-shape block but do not want to transform the characters, generate a second block, transform that, then superimpose the block with the text over it [Figure 2.3-5] (p.81).

```
<!-- Superimpose a diamond shape and a text block -->
<fo:block-container ...>
  <fo:block-container width="1cm" height="1cm" >
    <fo:block z-index="1"
      axf:transform="translate(5mm, 5mm), rotate(45)"
      background-color="gray"
        padding-top="0.5cm" padding-bottom="0.5cm">
    </fo:block>
    <fo:block z-index="2"
      axf:transform="translate(5mm, -3.5mm)"
        text-align="center" ...
      > 第...
    </fo:block>
  </fo:block-container>
</fo:block-container>
```

Figure 2.3-5 Superimpose text on a diamond shape

2.3.10 Add line numbers

You can add line numbers to line areas in blocks and table cells.

Property	`axf:line-number`
Value	`none` | `show` | `hide`
Initial	`none`
Applies to	all block-level-formatting objects that are decendants of fo:flow, fo:table-column
Inherited	yes (except for fo:table-column)

`axf:line-number` specifies when to display and count line numbers: `show` both counts and displays line numbers; `none` neither displays nor counts; and `hide` counts lines without displaying the number.

`axf:line-number` does not apply within floats or footnotes. Common settings are specified on `<fo:page-sequence>`, but settings for line numbers within a table can be set on each `<fo:table-column>`.

Markup such as `<fo:block> </fo:block>` is also counted.

`axf:line-number`	Count line numbers	Display line numbers
`none`	-	-
`show`	✔	✔
`hide`	✔	-

Count line numbers

Property	`axf:line-number-initial`
Value	`auto` | *<number>* | `#` *<id>*
Initial	`auto`
Applies to	fo:page-sequence, fo:table-column

Property	axf:line-number-reset
Value	auto | none | page | column | force
Initial	auto
Applies to	fo:page-sequence, fo:block, fo:table-column

axf:line-number-initial on <fo:page-sequence> specifies the initial value of the line number count.

axf:line-number-reset resets the current line number to the initial value.

axf:line-number-initial="auto" on the first page sequence is equivalent to 1, otherwise auto continues the current line count. #<*id*> on <fo:table-column> specifies to continue the line count from the preceding table column with the specified id.

force, and only force, can be specified on <fo:block>: the line number count is forcibly reset to the initial value.

Display line numbers

Property	axf:line-number-start
Value	<*number*> | auto
Initial	auto
Applies to	fo:page-sequence, fo:table-column

Property	axf:line-number-interval
Value	<*number*> | auto
Initial	auto
Applies to	fo:page-sequence, fo:table-column

Property	axf:line-number-show
Value	<*number*>] *
Initial	*empty*
Applies to	fo:page-sequence, fo:table-column

axf:line-number-interval specifies how often line numbers are displayed.

axf:line-number-start specifies the first line number to display. axf:line-number-show specifies one or more numbers that are always displayed even if they would not be

displayed because of `axf:line-number-interval`, etc. Those numbers are not displayed when `axf:line-number="hide"` is specified.

Avoid counting wrapped lines

Property	`axf:line-number-except-continued-line`
Value	`true` | `false`
Initial	`false`
Applies to	all block-level formatting objects that are descendants of fo:flow, fo:table-column
Inherited	yes

When `true` is specified for `axf:line-number-except-continued-line` property and a long line wraps onto another line, only the first line is assiged a line number. The line number is not displayed on the wrapped continuation lines. When `false` is specified, line numbers are assigned even for continuation lines.

`axf:line-number-except-continued-line` is useful when you want the line numbers in the formatted output to reflect the lines in the original text, such as when showing a source code example.

2.3.11 Line number font style

You can specify the line number font independently from the block style.

Property	`axf:line-number-color`
Value	*<color>*
Initial	the value of the 'color' property
Applies to	all block level formatting objects that are descendants of fo:flow, fo:table-column
Inherited	yes

Property	`axf:line-number-font-family`
Value	[*<family-name>* | *<generic-family>*]#
Initial	depends on UA
Applies to	all block level formatting objects that are descendants of fo:flow, fo:table-column
Inherited	yes

Property	`axf:line-number-font-size`
Value	*<absolute-size>* | *<relative-size>* | *<length>* | *<percentage>*
Initial	`medium`
Applies to	all block level formatting objects that are descendants of fo:flow, fo:table-column
Inherited	yes
Percentages	refer to the font size

Property	`axf:line-number-font-style`
Value	`normal` | `italic`
Initial	`normal`
Applies to	all block level formatting objects that are descendants of fo:flow, fo:table-column
Inherited	yes

Property	`axf:line-number-font-weight`
Value	`normal` | `bold` | `bolder` | `lighter` | `1... 1000`
Initial	`normal`
Applies to	all block level formatting objects that are descendants of fo:flow, fo:table-column
Inherited	yes

The `axf:line-number-color` property and the `axf:line-number-font-*` properties apply.

Different styles can be specified for some line numbers by ending one block and starting another, or by specifying a different style in a nested block.

```
<!-- Change the line number decoration by nesting -->
<fo:block axf:line-number="show"
  axf:line-number-color="black">
  ...
  <fo:block axf:line-number="show"
    axf:line-number-color="red"
    axf:line-number-font-style="italic" >
    ...
  </fo:block>
</fo:block>
```

2.3.12 Line number style

You can specify text decoration, background color, etc. for the line numbers.

Line number format

Property	`axf:line-number-format`
Value	*<string>*
Initial	`1`
Applies to	fo:page-sequence, fo:table-column
Inherited	yes

`axf:line-number-format` specifies the line number format. Values are the same as for the `format` property.

Line number prefix

Property	`axf:line-number-prefix`
Value	*<string>*
Initial	empty string
Applies to	fo:page-sequence, fo:table-column

`axf:line-number-prefix` specifies text to be output before each displayed line number. Only literal text can be specified.

Line number text decoration

Property	`axf:line-number-text-decoration`
Value	same as text-decoration
Initial	`none`
Applies to	all block level formatting objects that are descendants of fo:flow, fo:table-column
Inherited	yes

`axf:line-number-text-decoration` applies to both the line number and the content from `axf:line-number-prefix`.

Line number background color

Property	`axf:line-number-background-color`
Value	*<color>*
Initial	`transparent`
Applies to	all block level formatting objects that are descendants of fo:flow, fo:table-column
Inherited	yes

`axf:line-number-background-color` specifies the background color of the line number area. When assigning line numbers to consecutive lines, this background color does not apply to the space between lines.

Note

To make a column of background color behind the line numbers, it is better to specify the background color on the background of the area that overlaps with the line number display.

2.3.13 Line number position

You can adjust the position of line numbers.

Property	axf:line-number-position
Value	start | end | inside | outside | alternate
Initial	start
Applies to	all block level formatting objects that are descendants of fo:flow, fo:table-column
Inherited	yes

axf:line-number-position specifies the position of the line numbers relative to the block. alternate is for use with multiple columns: line numbers are placed at the end edge in the last column and at the start edge in all other columns.

Property	axf:line-number-offset
Value	<*length*>
Initial	0pt
Applies to	all block level formatting objects that are descendants of fo:flow, fo:table-column
Inherited	yes

axf:line-number-offset specifies a relative offset in the inline progression direction offset from the position specified by axf:line-number-position.

2.3.14 Line number width and alignment

You can specify the width of the line numbers and their alignment within that width.

Property	axf:line-number-width
Value	auto | <*length*>
Initial	auto
Applies to	all block level formatting objects that are descendants of fo:flow, fo:table-column
Inherited	yes

axf:line-number-width specifies the width of the line number area. If auto is specified, the width of the line number text is used.

Property	axf:line-number-text-align
Value	auto | start | center | end | inside | outside | left | right
Initial	auto
Applies to	all block level formatting objects that are descendants of fo:flow, fo:table-column
Inherited	yes

axf:line-number-text-align specifies the line number alignment in the inline progression direction. This is valid only when axf:line-number-width is not auto.

Property	axf:line-number-display-align
Value	auto | before | center | after
Initial	auto
Applies to	all block level formatting objects that are descendants of fo:flow, fo:table-column
Inherited	yes

axf:line-number-display-align specifies the line number alignment in the block progression direction within the line area. If auto is specified, it is considered after in horizontal layout and center in vertical layout.

2.3.15 Rotated line number

You can rotate the line number display.

Property	axf:line-number-orientation
Value	0 | 90 | 180 | 270 | -90 | -180 | -270
Initial	0
Applies to	fo:page-sequence, fo:table-column

axf:line-number-orientation specifies the angle to rotate the line numbers. Only the specified values can be used. This is useful to generate upright line numbers in vertical layout [Figure 2.3-6] (p.91).

```
<!-- Rotate the line number display in vertical layout -->
<fo:page-sequence ...
```

```
  axf:line-number-orientation="90"
  axf:line-number-initial="1" axf:line-number-reset="page"
  axf:line-number-font-style="italic" >
  <fo:flow>
    <fo:block
      axf:line-number="show"
      axf:line-number-color="cmyk(0,0,0,0.5)"
      axf:line-number-font-size="smaller"
      axf:line-number-offset="4q"...>エスペラントの話を聴きたい...
    </fo:block>
  </fo:flow>
</fo:page-sequence>
```

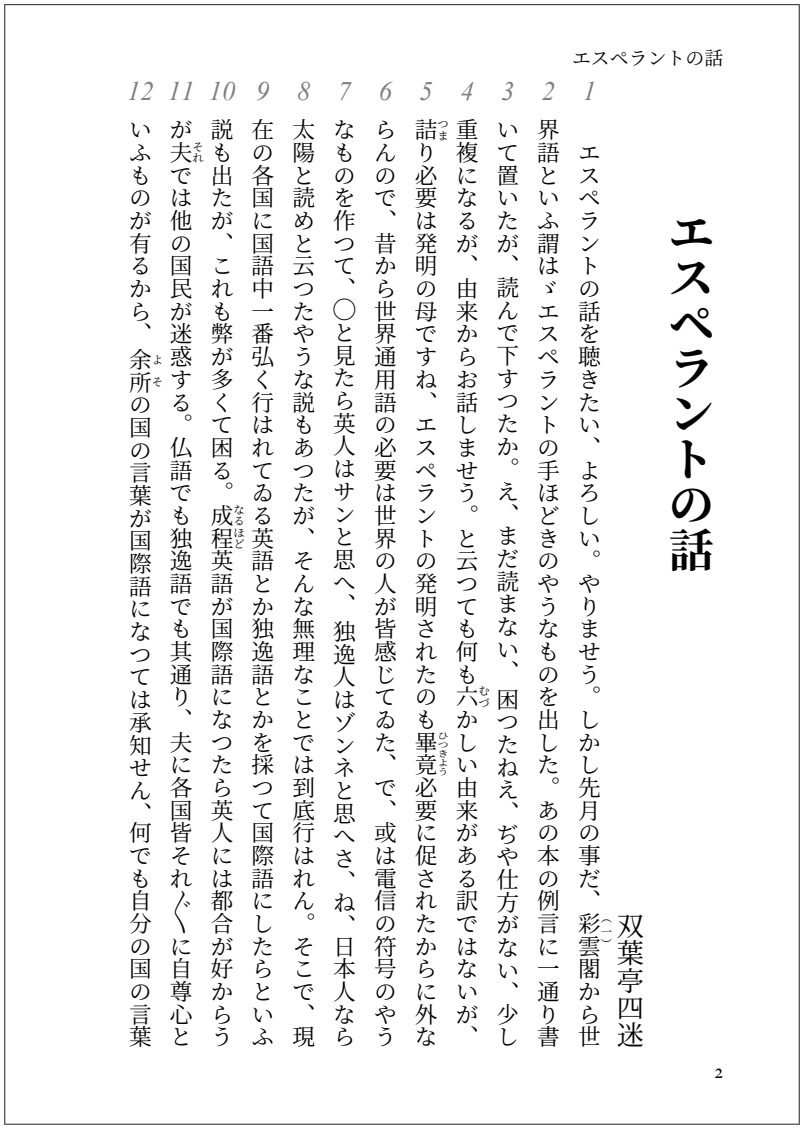


Figure 2.3-6 Rotated line numbers in vertical layout

2.3.16 Continuation mark at the end of a wrapping line

You can generate a character or symbol at the end lines that wrap.

Property	axf:line-continued-mark
Value	*<string>*
Initial	empty string
Applies to	all block level formatting objects that are descendants of fo:flow
Inherited	yes

When a line breaks, the symbol specified in `axf:line-continued-mark`on a block is displayed as a line continued mark in the area outside the block adjacent to the line break. Nothing is displayed when an empty string is specified.

This is only effective for scripts where it is simple to determine the line breaks. It does not work for blocks that mix multiple types of scripts or for complex scripts such as Arabic and Thai.

2.3.17 Line continued mark position

You can adjust the position where the line continued mark is displayed.

Property	axf:line-continued-mark-offset
Value	*<length>*
Initial	0pt
Applies to	all block level formatting objects that are descendants of fo:flow
Inherited	yes

`axf:line-continued-mark-offset` specifies a relative offset of the line continued mark.

Note

When a block has a border, if you want to display the line continued mark inside the border, specify a negative value for `axf:line-continued-mark-offset`.

2.3.18 Line continued mark style

You can specify the font and color of the line continued mark.

Property	`axf:line-continued-mark-color`
Value	<*color*>
Initial	depends on the current line area
Applies to	all block level formatting objects that are descendants of fo:flow
Inherited	yes

Property	`axf:line-continued-mark-font-family`
Value	[<*family-name*> | <*generic-family*>]#
Initial	depends on the current line area
Applies to	all block level formatting objects that are descendants of fo:flow
Inherited	yes

Property	`axf:line-continued-mark-font-size`
Value	<*absolute-size*> | <*relative-size*> | <*length*> | <*percentage*>
Initial	depends on the current line area
Applies to	all block level formatting objects that are descendants of fo:flow
Inherited	yes
Percentages	refer to the font size

Property	`axf:line-continued-mark-font-style`
Value	`normal` | `italic`
Initial	depends on the current line area
Applies to	all block level formatting objects that are descendants of fo:flow
Inherited	yes

Property	axf:line-continued-mark-font-weight
Value	normal | bold | bolder | lighter | 1... 1000
Initial	depends on the current line area
Applies to	all block level formatting objects that are descendants of fo:flow
Inherited	yes

2.3.19 Text and images in a change bar

You can display characters and images in a float as the child of a change bar.

XSL 1.1 defines a "change region" between an <fo:change-bar-begin> and its <fo:change-bar-end> that displays as a bar adjacent to block areas that can continue across pages and columns. Border decoration can be applied to this bar.

AH XSL Formatter has been extended so that floats can be placed as children of <fo:change-bar-begin>. The position of the change bar is the starting point for the reference area of this child float element.

Note

If the change-bar-offset property changes, this starting point also changes.

When change-bar-placement="alternate", the display position on the page is determined by the position of the change bar in the column. Use axf:float-x="alternate" to make the float follow the bar.

Since it follows the position of the bar, it behaves differently than alternate on a normal float.

2.3.20 Styling underlines, overlines, and strikethroughs

You can finely control the decoration of the line to be drawn in the text.

The underline, overline, and strikethrough values of the text-decoration property specify underline, overline and line-through, respectively. XSL 1.1 does not define how to style these lines.

Property	axf:text-line-color
Value	*<color>*
Initial	the value of 'color' property
Applies to	all elements with and generated content with textual content

Property	axf:text-line-style
Value	*<border-style>*
Initial	solid
Applies to	all elements with and generated content with textual content

Property	axf:text-line-width
Value	auto | *<border-width>*
Initial	auto
Applies to	all elements with and generated content with textual content

axf:text-line-color specifies the line color; axf:text-line-style, the border style; and axf:text-line-width, the line width.

Note

The same styles are applied to underline, overline and strikethrough. If you want to style each line on the same text differently, it is necessary to specify each of them on nested inline FOs.

none cannot be specified for axf:text-line-style. If you want to remove only a part of the underline, consider specifying text-decoration="no-underline".

The example specifies strikethrough as a red double line [Figure 2.3-7] (p.95).

```
<!-- Red double strikethrough line -->
<fo:inline text-decoration="line-through"
  axf:text-line-style="double"
  axf:text-line-color="red"
  axf:text-line-width="0.4em">...</fo:inline>
```

When double is specified for axf:text-line-style property, the axf:border-double-thickness property that adjusts the line thickness outside and inside of the border cannot be used.

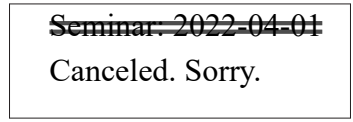


Figure 2.3-7 Red double strikethrough line

2.3.21 Position the line drawn in text

You can change the position of underlines and overlines.

Property	axf:text-underline-position
Value	auto | [[before-edge | alphabetic | after-edge] || [<*percentage*> | <*length*>]]
Initial	auto
Applies to	all elements with and generated content with textual content
Inherited	yes
Percentages	refers to the line height of the parent area

axf:text-underline-position specifies where to place an underline. If auto, it will be adjusted automatically.

axf:text-underline-position	Definition
alphabetic	Place the top edge of the underline at the baseline.
before-edge	Place the bottom edge of the underline at the before-edge.
after-edge	Place the top edge of the underline at the after-edge.
<*length*> | <*percentage*>	Shift the underline position by the specified amount.

If both placement and an offset are specified, the offset will be the distance from the center of the underline width.

Note

In vertical layout, axf:vertical-underline-side determines the underline position when axf:text-underline-position="auto". Any offset shifts in opposite directions on the left and right sides. When axf:text-underline-position="1pt" is specified, a left sideline is shifted 1pt to the left, and a right sideline is shifted 1pt to the right.

When a value is specified for the axf:text-underline-position property, the overline moves in the opposite direction to the underline. For example, if axf:text-underline-position="before-edge" is specified, text-decoration="overline" will be placed at the after-edge of the text.

axf:text-underline-position does not affect strikethrough (text-decoration="line-through") placement

2.3.22 Sideline position in vertical text

In vertical layout, an underline is treated as a sideline. You can control whether to place it on the before side or after side.

Property	`axf:vertical-underline-side`
Value	`left` | `right` | `depend-on-language` | `auto`
Initial	`auto`
Applies to	all block-level and inline-level formatting objects
Inherited	yes

Option Property	`vertical-underline-side`
Value	`left` | `right` | `depend-on-language` | `auto`
Initial	`auto`
Applies to	`<formatter-settings>`

When `axf:text-underline-position="auto"` is specified in vertical layout, `axf:vertical-underline-side` controls on which side to put the sideline. `text-decolation="overline"` will put the sideline on the opposite side to the sideline.

When `axf:vertical-underline-side` is `auto`, `vertical-underline-side` property in the option settings determines the side.

When `vertical-underline-side` is also `auto`, AH XSL Formatter defaults to using the right side with Japanese (jpn) and Korean (kor), and the left for other languages. If nothing is specified, it is determined by the `default-CJK` option setting. The same is applied for the `axf:vertical-underlineside="depend-on-language"`.

For example, the use case “when you want to include a European phrase in vertical Japanese and place all the sidelines on the right side” can be achieved with `axf:vertical-underline-side="right"` [Figure 2.3-8] (p.98). Using this, the underline is placed on the right side even though the European phrase is oriented upright.

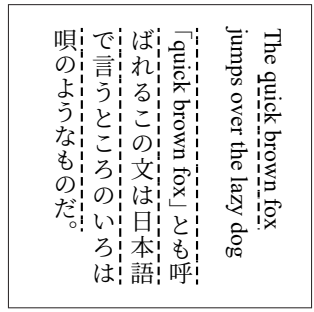


Figure 2.3-8 Example of sideline for European phrase in vertical layout

2.3.23 Text shadow

You can add shadows to text and control its blurring.

`text-shadow` allows you to add a shadow to text. The XSL 1.1 specification defines `text-shadow` as not inherited, but has been extended in AH XSL Formatter to be inheritable. In addition, it is extended to apply to more FOs.

Property	`text-shadow`
Value	`none` | [<*length*>{2,3} `&&` <*color*>] ?]#
Initial	`none`
Applies to	**all elements which can have borders**
Inherited	yes

No shadow is added when the value is `none`. <*Horizontal shadow offset*>, <*Vertical shadow offset*>, <*Blur radius*> and<*Shadow color*> can be used as a set to add multiple shadows.

When blurring, the shadows in the text are in a raster image format with transparency. Shadows in outputs that do not support transparency will not be blurred.

Option Property	`textshadow-resolution-minimum-dpi`
Value	<*integer*>
Initial	`108`
Applies to	`<formatter-settings>`

Option Property	textshadow-resolution-pixel-per-em
Value	*<integer>*
Initial	100
Applies to	<formatter-settings>

Option Property	textshadow-blur-cannot-embed-font
Value	true | false
Initial	false
Applies to	<formatter-settings>

Blurring of the text-shadow property can be controlled by option settings.

textshadow-resolution-pixel-per-em specifies the resolution for blurring. The value is specified by the number of pixels per side of the font size.

textshadow-resolution-minimum-dpi specifies the minimum dpi for blurring. If the dpi calculated from the textshadow-resolution-pixel-per-em property is less than this minimum dpi, this will take precedence.

textshadow-blur-cannot-embed-font controls whether text-shadow blurring is performed where non-embeddable fonts are used. Blurring is performed with true.

2.3.24 Use Case: Neumorphism

You can apply a neumorphism design to blocks.

Neumorphism uses multiple shadows to make elements look recessed and raised rather than simply [Figure 2.3-9] (p.100). Recessed and raised appearance can be achieved with inset and outset borders, but axf:box-shadow is useful if blurring the border.

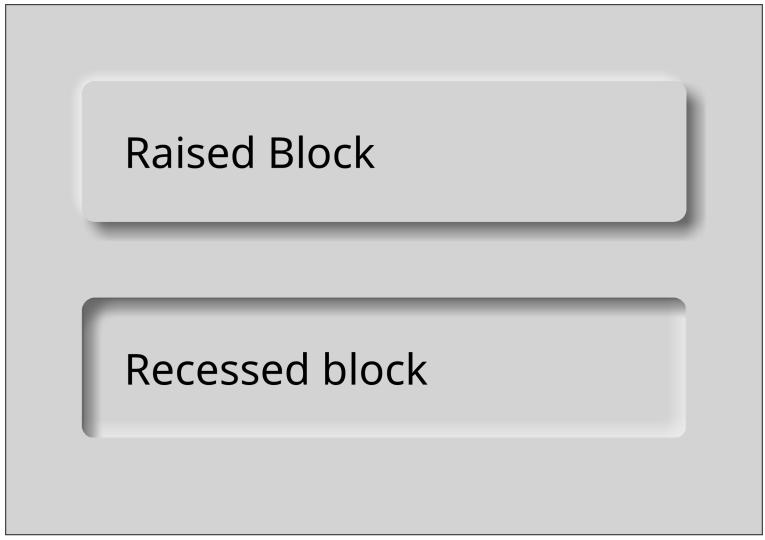


Figure 2.3-9 Neumorphism design block decoration

Raised block

```
<!-- Make the block appear raised  -->
<fo:block
  axf:box-shadow="-3px -3px 3px rgba(255,255,255,0.5),
    5px 5px 5px rgba(0,0,0,0.5)"
  margin="10mm" padding="1em"
  axf:border-radius="5pt">
  ...
</fo:block>
```

Suppose the light source is in the upper left and the element is raised. Place a bright shadow in the upper left and a dark shadow in the lower right. In the upper left, specify a negative value for the offset and a color brighter than the background color to express the reflected light.

Since multiple sets can be specified in the `axf:box-shadow` property, the shadow in the lower right is specified by placing " , " in succession. If the background is an image, the shadow color should be partially transparent so the shadow blends in better.

Margins have been added so that the shadow does not overlap with other objects.

Specify rounded corners all at once with `axf:border-radius`.

Recessed block

```
<!-- Make the block appear recessed - -->
<fo:block
  axf:box-shadow="inset 5px 5px 5px rgba(0,0,0,0.5),
    inset -5px -5px 5px rgba(255,255,255, 0.5)"
  padding="1em" margin="10mm"
  axf:border-radius="5pt">
  ...</fo:block>
```

Suppose the light source is in the upper left and the element is recessed. Place a dark shadow inside the element, in the upper left, and a light shadow in the lower right. Specify `inset` for `axf:box-shadow` to display the shadow inside the element.

2.3.25 Use case: Source code display

You can style source code listings.

Style the source code by specifying: background and border; captions that overlap blocks; line numbers and continuation marks; and controlled spaces and line breaks [Figure 2.3-10] (p.101).

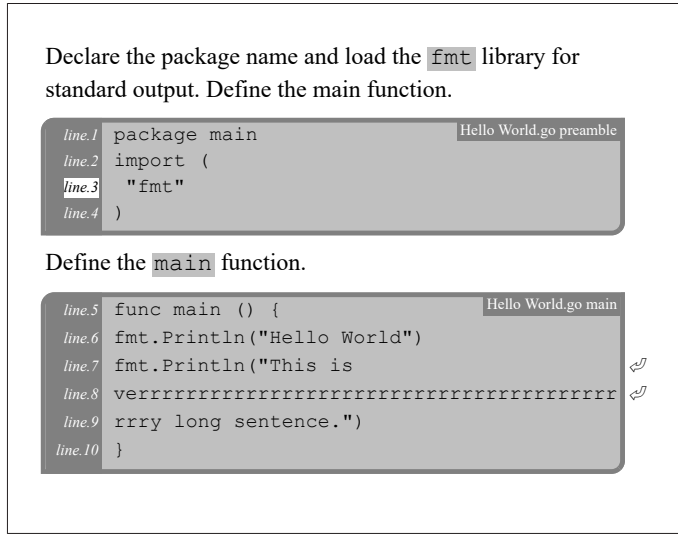


Figure 2.3-10 Add line numbers and continuation marks to the source code

Block container with background and border

```
<!-- Background and border block container  -->


  ...
<fo:block-container>
```

Generate a block container for the background and borders. By nesting and separating the blocks that describe the source code, it can flexibly respond when changing the line processing in the middle of the source code. The width is made slightly narrower than the body area.

The `linear-gradient()` function in the `background-image` property fills the background with two colors.

The border at the top left of the block is rounded with `axf:border-top-left-radius="5pt"`, and the border at the bottom right of the block is rounded with `axf:border-bottom-right="5pt"` so that page breaks can be easily distinguished. Borders are not drawn at the page break, so the rounded corner marks the beginning and end of the source code.

Caption in upper-right corner

```
<!-- Background and border block container  -->
<fo:block-container ...>
<!-- Caption block container  -->
  <fo:block-container
    z-index="10" absolute-position="absolute"
    top="0.2em" right="0">
    <fo:block line-height="1"
      margin-top="0pt" font-size="xx-small"
      text-align="right">
      <fo:inline padding="0.2em"
        background-color="gray" color="white">
        Hello World.go preamble</fo:inline>
      </fo:block>
  </fo:block-container>
...
</fo:block-container>
```

Use an absolutely positioned block container to display the caption of the source code in the upper right. When an absolutely positioned block container (`absolute-position="absolute"`) is placed as a child element of the block container

with the width of the source code specified, the points of `top="0"` and `right="0"` will be the top-right end.

The absolute position of the block container is displayed overlapping with other objects. So, specify `z-index="10` to place it above a normal object.

The background color is specified for the inline element, and the applied range of the background color is expanded using `padding="0.2em"`. The previously specified `top="0"` is reduced by this amount to be `top="0.2em"`.

Display line numbers and wrapping symbols in source code

```
<!-- A block that describes the source code -->
<fo:page-sequence ...
  axf:line-number-reset="none" axf:line-number-prefix="line.">
  ...
  <fo:block axf:line-number="show"
  axf:line-number-color="white"
  axf:line-number-font-size="xx-small" axf:line-number-font-style="
italic"
  axf:line-number-font-family="serif" axf:line-number-offset="-7mm"
  axf:line-continued-mark="&#x23CE;"
  axf:line-continued-mark-font-size="x-small"
  axf:line-continued-mark-font-style="italic"
  axf:line-continued-mark-font-family="fantasy"
  axf:line-continued-mark-offset="3pt"
  axf:line-continued-mark-color="black"

  font-family="monospace" font-size="smaller" line-height="1.4"
  white-space-treatment="preserve"
  linefeed-treatment="preserve"
  wrap-option="wrap"
  margin-left="9mm"
  padding-top="0.2em" padding-bottom="0.2em">package main
import (
    "fmt"
)
...</fo:block>
<fo:page-sequence>
```

To preserve spaces and line breaks and make long lines wrap, specify `preserve` for `white-space-treatment` and `linefeed-treatment`, and wrap for `wrap-option`.

`axf:line-number-initial` specifies the line number of the first line. Because `axf:line-number-reset="none"` is specified, the line numbers are basically continued. If you want to reset the line number, specify `axf:line-number-reset="force"` on `<fo:block>`.

A string is specified in `axf:line-number-prefix` to display "*lines.*K before the line numbers. If you want to use multiple types of line numbers with different prefix strings, start a new page sequence.

The line number is displayed next to the block where the source code is written using `axf:line-number="show"`. `axf:line-number-font-*` specify the font for the line numbers.

The initial position of the line number display is outside the block, but the negative `axf:line-number-offset` value moves it inside the block.

The text specified in `axf:line-continued-mark` is displayed when a line wraps. This example uses ⏎.

`axf:line-continued-mark-font-*` specify the font to use for this symbol.

To prevent counting and showing a line number on continued line, you can use `axf:line-number-except-continued-line`.

Change line number style in the middle of source code

```
import (<fo:block axf:line-number-color="black"
  axf:line-number-background-color="white"
  font-family="monospace">    "fmt"</fo:block>)</fo:block>
```

The nested block changed the text color and background color of line numbers.

2.3.26 Use case: Create a speech bubble conversation

You can combine the `axf:transform` property and the list block FO to create a speech bubble conversation [Figure 2.3-11] (p.105).

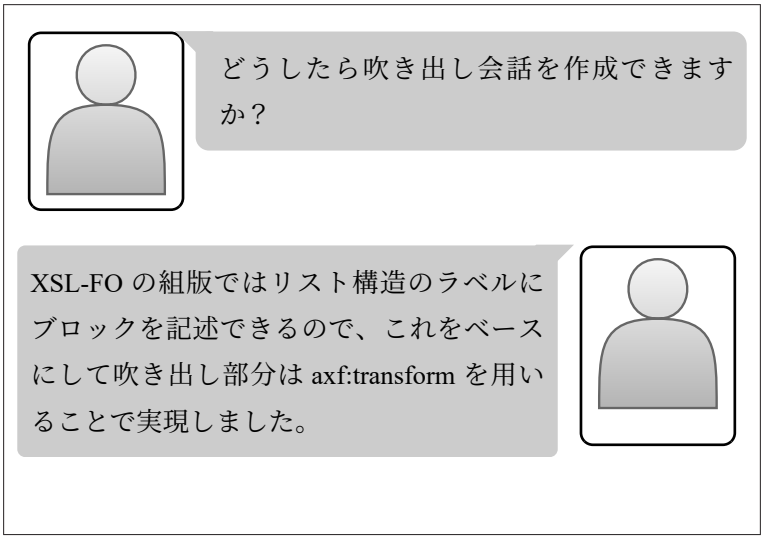


Figure 2.3-11　Create a speech bubble conversation

Create a list with labels on the right

```
<!--  List block description -->
<fo:list-block
  space-before="0.4em" space-end="0.4em">
  <!-- Item with labels on the left -->
  <fo:list-item space-after="0.5em">
    <fo:list-item-label
      start-indent="0mm" end-indent="100vw - 2cm">
      ...
    </fo:list-item-label>
    <fo:list-item-body
      start-indent="2.5cm" end-indent="0">
      ...
    </fo:list-item-body>
  </fo:list-item>
  <!-- Item with labels on the right -->
  <fo:list-item>
    <fo:list-item-label
      start-indent="100vw - 2cm" end-indent="0">
      ...
    </fo:list-item-label>
    <fo:list-item-body
      start-indent="0" end-indent="2.5cm">
      ...
```

```
    </fo:list-item-body>
  </fo:list-item>
</fo:list-block>
```

The list structure is useful because it allows a block in the label part.

Because the label is placed on the start side and the body is placed on the end side, it is not possible to use `label-end()` and `body-start()` to calculate indents when placing the label on the right side.

The `start-indent` and the `end-indent` of the item label and body are calculated assuming that the image in the label is 2 cm wide.

`100vw - 2cm` is specified for the start position of the label, where 100 vw is the width of the list block.

Place the image on the label

```
<!-- Specify an external image inside the label --->
<fo:list-item-label ...>
  <fo:block axf:border-radius="5pt" border="solid 1pt black">
    <fo:external-graphic src="url(./image.jpg)"
      width="2cm" height="2cm" content-width="scale-to-fit"/>
  </fo:block>
  ...
</fo:list-item-label>
```

`width="2cm"` and `content-width="scale-to-fit"` are specified for `<fo:external-graphic>`. The image is scaled to be 2 cm wide.

Transform the block container to create a speech bubble

```
<!-- List label -->
<fo:list-item-label ...>
  <fo:block ...>
    <fo:external-graphic .../>
  </fo:block>
  <fo:block-container background-color="cmyk(0,0,0,0.2)"
    axf:transform="translate(1.8cm, -2.38cm) skewX(45) scaleX(0.5)"
    width="1cm" height="0.8rem" />
</fo:list-item-label ...>
```

Use `axf:transform` to transform a block container with a specified size to create a parallelogram speech bubble.

Because the parallelogram is part of the label, it is not necessary to adjust its position even if the list item body becomes long.

Use `translate()` to move the position, use `skewX()` to tilt in the X direction and use `scaleX()` to adjust the width of the tilt. The background color matches the background color of the `<fo:list-item-body>`. Different shapes would be possible by using SVG or other formats to represent the speech bubble.

2.3.27 Use case: Add revision number to change bar

You can add text to the change bar to indicate the revision number and display them together.

A change bar has a specified style and color, but you may want to add more information. You can add the revision number information where it will not interfere with the body text [Figure 2.3-12] (p.110)

Placement of change bar and float

```
<!--  Change bar and float  -->
<fo:block>...複数用意されることになります。</fo:block>
<fo:block>
  <fo:change-bar-begin ...
    change-bar-class="tr0101"
    change-bar-placement="alternate"... >
    <fo:float axf:float-x="alternate" ... >
      <fo:block ...>R.1</fo:block>
    </fo:float>
  </fo:change-bar-begin>
  Word のような DTP ソフトでドキュメントを作る場合、...という仕組みもありま
す。...
  <fo:change-bar-end change-bar-class="tr0101" />
</fo:block>
```

`change-bar-placement` specifies where to place the change bar, but where to place the `<fo:float>` child is specified separately. The origin for placing the float is based on the placement of the change bar.

When the float is placed outside the change bar, if the placement setting of the change bar is in a fixed direction such as `start` or `end`, it is convenient to place an absolutely positioned block container as a child of the float.

However, absolute positioning cannot be used when the change bar position differs depending on the position on the page, such as when `change-bar-placement="alternate"` is specified. Therefore, it is necessary to specify `axf:float-x="alternate"` on the float so that it matches the change bar.

Condensed font in change bar

```
<!--  Use if there is a condensed font -->
<fo:declarations>
  <axf:formatter-config
      xmlns:axe="http://www.antennahouse.com/names/XSL/Settings">
    <axe:font-settings font-stretch-mode="6" />
  </axf:formatter-config>
</fo:declarations>
...
<fo:page-sequence ...>
  <fo:flow ...>
    ...
    <fo:block>...複数用意されることになります。</fo:block>
    <fo:block>
      <fo:change-bar-begin ...>
        <fo:float ... >
          <fo:block font-family="Bahnschrift" font-stretch="semi-co
ndensed"
          ... >R.1
          </fo:block>
        </fo:float>
      </fo:change-bar-begin>Word のような...
    </fo:block>
  ...
  </fo:flow>
</fo:page-sequence>
```

Place the change bar outside the body area and outside the line. Given that multiple change bars will be placed, a condensed font is used for the text that accompanies the change bars. In AH XSL Formatter, when `font-stretch` is `condensed` and the font family includes a condensed font, that font is selected.

Change bar offset

```
<!-- Specify the position of the change bar and float -->
<fo:change-bar-begin change-bar-class="tr0101"
  change-bar-placement="alternate" change-bar-width="4pt"
  change-bar-offset="10pt - 4pt">
  <fo:float
    axf:float-x="alternate" axf:float-offset-x="-42pt"  >
    <fo:block ...>R.1</fo:block>
  </fo:float>
</fo:change-bar-begin>
```

Change the offset to prevent the image and text in the float from overlapping with the change bar. Specify the `change-bar-offset` property so that the change bars do not overlap. The float offset is changed with the `axf:float-offset-x` property.

When `change-bar-placement="alternate"` or `axf:float-x="alternate"` is specified, each offset changes with the placement direction.

DITA とは

DITA とは、XML の技術を応用した、マニュアルや書籍等のコンテンツを執筆するための仕様です。この仕様に則って書かれたコンテンツから印刷物（PDF）やHTML、Web ヘルプ等の配布可能な媒体を効率よく作成することを目的としています。DITA はコンテンツを最小単位に切り分けたトピックファイル、そしてコンテンツの構造を決めるマップファイルで構成されており、ドキュメントのコンポーネント化、システム化を志向しています。

トピックファイル

トピックファイルとは、1 つの話題のみ書かれたファイルです。たとえば「電源の入れ方」とか「電源の切り方」といった話題毎にトピックファイルを作ることになります。そのため、トピックファイルは必然的に複数用意されることになります。

R.1

Word のような DTP ソフトでドキュメントを作る場合、一般的には 1 つのファイルに全ての話題を詰め込むことになりますが、DITA の場合、話題毎にトピックファイルに分割します。その際、ドキュメントを単に分割すればいいというわけではなく、情報構造をあらかじめ整理することが極めて重要です。 執筆する話題に合わせて、concept（概要）、task（手順）、reference（参考書類）、troubleshooting（トラブルシューティング）等のトピックタイプ（情報タイプ）が標準で用意されています。これらの情報タイプだけではどうしてもうまく表現できない（あるいは執筆する際にもっと強い制約が欲しい）場合のために特殊化という仕組みもあります。

R.2

R.2 R.1

マップファイル

マップファイルはドキュメントの構造を決める役割を持っています。具体的には、トピックファイルの出力順やトピック間の階層構造です（目次に相当する情報と言ってもいいでしょう）。トピックファイルとマップファイルは完全に分離されている為、マップファイルを変更する事で、コンテンツであるトピックファイルに手を加えることなくドキュメントの構造を変化させることができます。

Figure 2.3-12　Change bars with revision numbers

Section 2.4 Columns

You can specify the different numbers of columns, style the column gap on the page. And you can specify tricky spanning.

2.4.1 Mix different columns in the same flow

To mix different numbers of columns within the one region, you can create multiple block containers with differing numbers of columns.

XSL 1.1 defines `column-count`, for specifying the number of columns, and `column-gap`, for specifying the space between columns, only on `<fo:region-body>`.

AH XSL Formatter allows you to specify `column-count` on `<fo:block-container>`. This allows different numbers of columns in one flow.

```
<!-- - Example of mixing different columns -->
<fo:page-layout-master>
  <fo:simple-page-master ...>
    ... <!-- Two columns of body text -->
    <fo:region-body
      column-count="2" column-gap="8mm" ... />
  </fo:simple-page-master>
  ...
</fo:page-layout-master>
<fo:page-sequence ...>
  ...
  <fo:flow ...>
    <!-- Two columns -->
    <fo:block
    <fo:block-container
      span="all"
      column-count="3" column-gap="3mm" ...>
        <!-- Three columns -->
        <fo:block ...>...>/fo:block>
      </fo:block-container>
    <!-- Two columns -->
    ...
  </fo:flow>
</fo:page-sequence>
```

2.4.2 Align column content height

If you arrange the content so that it fills the columns sequentially from the first column, the last column may have more empty space than the other columns.
You can adjust the amount of content for each column and specify to some extent that the height of the columns should be aligned.

Property	axf:column-fill
Value	auto | balance | ~~balance-all~~
Initial	**balance**
Applies to	multi-column elements

axf:column-fill is based on 'column-fill' in [CSS3-Multicol] 'column-fill'. When balance is specified, the height of the content is adjusted (balanced) to the same level within the columns, as far as possible. When columns continue over multiple pages, only the last page is balanced.

When auto is specified, only the content before a column-spanning element (one with span="all") is balanced. 'column-fill' in [CSS3-Multicol] 'column-fill' also defines a balance-all value that AH XSL Formatter does not support.

2.4.3 Column rule appearance

You can specify the appearance of the ruled line between the columns (column rule).

Property	axf:column-rule-color
Value	<*color*>
Initial	the value of the <*color*> property
Applies to	fo:region-body, fo:block-container

Property	axf:column-rule-style
Value	<*border-style*>
Initial	none
Applies to	fo:region-body, fo:block-container

Property	axf:column-rule-width
Value	<*border-width*>
Initial	medium
Applies to	fo:region-body, fo:block-container

The rule specified by axf:column-rule* is drawn in the center of the column gap specified by column-gap.

2.4.4 Column rule length and alignment

The column height is calculated inside the padding box.

You can specify the length of the column rule independently from the height of the column. You can also specify how to align the rule when the rule length is less than 100%.

Property	`axf:column-rule-length`
Value	*<length>* | *<percentage>*
Initial	`100%`
Applies to	fo:region-body, fo:block-container
Percentages	refer to the height of the column

`axf:column-rule-length` specifies the length of the column rule. If a column is split by a column-spanning element, each part is considered to be 100 %.

Property	`axf:column-rule-align`
Value	`before` | `center` | `after`
Initial	`center`
Applies to	fo:region-body, fo:block-container

When the `axf:column-rule-length` length is other than 100%, `axf:column-rule-align` controls how to align the column rule: on the before side, after side or center.

2.4.5 Column rule for empty columns

You can control whether to draw column rules next to columns that have no content.

In a multi-column document, the last columns may be empty, such as on the last page.

Property	`axf:column-rule-display`
Value	`gap` | `end` | `all`
Initial	`gap`
Applies to	fo:region-body, fo:block-container

When the column rule is visible, `axf:column-rule-display` specifies whether to draw a column rule even for columns where there is no content (i.e., where there is no column, there is no column gap).

axf:column-rule-display	Definition
gap	Draw a column rule only when there is a column gap.
end	Except for the column on the end-side, draw a column rule on the end side of each column that has content.
all	Draw a column rule regardless of the presence of column contents.

2.4.6 Change the display position of the element depending on the column

In multi-column layouts, the arrangement and alignment of some areas can be specified to change depending on the current column number.

When the value of the `*-position` properties for placement of revision bars and line numbers is `alternate`, the placement changes depending on the column.

In a three-column horizontal layout, starting from the leftmost column, the positions will be start, start and end. Even when there is an even number of columns, the positions are not evenly divided between start and end. For example, in a four-column horizontal layout, the positions will be start, start, start and end.

The alternate value of `axf:float-x`, which specifies the x-axis alignment of a float, aligns the float towards the center. This is different from how alternate in other properties aligns areas relative to blocks. In a multi-column layout, a float with `axf:float-x="alternate"` that is a child of a change bar will follow the placement specified for the change bar.

Section 2.5 Table formatting

Flexibly control the arrangement and dividing etc. of elements in the table.

2.5.1 Change the operation of header and footer when breaking by page or column

The operation control of the header and footer when the table is breaking by page and column has been extended.

The elements of the table may break at the timing of page breaking in case of one column, and at the timing of page breaking or column breaking if the table is used in the columns.

Property	`table-omit-header-at-break`, `table-omit-footer-at-break`
Value	`true` | `false` | **`column`**
Initial	false
Applies to	fo:table

How to display the header and footer at the table breaking can be specified in the `table-omit-header-at-break` and `table-omit-footer-at-break` properties respectively.

In the XSL 1.1 specification, you can only specify whether to omit the display (`true`) or not (`false`) at the page breaking.

AH XSL Formatter adds control on columns. When `column` is specified, the table header and footer are omitted in column breaking and displayed in page breaking.

2.5.2 Controls the repetition of table header and footer notes

When breaking a table, specify whether to repeat the notes described in the header and footer of the table before and after the break.

Whether to repeat the table header and footer before and after breaking can be controlled by specifying the `table-omit-*-at-break` property. When displaying the header and footer of the table before and after the break, AH XSL Formatter can control whether to repeat the note described there.

Property	axf:repeat-footnote-in-table-header
Value	true | false
Initial	true
Applies to	fo:table-header
Inherited	yes

Property	axf:repeat-footnote-in-table-footer
Value	true | false
Initial	true
Applies to	fo:table-footer
Inherited	yes

When displaying the table header and footer at the page and column breaking, you can specify in the `axf:repeat-footnote-in-table-*` property whether the footnotes in the header and footer are displayed repeatedly before and after breaking.

If `true` is specified, it will be displayed repeatedly, and if `false` is specified, it will be hidden.

As a use case, if you specify `axf:repeatfootnote-in-table-*="false"` when displaying the table header or footer for each column in columns, it will avoid the display of the same footnote repeatedly in the page.

Note

When there are footnotes for each column by `axf:footnote-position="column"`, `axf:suppressduplicate-footnote="true"` suppresses the duplicate display of notes only within the same column. At this time, the same note may be displayed multiple times on the page.

2.5.3 Control the content when breaking cells

When a break occurs in the middle of a cell, change the content of the split cell.

When another cell breaks in the same row, or a cell that spans multiple rows with `number-rows-spanned` breaks, nothing is displayed behind the split cell.

With AH XSL Formatter, it is possible to control the display of cells before and after page and column breaking. It is also possible to display another element in the split cell only when the cell breaking occurs.

Property	axf:repeat-cell-content-at-break
Value	true | false
Initial	false
Applies to	fo:table-cell

In the axf:repeat-cell-content-at-break property, you can specify whether to display the cell contents repeatedly before and after the cell breaking.

When axf:repeat-cell-content-at-break="true" is specified, the cell content is copied and displayed on the cell in the page and column after breaking.

Element	<axf:table-cell-repeated-marker>
Parent	fo:table-cell

If you describe the content you want to display in <axf:cell-repeated-marker> and place it as the first child element of <fo:table-cell>, the described contents are displayed in the cell after the breaking when axf:repeat-cell-content-at-break="true" is specified.

The content of <axf:cell-repeated-marker> is independent of the content of other child elements of <fo:table-cell>. Suppose you want to display "Column Title" in the cell before breaking and “Column Title (continuation)” in the cell after breaking when the cell breaking occurred. In the description of the child of <axf:cell-repeated-marker>, describe the “Column Title (continuation)” including the “Column Title”.

```
<!-- When breaking the cell, display the "Column Title (continuatio
n)" in the cell in after. -->
<fo:table-cell ... number-rows-spanned="2"
  axf:repeat-cell-content-at-break="true">
  <axf:table-cell-repeated-marker>
    <fo:block>Column Title (continuation)</fo:block>
  </axf:table-cell-repeated-marker>
  <fo:block>Column Title</fo:block>
</fo:table-cell>
```

2.5.4 Limit the number of lines used for automatic width calculation

Limit the automatic calculation of the table width when the number of rows in the table is large, or increase the number of rows in the automatic calculation consideringthe large number of rows.

`<fo:table>` The table-layout property of the `<fo:table>` element specifies whether the column width of the table is fixed (`fixed`) or automatically calculated (`auto`). If specifying `fixed`, the formatting speed will be faster. However, if the content exceeds the specified width at this time, the content may be exceeding the cell width. It is recommended to use it when you can specify the cell contents and their width in advance.

With `table-layout="auto"`, the XSL-FO processor examines the contents of the table and calculates the width of each. Since the width is calculated after examining multiple rows instead of just single row, the more rows to examine, the longer it takes to calculate. See the manual for details.

The initial value for the number of columns to check the contents for automatic calculation in the AH XSL Formatter is 100 rows, but this value can be changed.

Property	`axf:table-auto-layout-limit`
Value	`auto` | *<integer>*
Initial	`auto`
Applies to	fo:table
Inherited	yes

Option Property	`table-auto-layout-limit`
Value	*<integer>*
Initial	`100`
Applies to	`<formatter-settings>`

Specify the number of rows to be used for automatic calculation of width in the table in the `axf:table-auto-layout-limit` property in the option settings. When `axf:table-auto-layout-limit="auto"` is specified, it will be the value specified in the `table-auto-layout-limit` property in the option settings.

2.5.5 Control the text alignment within cells

If you specify *<string>* for the `text-align` property in the `<fo:table-cell>` element, the text on each line will be aligned with the specified character position.

`<fo:table-cell>` Specifying *<string>* is mainly useful when aligning the numerical display to the position of the decimal point, but at this time, there is no definition in the XSL 1.1 specification as to which side the aligned strings should be aligned. In AH XSL Formatter, it is aligned on the end side, but this behavior can be changed.

Property	axf:text-align-string
Value	start | center | end | inside | outside | left | right
Initial	end
Applies to	fo:table-cell
Inherited	yes

By specifying the alignment position in the axf:text-align-string property, the entire text of each line aligned with text-align="<*string*>" will be aligned to the <*string*> position.

2.5.6 Justify the text in the cell in the block progression direction

Property	display-align
Value	auto | before | center | after | **justify**
Initial	auto
Applies to	block-level objects
Inherited	yes

When display-align in <fo:table-cell> is justify, the text in the cell is justified in the block progress direction. This is useful for cells with text of different heights, for cells that span multiple rows, or for a cell that is a header in both vertical and horizontal directions.

2.5.7 Equal height for spanned rows

When a table has cells that spans rows (with number-rows-spanned), the heights of the spanned rows may be unequal.

When the justify-rowspan-height option is true, the heights of the rows containing the merged cells are adjusted to be equal, where possible.

Option Property	justify-rowspan-height
Value	true | false
Initial	false
Applies to	<formatter-settings>

The height adjustment depends on the contents of the target cells. For rows and cells with specified heights and for cells with rotated text, only their following rows are justified. `justify-rowspan-height` also applies to `<fo:table>` that have a specified height.

2.5.8 Number of rows retrieved from table marker

When a marker referenced in the header or footer contains multiple rows, you can specify the maximum number of rows to retrieve to avoid unwanted effects when too many rows are retrieved.

For example, when the header contains a cell that spans multiple rows plus other content in the spanned rows, you may generate a marker containing multiple rows and use them as the spanned rows. In AH XSL Formatter, when drawing a table, each row specifies the number of rows that it spans. `<fo:marker>` can contain multiple `<fo:table-row>`. There may be problems in handling the number of rows when retrieving a marker.

Property	`axf:retrieve-table-rows`
Value	*<integer>*
Initial	`1`
Applies to	fo:retrieve-table-marker

`axf:retrieve-table-rows` specifies the maximum number of rows to be extracted by `<fo:retrieve-table-marker>`. By using the specified maximum number of rows, you can avoid problems in handling the number of rows and display them correctly.

```
<!-- Extract markers containing multiple rows  -->
<fo:table>
  <fo:table-header>
    <fo:retrieve-table-marker retrieve-class-name="multiple"
      axf:retrieve-table-rows="3"/>
  </fo:table-header>
  ...
  <fo:marker marker-class-name="multiple">
        <fo:table-row>...</fo:table-row>
        <fo:table-row>...</fo:table-row>
        <fo:table-row>...</fo:table-row>
  </fo:marker>
  ...
</fo:table>
```

2.5.9 Example: Styling a table

Diagonal border and vertical heading

[Figure 2.5-1] (p.124) shows an example of a diagonal border in a table with headers in two directions.

```
<!--  Align the height of cells spanning rows -->
<fo:declarations>
  <axf:formatter-config
    xmlns:axs="http://www.antennahouse.com/names/XSL/Settings">
    <axs:formatter-settings justify-rowspan-height="true" />
  </axf:formatter-config>
</fo:declarations>
```

Since the table has cells spanning multiple rows, `justify-rowspan-height="true"` in the option setting makes the cell heights equal, to some extent.

```
<!-- Headings with diagonal border and vertical headings  -->
<fo:table padding="4mm" table-layout="fixed"
    table-omit-header-at-break="page"
    display-align="center" border="1pt solid k100">
  <fo:table-column
    column-number="1" border="0.4pt solid k100" margin="2mm" />
  <fo:table-column
    column-number="2" column-width="3rem" border="0.4pt solid k100
" />
  ...
  <fo:table-header
    border="0.4pt solid k100">
    <fo:table-row height="7rem">
      <fo:table-cell padding="3mm"
        axf:diagonal-border-style="dashed"
        axf:diagonal-border-width="0.4pt"
        axf:diagonal-border-color="k100"
        axf:scope="both">
          <fo:block text-align="end">Name</fo:block>
          <fo:block text-align="start">License</fo:block>
      </fo:table-cell>
      <fo:table-cell >
        <fo:block-container writing-mode="tb-rl"
          block-progression-dimension="3rem">
          <fo:block>アンテナ太郎</fo:block>
        </fo:block-container>
      </fo:table-cell>
```

```
    ...
    </fo:table-row>
  </fo:table-header>
  ...
</fo:table>
```

`display-align="justify"` specifies vertical justification of cell content. `column-width` specifies the width of the `<fo:table-column>`. Also `block-progression-dimension` specifies the width of the rotated block container inside the cell.

In the upper left cell, diagonal border and alignment changes are specified to indicate that headers are for both vertical and horizontal directions. Since the a body cell also uses diagonal borders, the header cell uses `axf:diagonal-borderstyle="dashed"` to generate a dashed line. In order to make it easier to see which heading applies in which direction, in addition to the diagonal border, `axf:display-align="justify"` vertically separates the headings.

```
<!-- Specify the diagonal border for both directions to the cel
l -->
<fo:table>
  <fo:table-body>
    <fo:table-row height="2em" padding="3mm"  border="0.3pt solid k
100">
      <fo:table-cell text-align="left" margin="3mm"
        padding-left="3mm" padding-right="3mm">
          <fo:block>AH XSL Formatter</fo:block>
        </fo:table-cell>
      <fo:table-cell number-rows-spanned="2"
        axf:diagonal-border-width="0.3pt"
        axf:diagonal-border-style="solid"
        axf:diagonal-border-color="k100"
        axf:reverse-diagonal-border-width="0.3pt"
        axf:reverse-diagonal-border-style="solid"
        axf:reverse-diagonal-border-color="k100">
          <fo:block>    </fo:block>
      </fo:table-cell>
      <fo:table-cell>
        <fo:block>
          <fo:character font-family="fantasy" character="&#x2713;"
/>
        </fo:block>
      </fo:table-cell>
    ...
    </fo:table-row>
  ...
  </fo:table-body>
</fo:table>
```

By specifying `axf:diagonal-border-*` and `axf:reverse-diagonal-border-*`, an "X" mark was drawn in the cell. `number-rows-spanned="2"` makes the cell spans two columns.

`fo:character character="✓"/` generates the check mark.

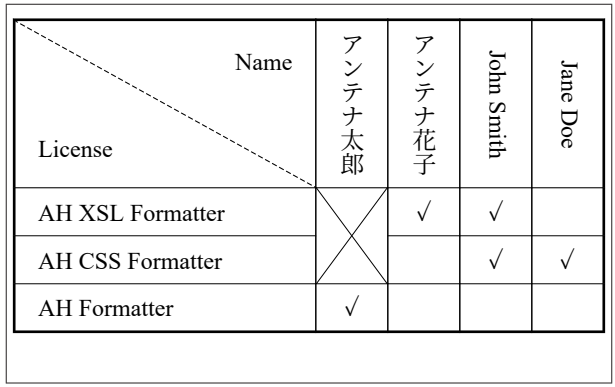


Figure 2.5-1 Diagonal border and vertical heading

A table with numbers aligned on decimal point

[Figure 2.5-2] (p.125) shows a table with numerical values.

```
<!-- A table with numbers aligned on decimal point  -->
<fo:table ...  table-omit-header-at-break="false">
  <fo:table-column border="0.3pt solid k100"/>
  <fo:table-column  border="0.3pt solid k100"/>
  <fo:table-column  border="0.3pt solid k100"/>
  <fo:table-header
   >
    <fo:table-row text-align="center"
      border="0.3pt solid k100">
      <fo:table-cell padding="1mm">
        <fo:block>製品名</fo:block>
      </fo:table-cell>
      <fo:table-cell display-align="justify" ...>
        <fo:block>処理 1<fo:block/>実行時間（秒）</fo:block>
      </fo:table-cell>
      ...
    </fo:table-row>
  </fo:table-header>
    <fo:table-body>
  <fo:table-row background-color="cmyk(0,0,0,0.3)">
    <fo:table-cell padding="1mm" font-size="80%">
            <fo:block>製品 A</fo:block>
```

```
        </fo:table-cell>
    <fo:table-cell padding="1mm"
       text-align="'.'" axf:text-align-string="end">
            <fo:block font-variant="tabular-nums">
         800.00</fo:block>
    </fo:table-cell>...
  </fo:table-row>
  <fo:table-row background-color="cmyk(0,0,0,0.1)">
    <fo:table-cell ...>
      <fo:block>製品 B</fo:block>
    </fo:table-cell>
    ...
    </fo:table-row>
  ...
  </fo:table-body>
</fo:table>
```

When `text-align` property in a cell is <*string*>, `axf:text-string` specifies the overall alignment of all of the text that is aligned on <*string*>, assuming that <*string*> is a decimal point or similar.

Most cells are center-aligned vertically with `display-align="center"`, but the header cell with two lines of content specifies `display-align="justify"`.

When the content is aligned on the decimal point, the edges of the text becomes ragged and it becomes difficult to distinguish items next to each other, so a border is specified for each item. The only horizontal border is for header row, and the other rows have alternating background colors. `font-variant="tabular-nums"` explicitly specifies the glyphs designed to align the numbers.

製品名	処理 1 実行時間（秒）	処理 2 実行時間（秒）
製品 A	800.00	8.01
製品 B	10.112	220.3
製品 C	1000.1	1000.118

Figure 2.5-2　A table with numbers aligned on the decimal point

Section 2.6 Leader

You have considerable flexibility in specifying how leaders are aligned.

2.6.1 Leader alignment

AH XSL Formatter extends the values for `leader-alignment`.

Property	`leader-alignment`
Value	`none` | `reference-area` | ~~`page`~~ | **`start`** | **`center`** | **`end`**
Initial	`none`
Applies to	fo:leader
Inherited	yes

`start`, `center` and `end` align the leader, but only with `leader-pattern="dots"` or with `leader-pattern="use-content"` when the content is only text [Figure 2.6-1] (p.126).

Since the `page` value is not required, it is not supported by AH XSL Formatter.

Figure 2.6-1 `leader-alignment`

2.6.2 Justify the leader characters

You can align the edges of the leader itself when some conditions are met.

`leader-alignment` controls how the edges of the leader are aligned, but the dots or text spacing used by the leader remains the same.

Option Property	`justify-leader`
Value	`true` | `false`
Initial	`false`
Applies to	`<formatter-settings>`

If `justify-leader` is `true`, then the content of leaders for which `leader-alignment` is `none` and `leader-pattern` is `dots` or `use-content` is justified so that there are no spaces at either end [Figure 2.6-2] (p.127).

With `justify-leader="true"`, the space between characters is adjusted to eliminate the space at the ends. Because of this, leaders on different lines may not look like they are aligned.

Note

`justify-leader` affects the entire document, but it only affects leaders with `leader-alignment="none"`. For leaders on multiple consecutive lines, specify a value other than none for their `leader-alignment` properties.

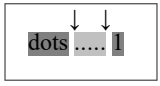


Figure 2.6-2 Justify the leader

2.6.3 Justify line containing leader

You can change the text alignment of just the line containing the leader.

Property	`axf:leader-expansion`
Value	`auto` | `force`
Initial	`auto`
Applies to	fo:block
Inherited	yes

When `axf:leader-expansion` is `force`, the line containing the leader is justified. This is useful, for example, when there is content on both ends of the leader and the length of the content varies, such as in a table of contents. It is also useful when the leader continues across two lines, or when the page numbers in an index entry continue across two lines.

Section 2.7 List

There is no specification that directly extends the list structure, but a more flexible list expression is possible by applying the extended specification.

2.7.1 Use case: Make a multi-column list

Span a multi-column list partially.

The entire page is composed of single column.

Only the list block is composed of multi column.

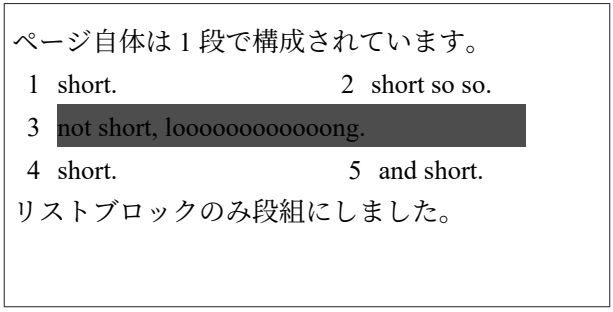


Figure 2.7-1 Make only the list block in multi columns

When describing lists without columns, there may be a large blank on the page depending on the length of the item. By combining the columns in the block container and the column spanning in the list block, create a column list according to the length of the list items [Figure 2.7-1] (p.128)

Make only the list block in multi columns

```
<!-- Specify columns for block container -->
<fo:block-container
  coumn-count="2" column-gap="5mm">
  <fo:list-block
    provisional-distance-between-starts="4mm"
    provisional-label-separation="2mm">
        ...
  </fo:list-block>
</fo:block-container>
```

The basic printing area is in single column, and a list with short list items is in 2 columns. Specify `column-count="2"` for `<fo:block-container>` and make `<fo:list-block>` as

its child element. The bullet start position is specified in consideration of the space between columns.

Span only a part of multi-column list

```
<!-- Multi-column list block -->
<fo:list-block
  span="all"
  ... >
  <fo:list-item>
    <fo:list-item-label
      start-indent="2mm"
      end-indent="label-end()">
      <fo:block>3</fo:block>
    </fo:list-item-label>
    <fo:list-item-body
      start-indent="body-start()"
      end-indent="1cm">
        <fo:block background-color="red">not short, ...</fo:block>
    </fo:list-item-body>
  </fo:list-item>
</fo:list-block>
<fo:list-block ...>
...
</fo:list-block>
```

If you want to span only a part of the multi-column list, terminate `<fo:list-block>` once and specify `span="all"` for the new `<fo:list-block>`, then a span-all column list will be generated.

When the span-all column list is finished, terminate this `<fo:list-block>` and start the column list without `span="all"` again.

Section 2.8 Graphics (color, image, background and math formula)

Various color expressions, image formats and control of drawing methods are supported.

2.8.1 Specify the color

Specify the color in various ways.

The range of colors that can be expressed depends on the color space in which you decide how to express the colors. Colors are represented on the display by additive mixing of Red, Green and Blue (RGB), while printing with ink is represented by subtractive mixing of Cyan, Magenta, Yellow and Keyplate (CMYK). In addition, spot colors may be specified as colors that cannot be expressed in a normal color space, such as special inks.

In the XSL 1.1 specification, the value of *<color>* is specified by RGB specification, sixteen basic color names, `system-color()` function and `rgb-icc()` function. AH XSL Formatter extends the values you can specify, such as CMYK, HSL and colors including transparency.

2.8.2 Specify by color name

The RGB color space can be specified by rgb() function, by #RRGGBB or by the color name. In AH XSL Formatter, the color names that can be specified have been extended.

Sixteen basic color names	aqua, black, blue, fuchsia, gray, green, lime, maroon, navy, olive, purple, red, silver, teal, white, yellow
131 extended color names, and rebeccapurple	aliceblue, antiquewhite, aquamarine, azure, beige, bisque, blanchedalmond, blueviolet, brown, burlywood, cadetblue, chartreuse, chocolate, coral, cornflowerblue, cornsilk, crimson, cyan, darkblue, darkcyan, darkgoldenrod, darkgray, darkgreen, darkgrey, darkkhaki, darkmagenta, darkolivegreen, darkorange, darkorchid, darkred, darksalmon, darkseagreen, darkslateblue, darkslategray, darkslategrey, darkturquoise, darkviolet, deeppink, deepskyblue, dimgray, dimgrey, dodgerblue, firebrick, floralwhite, forestgreen, gainsboro, ghostwhite, gold, goldenrod, greenyellow, grey, honeydew, hotpink, indianred, indigo, ivory, khaki, lavender, lavenderblush, lawngreen, lemonchiffon, lightblue, lightcoral, lightcyan, lightgoldenrodyellow, lightgray, lightgreen, lightgrey, lightpink, lightsalmon, lightseagreen, lightskyblue, lightslategray, lightslategrey, lightsteelblue, lightyellow, limegreen, linen, magenta, mediumaquamarine, mediumblue, mediumorchid, mediumpurple, mediumseagreen, mediumslateblue, mediumspringgreen, mediumturquoise, mediumvioletred, midnightblue, mintcream, mistyrose, moccasin, navajowhite, oldlace, olivedrab, orange, orangered, orchid, palegoldenrod, palegreen, paleturquoise, palevioletred, papayawhip, peachpuff, peru, pink, plum, powderblue, rosybrown, royalblue, saddlebrown, salmon, sandybrown, seagreen, seashell, sienna, skyblue, slateblue, slategray, slategrey, snow, springgreen, steelblue, tan, thistle, tomato, turquoise, violet, wheat, whitesmoke, yellowgreen, rebeccapurple
28 system color names	ActiveBorder, ActiveCaption, AppWorkspace, Background, ButtonFace, ButtonHighlight, ButtonShadow, ButtonText, CaptionText, GrayText, Highlight, HighlightText, InactiveBorder, InactiveCaption, InactiveCaptionText, InfoBackground, InfoText, Menu,MenuText, ScrollBar, ThreeDDarkShadow, ThreeDFace, ThreeDHighlight, ThreeDLightShadow, ThreeDShadow, Window, WindowFrame, WindowText

Sixteen basic color names: The colors specified in the XSL 1.1 specification.

131 extended color names : It is a color that is “Extended Color” in CSS3 Color. Extended Color also includes the equivalent of sixteen basic color names.

28 system color names: The color name of CSS2 that has become Deprecated. The color name specification belongs to the RGB color space.

2.8.3 Specify the color with CMYK

Specify the color with the value of CMYK, which is the color space used for printing purpose.

Function Name	`cmyk()`
Argument Value	<*C*>, <*M*>, <*Y*>, <*K*>

Specify 0.0 to 1.0 or a percentage.

As a shorthand, you can treat `k100` in the same way as specifying `cmyk(0, 0, 0, 100)`.

`cmyk(`<*C*>, <*M*>, <*Y*>, <*K*>`)` is equivalent to `rgb-icc(#CMYK,` <*C*>, <*M*>, <*Y*>, <*K*>`)`.

2.8.4 Specify with HSL

In addition to RGB, HSL color space can be specified.

In AH XSL Formatter, you can specify colors using the HSL color space that expresses colors with Hue, Saturation and Lightness.

Function Name	`hsl()`
Argument Value	<*H*>, <*S*>, <*L*>

Specify with `hsl()` function.

Specify <*H*> as a number without units, and <*S*> and <*L*> percentage or a range from 0.0 to 1.0.

2.8.5 Specify a color that includes a transparent color

Transparency can be specified in addition to each color specification method.

Function Name	`rgba()`
Argument Value	[<*R*>, <*G*>, <*B*> | <*string*>], <*A*>

Function Name	`hsla()`
Argument Value	<*H*>, <*S*>, <*L*>, <*A*>

Function Name	`cmyka()`
Argument Value	<*C*>, <*M*>, <*Y*>, <*K*>, <*A*>

Specify transparency <*A*> in addition to the arguments to the functions `rgb()`, `hsl()` and `cmyk()` that do not include transparency. The value of <*A*> ranges from 0 to 1.

In `rgba()`, you can also specify by color name like `rgba(green,0.4)`.

Transparency may not be available depending on the PDF output version and profile specifications.

Option Property	transparency-color-space
Value	None | DeviceRGB | DeviceCMYK
Initial	DeviceRGB
Applies to	<pdf-settings>

Specify the color space for transparency processing in PDF output in the `transparency-color-space` property in the option setting.

2.8.6 Control transparent conversion when outputting PDF

>Depending on the output format and PDF type, colors and images including transparency may be converted without being output as they are. It's possible to controls this operation.

Option Property	transparency
Value	true | false
Initial	true
Applies to	<pdf-settings>

When outputting to PDF, specify the color including transparency and whether to convert the image to non-transparency in the `transparency` property in the option setting.

If `true` is specified, transparency is retained, and if `false` is specified, converted to non-transparency.

However, if the PDF to be embedded with `<fo:external-graphic>` etc. is transparent, this cannot be converted.

2.8.7 Convert color space when outputting PDF

PDF keeps a color space inside. When outputting PDF, the color specified by FO is converted to the specified one.

Option Property	color-conversion
Value	none | black | gray | all-gray | all-cmyk
Initial	none
Applies to	<pdf-settings>

You can specify how to convert the color of the color space specified by RGB or CMYK to DeviceGray or DeviceCMYK in the `color-conversion` property in the option setting.

`black` outputs RGB black, and gray outputs colors corresponding to RGB gray tones as DeviceGray. `all-gray` converts all colors to DeviceGray and `all-cmyk` converts all RGB to CMYK.

This conversion is also effective for images rendered using the original drawing engine (SVG, CGM, MahML, EMF, WMF), but other images are converted only when `all-*` is specified.

When the output PDF format is PDF/X-1a, it is considered `all-gray` or `all-cmyk`.

Note

Embedded PDFs are not subject to conversion.

2.8.8 Detect scale fitting of the image

When it is not desirable to adjust the size of the image and output it, it is detected at the time of formatting and report an error.

Mainly for external image placement, when `scale-*to-fit` is specified for the `content-width` or `content-height` property, scale the image if it does not match the specified area size.

Option Property	`issue-scale-to-fit`
Value	`true` | `false`
Initial	`false`
Applies to	`<formatter-settings>`

Specify the control over detecting scale changes in the `issue-scale-to-fit` property in the option setting. If `true` is specified, level 1 will be reported when a scale change occurs.

2.8.9 Specify the resolution of the rasterized vector image

In the Windows version, some vector images that cannot be stored directly in PDF can be converted to raster images and stored.

Option Property	`rasterize-resolution`
Value	*<integer>*
Initial	`108`
Applies to	`<pdf-settings>`

In the `rasterize-resolution`s property in the option setting, you can specify the resolution of the raster image at the time of conversion with a numerical value from 70 to 500. The specified number is a dpi value, but since it is *<integer>*, do not add a unit when specifying it.

2.8.10 Control the conversion of image formats that cannot be stored directly in PDF

AH XSL Formatter converts Image formats that PDF does not support to the supported formats and stored at the time of output. It's possible to control the conversion of some of the input image formats.

For details on the image formats and conversion settings that can be stored in PDF, refer to Manual

Option Property	`color-compression`, `grayscale-compression`
Value	`auto` | `zlib` | `jpeg` | `jpeg2000` | `keeplzw` | `auto2k` | `keeplzw2k`
Initial	`auto`
Applies to	`<pdf-settings>`

Option Property	`monochrome-compression`
Value	`ccitt4` | `ccitt3` | `runlength` | `zlib` | `none`
Initial	`ccitt4`
Applies to	`<pdf-settings>`

When an image format that cannot be stored directly in PDF is embedded, the image is converted to a bitmap format compatible with PDF and stored into PDF. At that time, the bitmap format compression method can be controlled by the option setting.

Specify the compression method with the `color-compression` property for color images, the `grayscale-compression` property for grayscale images, and the `monochrome-compression` property for black-and-white images.

When `auto` is specified for the `color-compression` and `grayscale-compression` properties, image data is created according to the values specified for each `*-jpeg-quality`. At this time, compare the sizes of this JPEG and ZLIB compression and choose the smaller one.

Specify the image quality when converting to JPEG

The image quality can be specified when converting an image format that cannot be stored directly to JPEG or JPEG2000 format. It does not affect images that are originally JPEG or JPEG2000.

Option Property	`color-jpeg-quality`, `grayscale-jpeg-quality`
Value	*<number>*
Initial	`80`
Applies to	`<pdf-settings>`

In the option settings, specify the image quality to the `color-jpeg-quality` property for color images and the `grayscale-jpeg-quality` property for grayscale images.

When the converted image is stored as JPEG2000, the value is converted to compression ratio.

2.8.11 Control the pass-through of image embedding

Controls whether processing is done when embedding an image.

Option Property	`pass-through`
Value	`[ all || -gif || -tiff || -png || -jpeg || -jpeg2000 || -jbig2] | [ gif || tiff || png || jpeg || jpeg2000 || jbig2] | none`
Initial	`all`
Applies to	`<pdf-settings>`

When embedding an image, if it can be embedded as it is, the pass-through to be embedded can be controlled by specifying the image type with the `pass-through` property in the option setting. You can list the image types you want to pass-through separated by spaces, or add "-" to the image types you want to exclude, such as all `-gif`. When down sampling is specified at the same time, down sampling takes precedence. Even if `pass-through` is specified, pass-through will not be performed if it is not possible due to circumstances.

Note

For example, it is useful when you want to speed up processing in formatting that contains a large number of images.

2.8.12 Control the resolution of images and background

Specify the resolution of the image and background image.

Property	`axf:image-resolution`
Value	`normal` | [`from-image` || <*dpi*>]
Initial	`from-image`
Applies to	fo:external-graphic, fo:instream-foreign-object

Property	`axf:background-image-resolution`
Value	[`normal` | [`from-image` || <*dpi*>]]#
Initial	`from-image`
Applies to	fo:simple-page-master, fo:page-sequence, all formatting objects

>Specify the image resolution at the time of output in the `axf:image-resolution` property.

Specify `normal` to match the system default value specified by the `pxpi` property in the option setting, specify `from-image` to use the resolution of the image, and use <*dpi*> to specify the resolution.

If `axf:image-resolution="from-image <dpi>"` is specified, it will follow the resolution if image has a resolution, and if it does not exist, fallback to <*dpi*>. If there is no <*dpi*> to fallback, it will fallback to `normal`.

Vector images such as SVG are used to convert from unitless numbers.

The background image resolution can also be controlled with the `axf:background-image-resolution` property. When there are multiple background images, if you specify multiple values, they will be applied in the same order as the background image settings.

2.8.13 Image anti-aliasing

You can control whether anti-aliasing is performed on an image when it is displayed on the screen.

Property	axf:image-smoothing
Value	auto | true | false
Initial	auto
Applies to	fo:external-graphic, fo:instream-foreign-object

axf:image-smoothing specifies whether to perform anti-aliasing on an image when displaying it on the screen.

If auto, it follows the GUI settings. If true, anti-aliasing is performed, and if false is specified, no processing is performed.

2.8.14 Page master and page sequence background

AH XSL Formatter has been extended to allow you to specify a background for the page master and page sequence.

The background can be specified for the entire page and can switch the background for each page sequence.

If any of the background* properties are specified on the page master, the background setting in the page sequence is ignored.

Note

The background specified for the page master is also drawn in the bleed area added by the axf:bleed property.

2.8.15 Raster images downsampling

You can control whether and how images are downsampled.

Option Property	color-downsampling, grayscale-downsampling, monochrome-downsampling
Value	none | average | bicubic | subsampling
Initial	none
Applies to	<pdf-settings>

Option Property	`color-downsampling-above-dpi`, `grayscale-downsampling-above-dpi`
Value	*<number>*
Initial	`450`
Applies to	`<pdf-settings>`

Option Property	`color-downsampling-target-dpi`, `grayscale-downsampling-target-dpi`
Value	*<number>*
Initial	`300`
Applies to	`<pdf-settings>`

Option Property	`monochrome-downsampling-above-dpi`
Value	*<number>*
Initial	`1800`
Applies to	`<pdf-settings>`

Option Property	`monochrome-downsampling-target-dpi`
Value	*<number>*
Initial	`1200`
Applies to	`<pdf-settings>`

The `*-downsampling` options specify the downsampling method for raster images. `average` uses the bilinear method; `bicubic` uses the bicubic method; and `subsampling` uses the nearest neighbor method.

The `*-downsampling-above-dpi` options specify the lowest resolution to downsample, and `*-downsampling-target-dpi` options specify the resulting resolution after downsampling.

Color, grayscale images and black-and-white images (`monochrome-*`) have very different initial lower limits and target resolutions.

2.8.16 SVG image size

You can specify how the display size of embedded SVG images is determined.

When embedding an SVG image and `width` and `height` are not specified on the SVG root element, AH XSL Formatter uses `width="100%"` and `height="100%"` as per the SVG specification. You can change this operation.

Option Property	`responsive-svg-size`
Value	`reference` | `viewBox`
Initial	`reference`
Applies to	`<formatter-settings>`

When the `responsive-svg-size viewBox`, if `viewBox` is present on the SVG root element but not `width` or `height`, then `viewBox` is used as the size.

2.8.17 Background gradient

The `background-image` value can include functions that generate a linear or circular gradient.

Function Name	`linear-gradient()`
Argument Value	[[<*angle*> | `to` <*side-or-corner*>,]] ? <*color-stop*>#{2,}

Function Name	`radial-gradient()`
Argument Value	[[<*shape*> || <*size*>] | [`at` <*position*>,]] ? | `at` <*position*>,] ?] <*color-stop*>#{2,}

Value Name	Value Composition
<*color-stop*>	<*color*>[<*percentage*> | <*length*>]
<*side-or-corner*>	[`left` | `right`] || [`top` | `bottom`]
<*position*>	[<*percentage*> | <*length*> | `left` | `center` | `right`] [<*percentage*> | <*length*> | `top` | `center` | `bottom`]] ? | [[`left` | `center` | `right`] || [`top` | `center` | `bottom`]] | `inherit`
<*shape*>	`circle` | `ellipse`
<*size*>	`closest-side` | `farthest-side` | `closest-corner` | `farthest-corner` | [<*length*> | [<*length*> | <*percentage*>]{2}]

`linear-gradient()` specifies a linear gradient from a specified direction. <*angle*> specifies the gradient direction. <*side-or-corner*> is a keyword or keywords that maps to an

angle. For example, `to right` indicates the direction from left to right, which corresponds to an *<angle>* of `90deg`.

<color-stop> is a color on the gradient and, optionally, a length or percentage for the color's position along the gradient line.

`radial-gradient()` is a gradient drawn outwards from a point. *<position>* determines the placement of this origin, and *<size>* controls the end of the gradient. `closest-*` marks the edge or corner closest to the origin as the end of the gradient, and `farthest-*` marks the edge or corner farthest from the origin as the end.

2.8.18 Repeating gradient

You can also specify gradients that repeat.

Function Name	`repeating-linear-gradient()`
Argument Value	[[*<angle>* | `to` *<side-or-corner>*],]] ? *<color-stop>*#

Function Name	`repeating-radial-gradient()`
Argument Value	[[*<shape>* || *<size>*] | [`at` *<position>*,]] ? | `at` *<position>*,] ?] *<color-stop>*#{2,}

`repeating-*-gradient()` are used in the same way as normal gradients. However, their color stops are repeated.

If *<color-stop>* is 100 %, they operate like normal `*-gradient()` because there is no room in the area to repeat the gradient.

2.8.19 EPS images

AH XSL Formatter does not process EPS images. However, it can pass the image to an external processor and include the converted image in the output.

Note

It is recommended to convert EPS images to PDF with an external tool and then embed that PDF.

Option Property	`EPS-processor`
Value	`none` | `distiller` | `ghostscript`
Initial	`none`
Applies to	`<pdf-settings>`

When the `EPS-processor` option is `distiller`, Adobe Distiller is used as the external processor, and when `ghostscript` is specified, GhostScript is used. The external processor must be installed. `EPS-processor="distiller"` is only valid for the Windows version, which uses acrodist.exe. Make sure that it is on the path. EPS images have the following restrictions when using acrodist.exe:

- PS-Adobe-2.0 or higher
- `%%BeginProlog` and `%%EndProlog` should be present

Distiller options

Option Property	`joboptions`
Value	*<string>*
Initial	empty string
Applies to	`<pdf-settings>`

The `joboptions` option allows you to specify the path of the Adobe PDF Settings File used by Distiller when converting EPS to PDF.

A relative path is resolved relative to the target EPS file. If the joboptions value is a relative path, the result will be undefined, so specify it as an absolute path if possible.

This setting is valid only for the Windows version.

GhostScript path

The GhostScript that is used when `EPS-processor="ghostscript"` is gswin32c.exe or gswin64c.exe in the Windows version, and gs in other versions. Because GhostScript is started by `fork()`, there is no problem with the GPL license.

Option Property	`ghostscript`
Value	*<string>*
Initial	*empty*
Applies to	`<pdf-settings>`

If the `ghostscript` option contains the full path to GhostScript, it will work even when GhostScript is not on the PATH environment variable.

GhostScript arguments

The default parameters when processing EPS with GhostScript:

```
  -dPDFSETTINGS=/printer
-dNOPAUSE
-dBATCH
-dSAFER
-sDEVICE=pdfwrite
-dDEVICEWIDTHPOINTS=%width
-dDEVICEHEIGHTPOINTS=%height
-dEPSFitPage
-dCompatibilityLevel=1.3
-dAutoRotatePages=/None
-q
-sOutputFile=%temp
-c
.setpdfwrite
-f
%eps
```

AH XSL Formatter inserts values for %width, %height, %temp and %eps.

Option Property	`gs-options`
Value	<*string*>
Initial	*empty*
Applies to	`<pdf-settings>`

The `gs-options` option value completely replaces the parameters provided to GhostScript.

The parameter delimiter is <control>(U+000A), but because it is not preserved in XML processing, you must use `
` or `
` numeric character references.

2.8.20 Example: Text color of the entire document

The color specified for the root element is used as the default color for virtually the entire document.

The default text color for AH XSL Formatter is RGB `black`. If you want to specify the default text color in FO, specify color in the `<fo:root>` element and it will be inherited as the color for all text.

For example, if you specify `<fo:root color="k100">`, text that does not specify a `color` setting will be drawn with `k100`.

```
<fo:root ... color="k100">...</fo:root>
```

2.8.21 Example: Display image as a circle

There are two ways to use extension properties to cut out an image and display it as a circle. In both cases, the image is specified as the background image.

Border as circle

```
<fo:block-container
  width="5cm" height="5cm"
  background-image="url(image.jpg)"
  axf:border-radius="50%">
 <fo:block>...</fo:block>
</fo:block-container>
```

With `axf:border-radius="50%"`, that is, when all radii are equal, the area with the background image looks circular. The `<fo:block-container>` has an explicit size.

If you want to blur the edges of the background image, you can specify `inset` for the `axf:box-shadow` property.

Overlay radial gradient

```
<fo:block-container width="5cm" height="5cm"
  background-image="radial-gradient(circle closest-side,
    rgba(255,255,255,0.0), rgba(255,255,255,0.0), rgba(255,255,255,
1.0)),
    url(image.jpg)" >
  <fo:block>...</fo:block>
</fo:block-container>
```

The image is displayed in a circle by specifying `radial-gradient()`, which uses a transparent inner color over the image. This is an easy way to blur the edges.

Section 2.9 Float

You have more control over placing floats than provided by the XSL 1.1 specification.

[Figure 2.9-1] (p.146) shows basic float placement. The float is placed based on the place (anchor) where the `<fo:float>` occurs in the flow.

`axf:float-reference` specifies the reference area in which to place the float.

`axf:float-move` controls the movement of float and its anchor when there is no room to place the float in the area.

`axf:float-x` and `axf:float-y` position the float in the reference area.

`axf:float-wrap` specifies whether text can wrap the placed float.

Property	`axf:float`
Value	*<float-x>* || *<float-y>* || *<float-wrap>* || *<float-reference>* || *<float-move>*
Initial	`none`
Applies to	fo:float / floated elements

`axf:float` is a shorthand for all the extension properties for float placement. Each value is described in a separate section.

Margin and wrapping conditions cannot be specified with `axf:float`.

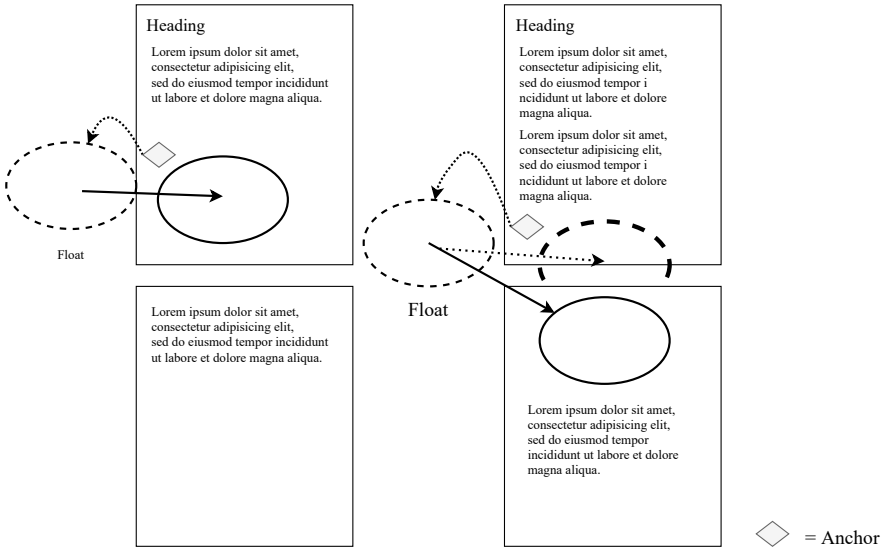


Figure 2.9-1 When sufficient space is available (left) and when not available (right)

2.9.1 Reference area

You can specify the reference area in which to place the float.

Property	`axf:float-reference`
Value	`auto` | `normal` | `page` | `column` | `multicol`
Initial	`auto`
Applies to	fo:float

`axf:float-reference` specifies the reference area for the float [Figure 2.9-2] (p.147).

`axf:float-reference`	**Definition**
`auto`, `normal`	Current reference area.
`page`	Page (body region).
`column`	Column.
`multicol`	The reference area is all columns. The float can span fewer than all of the columns.

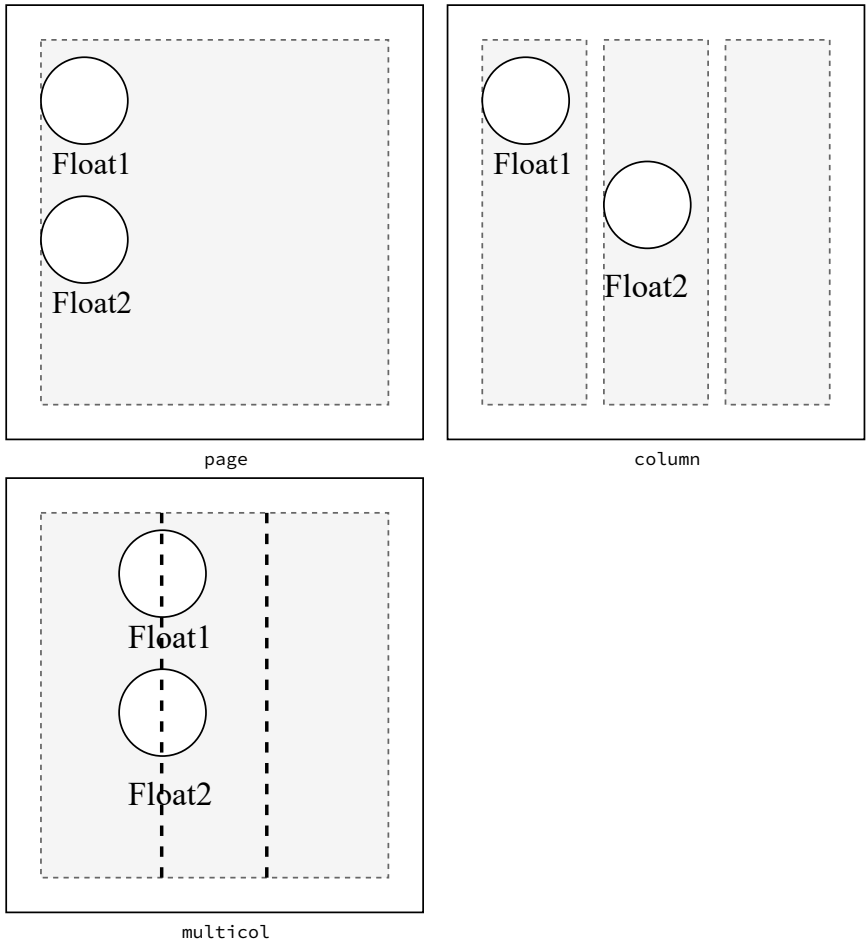


Figure 2.9-2 `axf:float-reference` with `axf:float-x="start"`

Since the reference area for column is a `column`, for a three-column area, the candidate areas for placing the float are the first, second and third column, in sequence.

When `multicol`, the placement position is the columns in sequence, but the float can span fewer than all of the columns. By using *gr* for the float width, you can, for example, specify the float as spanning two of the three columns. `auto` and `normal` use the current reference area.

Note

When `multicol` is specified, the reference area is the same size as when `page` is specified, so when specifying `axf:float-min-wrap-x` and so on, consider the page as the reference area.

2.9.2 Float and anchor positions

You can control the anchor that is the point of origin for placing the float.

The anchor area is at the place where the `<fo:float>` occurs in the markup, and the float is placed in the reference area containing the anchor if there is sufficient space.

Property	`axf:float-move`
Value	`auto` | `next` | `auto-next` | `auto-move` | `keep` | `keep-float` | <*integer*>
Initial	`auto`
Applies to	fo:float / floated elements

`axf:float-move` lets you specify whether to move the float to the next reference area or move the anchor area when there is not enough space.

`axf:float-move`	**Definition**
`auto`	Same as keep when `axf:float-y` is `none`; otherwise, same as `auto-next`.
`next`	Next reference area.
`auto-next`	Send the float to the next reference area If there is not enough space in the current reference area.
`auto-move`	Send the float to the next reference area If there is not enough space in the current reference area. Alternatively, do not move the float and send the anchor and surrounding text to the next reference area.
`keep`	When there is not enough space, send both the float and its anchor to the next reference area. Automatically set `keep-with-next="always"` on the anchor area.
`keep-float`	When there is not enough space, send both the float and its anchor to the next reference area.
<*integer*>	Place the float on the specified page. If the number is more than the total number of pages, empty pages will be inserted as necessary. When `axf:float-y="none"` is specified, this is regarded as `auto`.

2.9.3 Position

You can control the position of the float within its reference area.

`axf:float-x` specifies the horizontal position (vertical position in vertical writing-mode), and `axf:float-y` specifies the vertical position (horizontal position in vertical writing-mode). The allowed values differ depending on the writing mode.

Property	`axf:float-x`
Value	`none` | `start` | `end` | `left` | `right` | `top` | `bottom` | `center` | `inside` | `outside` | `alternate` | `column-outside`
Initial	`none`
Applies to	fo:float / floated elements

`axf:float-x`	Definition	Horizontal text	Vertical text
`none`	Do not float horizontally (vertically in vertical text).	✓	✓
`start`	Position on the start side. Same as `left` in left-to-right horizontal text.	✓	✓
`end`	Position on the end side. Same as `right` in left-to-right horizontal text.	✓	✓
`left`	Position on the left side.	✓	-
`right`	Position on the right side.	✓	-
`top`	Position on the upper side.	-	✓
`bottom`	Position on the lower side.	-	✓
`center`	Position in the horizontal center (vertical center in vertical text).	✓	✓

axf:float-x	Definition	Horizontal text	Vertical text
inside	Position on the inside.	✓	-
outside	Position on the outside.	✓	-
alternate	Treated as end if the first of multiple columns, start if the last column and center for other columns.	✓	✓
column-outside	Treated as end for the first of multiple columns and start for the others.	✓	✓

Property	axf:float-y
Value	none | start | end | left | right | top | bottom | center | inside | outside
Initial	none
Applies to	fo:float / floated elements

axf:float-y	Definition	Horizontal text	Vertical text
none	Do not float vertically (horizontally in vertical text).	✓	✓
before	Position on the before side. Same as top in left-to-right horizontal text.	✓	✓
after	Position on the after side. Same as bottom in left-to-right horizontal text.	✓	✓
left	Position on the left side.	-	✓
right	Position on the right side.	-	✓
top	Position on the upper side.	✓	-
bottom	Position on the lower side.	✓	-
center	Position in the vertical center (horizontal center in vertical text).	✓	✓
inside	Position on the inside.	✓	✓
outside	Position on the outside.	✓	✓

When axf:float and axf:float-y are both none, floating is disabled. The contents will be placed at the current position.

2.9.4 Text wrapping

You have fine control over whether and how text can wrap around the float.

Property	axf:float-wrap
Value	auto | wrap | skip
Initial	auto
Applies to	fo:float / floated elements

axf:float-wrap specifies whether text can wrap.

When skip, text does not wrap, and when wrap, text wraps. However, if the float is in the center of the column and there is enough width for the text to wrap on both sides, axf:float-wrap will be treated as skip, even if wrap is specified, and the text will skip the float. When axf:float-x is none, auto is treated as skip, otherwise it is treated as wrap.

Wrapping, alignment and available space

Property	axf:float-min-wrap-x,axf:float-min-wrap-y
Value	normal | <*length*> | <*percentage*>
Initial	normal
Applies to	fo:float / floated elements
Percentages	refer to the size of containing block

Property	axf:float-centering-x,axf:float-centering-y
Value	none | auto | <*length*> | <*percentage*>
Initial	none
Applies to	fo:float / floated elements
Percentages	refer to the size of containing block

Large figures and tables sometimes allow only one or two characters of wrapped text per line, which makes the content difficult to read.

axf:float-min-wrap-x and axf:float-min-wrap-y specify the minimum width for text to be allowed to wrap the float. normal is the same as specifying 0pt.

axf:float-centering-x and axf:float-centering-y specify whether to center the float when there is insufficient width or height to allow text to wrap.

When *<length>* or *<percentage>* is specified, the float will be centered if the width for the text to wrap is less than the specified value. `auto` centers the float when the available space is less than the corresponding `axf:float-min-wrap-*` value.

2.9.5 Margin

You can specify the margin between the float and text as well as the margin between floats.

Property	`axf:float-margin-x`, `axf:float-margin-y`
Value	[*<length>* | *<percentage>*][*<length>* | *<percentage>*]] ?
Initial	`0pt`
Applies to	fo:float / floated elements
Percentages	refer to the size of containing block

`axf:float-margin-x` and `axf:float-margin-y` specify the margin between the float and its wrapping text.

`axf:float-margin-x` sets the margins for the inline progression direction: when two values are specified, the first is for the start side and the second is for the end side.

`axf:float-margin-y` sets the margins for the block progression direction: when two values are specified, the first is for the before side and the second is for the after side.

Margin between floats

You can specify the margin between adjacent floats independently from the margin between the float and other objects [Figure 2.9-3] (p.153).

Property	`axf:float-float-margin-x`, `axf:float-float-margin-y`
Value	`auto` | [[*<length>* | *<percentage>*][*<length>* | *<percentage>*]] ?]
Initial	`auto`
Applies to	fo:float / floated elements
Percentages	refer to the size of containing block

`axf:float-float-margin-x` and `axf:float-float-margin-y` specify margins between adjacent floats in the X and Y directions, respectively. The value cannot exceed the corresponding `axf:float-margin-x` or `axf:float-margin-y` value.

When `auto`, the `axf:float-margin-x` or `axf:float-margin-y` value is used.

[Figure 2.9-3] (p.153), the value of the `axf:float-float-margin-y` property specified on B is valid.

When adjacent floats both specify the margin between them, only one value is valid.

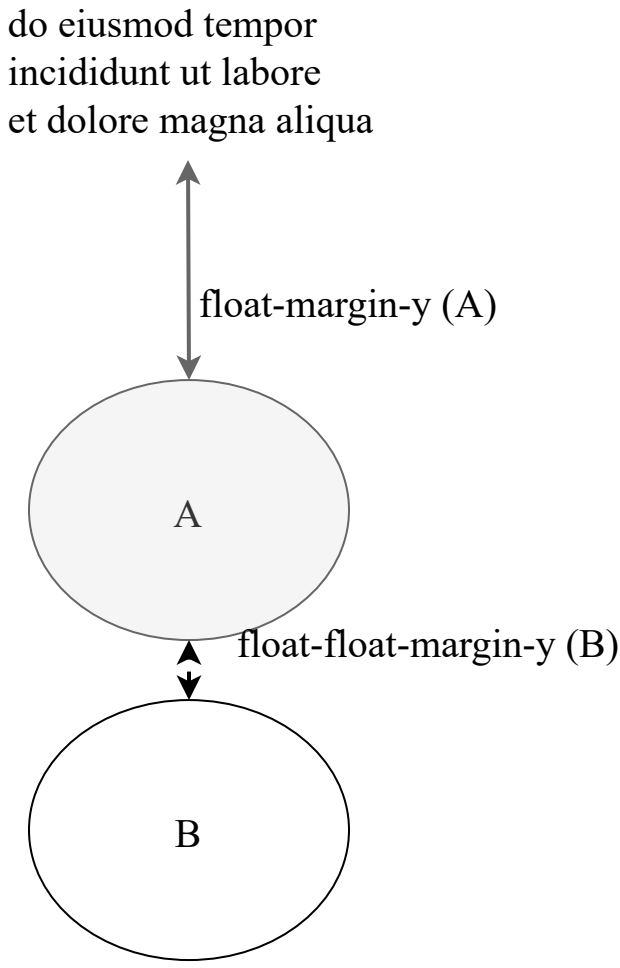


Figure 2.9-3 float-margin and float-float-margin

2.9.6 Float offset position

You can specify the offset from the position where the float is placed.

Property	`axf:float-offset-x`, `axf:float-offset-y`
Value	*<length>* | *<percentage>*
Initial	`0pt`
Applies to	fo:float / floated elements
Percentages	refer to the size of containing block minus the size of float

`axf:float-offset-x` specifies an X offset away from the specified position of the float. If positioned on the start side, the offset is towards the end side, and if positioned on the end side, the offset is towards the start side.

`axf:float-offset-y` specifies a Y offset away from the specified position of the float. If positioned on the before side, the offset is towards the after side, and if positioned on the after side, the offset is towards the before side.

Section 2.10 Footnotes and Sidenotes

You can control how the notes are placed and how they are numbered.

2.10.1 Numbers and symbols

You can restart footnote numbering at page and column breaks.

Element	<axf:footnote-number>
Property	id
Parent	descendants of fo:footnote's child fo:inline

Element	<axf:footnote-number-citation>
Property	ref-id
Parent	decsendants of fo:footnote-body

Reuse the footnote number using the <axf:footnote-number-citation> element. Specify the id of an <axf:footnote-number> as the ref-id property of <axf:footnote-number-citation>. The <axf:footnote-number> must be a descendant of the same <fo:footnote>.

Generate and display footnote numbers with the <axf:footnote-number> element.

Property	axf:footnote-number-initial
Value	auto | *<number>*
Initial	auto
Applies to	fo:page-sequence

Property	axf:footnote-number-reset
Value	auto | none | page | odd-page | even-page | column
Initial	auto
Applies to	fo:page-sequence

Specify the initial footnote number in the axf:footnote-number-initial property and the initialization context in the axf:footnote-number-reset property. When axf:footnote-number-initial="auto" is specified, footnote numbering is carried over from the previous page sequence. In the first page sequence, it is set to 1.

axf:footnote-number-reset	Definition
auto	Carry over the specification from the previous page sequence. In the first page sequence, use none.
none	Do not reset.
page	Reset at each page break.
odd-page	Reset at a page break to an odd page.
even-page	Reset at a page break to an even page.
column	Reset at each column break.

When the footnote number is reset as specified by axf:footnote-number-reset, its value is as specified axf:footnote-number-initial.

Use even-page to reset the footnote number for each pair of facing pages in horizontal layout, and use odd-page for facing pages in vertical layout.

If you want, for example, to restart footnote numbers for each chapter, each chapter can start a new page sequence and you can specify axf:footnote-number initial="1" on each.

Note

The footnote area itself can contain a mixture of sidenotes and footnotes, but since the axf:footnote-number-* properties are specified on <fo:page-sequence>, numbers generated using <axf:footnote-number> are consistent within the page sequence.

Footnote number format

Property	axf:footnote-number-format
Value	*<string>*
Initial	1
Applies to	fo:page-sequence

axf:footnote-number-format specifies how to format the footnote number. The allowed formats are the same as for the format property.

2.10.2 Footnote position

Specifies the position of notes, such as sidenotes and footnotes, within the page or column.

In the XSL 1.1 specification, an <fo:footnote> spans all columns. AH XSL Formatter allows fine control of footnote placement.

Property	axf:footnote-position
Value	page | odd-page | even-page | start | end | inside | outside | column | start-column | end-column | inside-column | outside-column
Initial	page
Applies to	fo:region-body, fo:footnote

axf:footnote-position specifies where on the page to place <fo:footnote> notes [Figure 2.10-1] (p.159).

odd-page and even-page can only be specified on <fo:region-body>.

Footnote in horizontal layout

axf:footnote-position	Definition in horizontal layout
page	Placed on the after side in the body section of each page. An XSL 1.1 footnote.
odd-page	Placed on the after side in the body section of odd pages.
even-page	Placed on the after side in the body section of even pages.

Sidenote in horizontal layout

A sidenote in horizontal layout is placed in the region-start or region-end [Figure 2.10-2] (p.160).

If a sidenote does not fit in the area, it will not be sent to the next area and it will overflow.

axf:footnote-position	Definition in horizontal layout
start	Placed in the region-start of each page.
end	Placed in the region-end of each page.
inside	Placed in the region-end of even pages and in the region-start of odd pages.
outside	Placed in the region-start of even pages and in the region-end of odd pages.

inside specifies a sidenote closer to the inside, and outside specifies a sidenote closer to the outside. outside is most conventional for sidenotes.

Footnote in columns

You can allow footnotes is every column, the first column, the last column, the inside column of the page or outside column [Figure 2.10-3] (p.161).

*-column can be specified only on <fo:region-body>. Also, when these are specified, column cannot be specified for axf:footnote-position on <fo:footnote>.

Specifying start, end, inside and outside generates a sidenote irrespective of the number of columns. Also, specifying *-column when there is a single column treats the footnote is if it is the corresponding sidenote.

axf:footnote-position	Definition in columns
page	Footnotes span all columns (initial value)
column	Footnotes in any column
start-column	Footnotes placed in the start-side column
end-column	Footnotes placed in the end-side column
inside-column	Footnotes placed in the inside column
outside-column	Footnotes placed in the outside column

Note in vertical layout

Specifies the position of notes when writing-mode="tb-rl" is specified [Figure 2.10-4] (p.162).

axf:footnote-position	Definition in vertical layout
page	Placed on the after side in the body section of each page.
odd-page	Placed on the after side in the body section of odd pages.
even-page	Placed on the after side in the body section of even pages.
start	Placed in the region-start of each page. Headnote in vertical layout.
end	Placed in the region-end of each page. Footnote in vertical layout.

In AH XSL Formatter, when `writing-mode="tb-rl"` is specified for a page, the initial value of note position is `axf:footnote-position="odd-page"`, that is, only at the edge of odd pages.

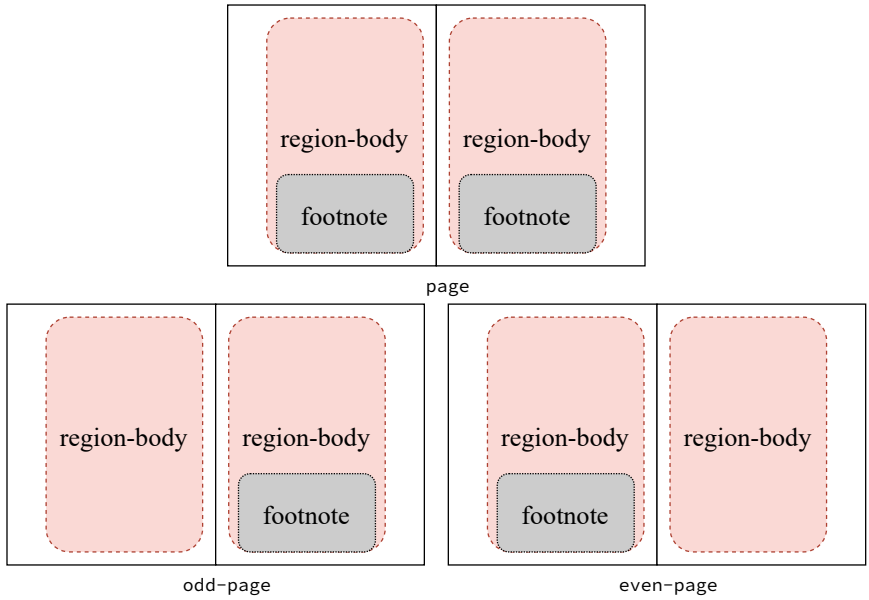


Figure 2.10-1 Notes in horizontal layout

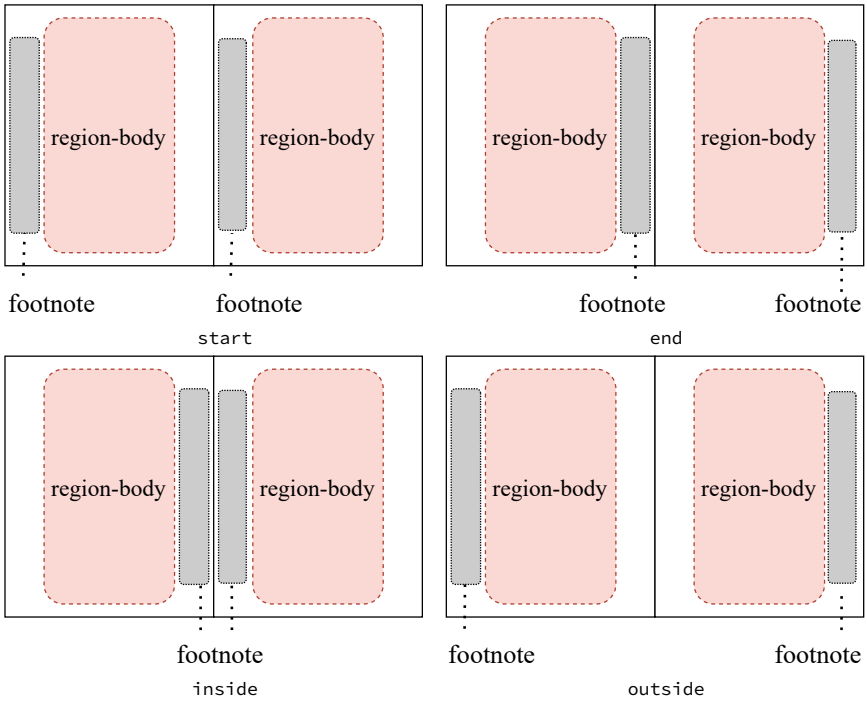


Figure 2.10-2 Sidenotes in horizontal layout

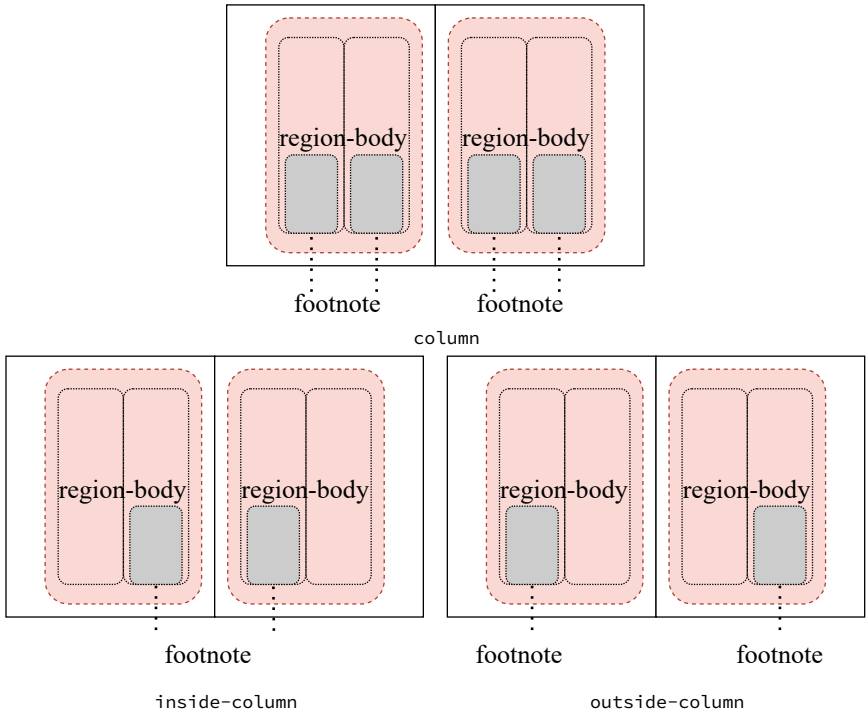


Figure 2.10-3 Notes in columns

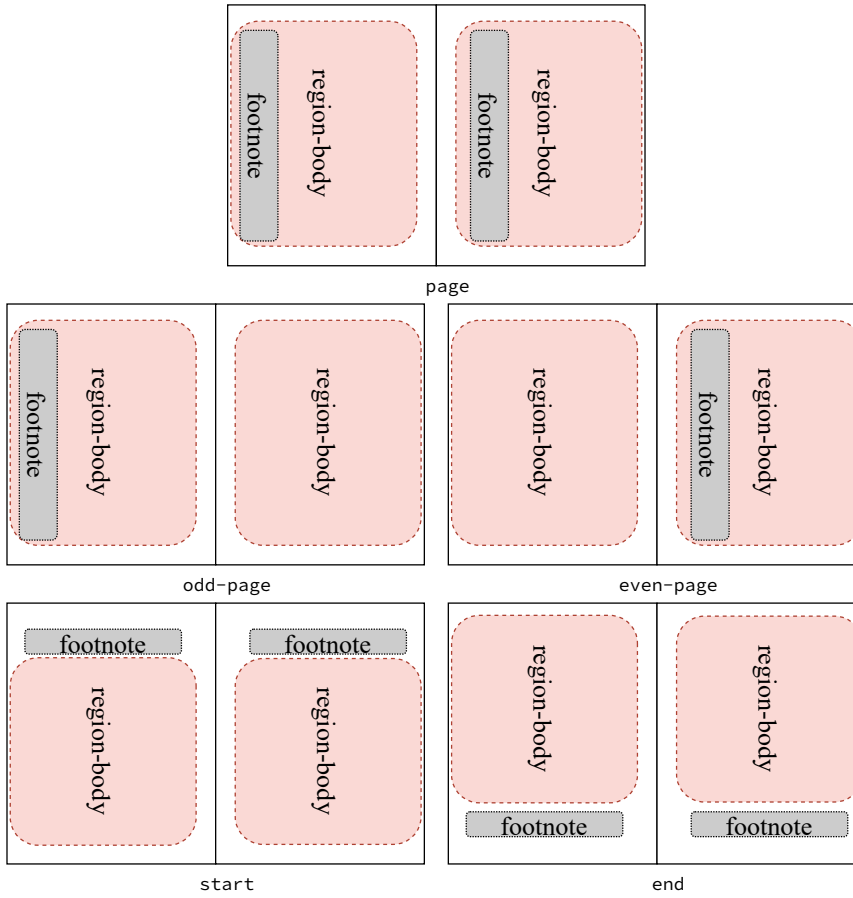


Figure 2.10-4 Notes in vertical layout

2.10.3 Footnote and sidenote alignment

Specifies the alignment of notes.

Property	`axf:footnote-align`
Value	`auto` | `before` | `after`
Initial	`auto`
Applies to	fo:region-body, fo:footnote

The effect depends on whether the note is a footnote (headnote) or sidenote.

axf:footnote-align	Note position	Definition
auto	Sidenote	Align to the anchor position for single column, and to the before side for two or more columns.
auto	Footnote	Normal position at the page foot.
before	Sidenote	Align to the before side.
before	Footnote	Placed just after the character string on the page.
after	Sidenote	Align to the after side.
after	Footnote	The end of the page. No change from the normal position.

2.10.4 Suppress display of the same note

You can stop the same note from appearing multiple times on the one page.

Property	axf:suppress-duplicate-footnote
Value	true | false
Initial	false
Applies to	fo:footnote
Inherited	yes

Specifying true for the axf:duplicate-footnote property suppresses the display of the subsequent notes with the same content when the duplicates would appear on the same page.

When axf:footnote-position="column", only notes in the same column are suppressed.

The contents of <fo:footnote-body> determine whether or not it is the same note.

2.10.5 Footnote stacking

You can display multiple footnotes on the same line.

The XSL 1.1 Recommendation defines a footnote as a separate block, so each note begins on a separate line. AH XSL Formatter can change this operation and treat footnotes as inline.

Property	axf:footnote-stacking
Value	block | inline
Initial	block
Applies to	fo:region-body, fo:footnote

When axf:footnote-stacking is inline, the notes are placed as inline areas.

2.10.6 Keep note and anchor together

A note may be sent to a later page and column than its anchor. Use axf:footnote-keep to send both a note and its anchor to the next page or column if necessary.

Normally, when a note no longer fits within a note area, that note is sent to the next note area, but its anchor does not also move to the new page or column. The note and its note marker might be displayed on separate pages.

Property	axf:footnote-keep
Value	auto | always | none
Initial	auto
Applies to	fo:region-body, fo:footnote

When axf:footnote-keep is always, the anchor line and subsequent lines are sent to the next page so that the footnote its anchor are placed on the same page. For columns, when axf:footnote-position="column", the footnote and anchor are placed in the same column.

If none is specified and the footnote does not fit within the column, the anchor line and subsequent lines will be sent to the next page, and the footnote will try to fit on the original page.

Note

Depending on the length of the footnote and the position of the anchor, page breaks may occur even if the body text is about 3 lines long.

2.10.7 Break a long footnote over multiple pages or columns

You can specify whether and when a long footnote can break to the next page or column.

Property	`axf:footnote-max-height`
Value	`auto` | *<length>* | *<percentage>*
Initial	`auto`
Applies to	fo:region-body
Percentages	refer to the height of the page

Option Property	`auto-break-footnote`
Value	`true` | `false`
Initial	`true`
Applies to	`<formatter-settings>`

`axf:footnote-max-height` specifies the maximum height of a footnote. When a footnote exceeds the specified height, the overflow will be sent to the next page or column.

`axf:footnote-max-height` is not applied on pages and columns without body text. When `axf:footnote-max-height="auto"` is specified, footnote breaking depends on the `auto-break-footnote` setting in the Option Setting File.

axf:footnote-max-height	auto-break-footnote	Result
auto	true	Do not limit footnote height, and break the footnote across the page and column.
auto	false	Do not limit footnote height, but also do not break the footnote across the page or column. Footnotes may overflow.
Specify a positive length or percentage	-	auto-break-footnote has no effect. Limit footnote height.

2.10.8 Endnotes at the bottom of the page

You can float endnotes to the bottom of the page.

Endnotes do not need any special extensions, but by using `axf:float-y="bottom"`, their position can be fixed to the bottom of the page [Figure 2.10-5] (p.167).

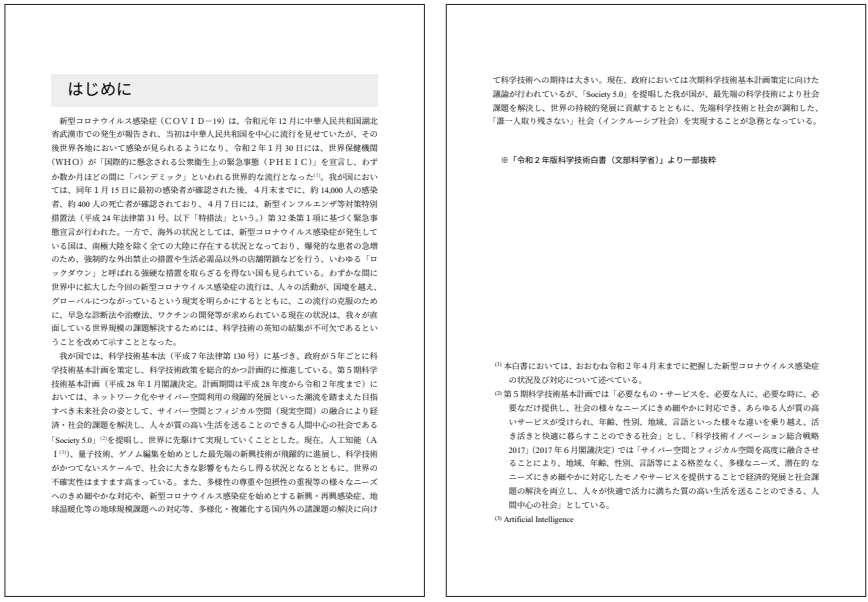
Endnotes at the bottom of the page

```
<!-- Put the endnotes together in a float -->
<fo:float axf:float-y="bottom" axf:float-x="none" ...>
  ...
  <fo:block ...>
    <fo:inline font-size="70%"
      baseline-shift="super">(1)</fo:inline>本白書においては、...
  </fo:block>
  ...
</fo:float>
```

Border between body text and endnotes

```
<!-- Describe a block of border before the endnote -->
<fo:float ...>
  <fo:block keep-with-next="always">
    <fo:leader leader-pattern="rule"
      rule-style="solid" rule-thickness="0.7pt" leader-length="50mm
" />
  </fo:block>
  ...
</fo:float>
```

To draw a border at the start position of the endnote, specify `border-top property`, etc. on the endnote block, or place an `<fo:leader>` element before the first endnote. Parameters of a wave or double-line border can be adjusted. It is easy to adjust the border length with `<fo:leader>`.



はじめに

新型コロナウイルス感染症（ＣＯＶＩＤ−19）は、令和元年12月に中華人民共和国湖北省武漢市での発生が報告され、当初は中華人民共和国を中心に流行を見せていたが、その後世界各地において感染が見られるようになり、令和２年１月30日には、世界保健機関（WHO）が「国際的に懸念される公衆衛生上の緊急事態（ＰＨＥＩＣ）」を宣言し、わずか数か月ほどの間に「パンデミック」といわれる世界的な流行となった[1]。我が国においては、同年１月15日に最初の感染者が確認された後、４月末までに、約14,000人の感染者、約400人の死亡者が確認されており、４月７日には、新型インフルエンザ等対策特別措置法（平成24年法律第31号。以下「特措法」という。）第32条第１項に基づく緊急事態宣言が行われた。一方で、海外の状況としては、新型コロナウイルス感染症が発生している国は、南極大陸を除く全ての大陸に存在する状況となっており、爆発的な患者の急増のため、強制的な外出禁止の措置や生活必需品以外の店舗閉鎖などを行う、いわゆる「ロックダウン」と呼ばれる強硬な措置を取らざるを得ない国も見られている。わずかな間に世界中に拡大した今回の新型コロナウイルス感染症の流行は、人々の活動が、国境を越え、グローバルにつながっているという現実を明らかにするとともに、この流行の克服のために、早急な診断法や治療法、ワクチンの開発等が求められている現在の状況は、我々が直面している世界規模の課題解決するためには、科学技術の英知の結集が不可欠であるということを改めて示すこととなった。

我が国では、科学技術基本法（平成７年法律第130号）に基づき、政府が５年ごとに科学技術基本計画を策定し、科学技術政策を総合的かつ計画的に推進している。第５期科学技術基本計画（平成28年１月閣議決定。計画期間は平成28年度から令和２年度まで）においては、ネットワーク化やサイバー空間利用の飛躍的発展といった潮流を踏まえた目指すべき未来社会の姿として、サイバー空間とフィジカル空間（現実空間）の融合により経済・社会的課題を解決し、人々が質の高い生活を送ることのできる人間中心の社会である「Society 5.0」[2]を提唱し、世界に先駆けて実現していくこととした。現在、人工知能（ＡＩ[3]）、量子技術、ゲノム編集を始めとした最先端の新興技術が飛躍的に進展し、科学技術がかつてないスケールで、社会に大きな影響をもたらし得る状況となるとともに、世界の不確実性はますます高まっている。また、多様性の尊重や包摂性の重視等の様々なニーズへのきめ細やかな対応や、新型コロナウイルス感染症を始めとする新興・再興感染症、地球温暖化等の地球規模課題への対応等、多様化・複雑化する国内外の諸課題の解決に向けて科学技術への期待は大きい。現在、政府においては次期科学技術基本計画策定に向けた議論が行われているが、「Society 5.0」を提唱した我が国が、最先端の科学技術により社会課題を解決し、世界の持続的発展に貢献するとともに、先端科学技術と社会が調和した、「誰一人取り残さない」社会（インクルーシブ社会）を実現することが急務となっている。

※「令和２年版科学技術白書（文部科学省）」より一部抜粋

[1] 本白書においては、おおむね令和２年４月末までに把握した新型コロナウイルス感染症の状況及び対応について述べている。

[2] 第５期科学技術基本計画では「必要なもの・サービスを、必要な人に、必要な時に、必要なだけ提供し、社会の様々なニーズにきめ細やかに対応でき、あらゆる人が質の高いサービスが受けられ、年齢、性別、地域、言語といった様々な違いを乗り越え、活き活きと快適に暮らすことのできる社会」とし、「科学技術イノベーション総合戦略2017」（2017年６月閣議決定）では「サイバー空間とフィジカル空間を高度に融合させることにより、地域、年齢、性別、言語等による格差なく、多様なニーズ、潜在的なニーズにきめ細やかに対応したモノやサービスを提供することで経済的発展と社会課題の解決を両立し、人々が快適で活力に満ちた質の高い生活を送ることのできる、人間中心の社会」としている。

[3] Artificial Intelligence

Figure 2.10-5　Endnotes at the bottom of the page

Section 2.11 Markers and cross-references

You can control the content of cross-references and the markers that are displayed.

2.11.1 Remove duplicate marker references

When consecutive marker references generate duplicate contents, you can suppress one of them.

For example, when a marker reference in the header is used to display a section title, it is unknown how many section title markers are on the page. You may want to display the "first section title to last section title" when there are multiple sections on the same page, but only "section title" when there is one section [Figure 2.11-1] (p.169)

```
<!-- - Display "first section title to last section title" in the h
eader  -->
<fo:static-content
  flow-name="xsl-region-before">
  <fo:block ...>
   <fo:retrieve-marker
      retrieve-class-name="heading"
      retrieve-position="first-including-carryover"/>
    to
    <fo:retrieve-marker
      retrieve-class-name="heading"
      retrieve-position="last-starting-within-page" />
  </fo:block>
</fo:static-content>
```

Consecutive inline marker references may refer to markers with the same content. In the above example, if a `<fo:marker marker-class-name="heading">` does not appear on the page, the heading will be something like "section title 1 to section title 1".

Property	`axf:suppress-duplicate-marker-contents`
Value	`true` | `false`
Initial	`false`
Applies to	all formatting objects
Inherited	yes

`axf:suppress-duplicate-marker-contents` implements the "coalescing markers" feature from the XSL 2.0 requirements. If `true` is specified and there are duplicate marker references, the duplicate marker references and the content between them are removed.

```
<!--  Suppress duplicate display of marker reference contents  -->
<fo:page-sequence ...
  axf:suppress-duplicate-marker-contents="true">
<fo:static-content flow-name="xsl-region-before">...
  </fo:static-content>
  ...
</fo:page-sequence>
```

[Figure 2.11-2] (p.169) shows an example of applying `axf:suppress-duplicate-marker-contents`.

The following conditions are required for marker reference suppression to work:

- `retrieve-marker-class` values are the same
- The `fo:retrieve-marker` or `fo:retrieve-table-marker` are sibling inline elements
- `<fo:marker>` produces an inline element

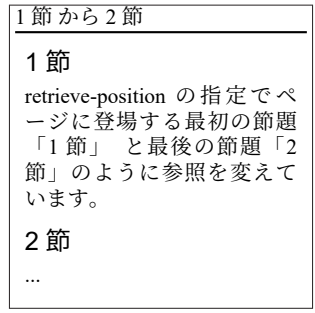


Figure 2.11-1　Display "*first section title to last section title*" in the header

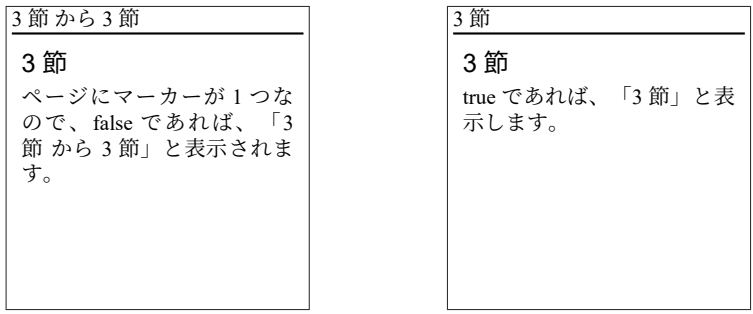


Figure 2.11-2　`axf:suppress-duplicate-marker-contents`

2.11.2 Assumed page number length

When a page number reference refers to a later page, AH XSL Formatter initially uses a temporary value until the correct page number is known.

AH XSL Formatter reserves space for the number of digits in the temporary value, and later adjusts the content when the page number is known. For example, if there is a sentence on page 10 that refers to page 2000, if three digits were assumed, the correction is only the width of one digit. However, when three digits are expected, this may have a significant impact on page references in a booklet with single-digit page numbers or a manual with five-digit page numbers.

Property	`axf:assumed-page-number`
Value	*<number>*
Initial	N/A
Applies to	all formatting objects
Inherited	yes

`axf:assumed-page-number` specifies the temporary page number used for initial formatting.

2.11.3 Page numbers in reverse order

You can make the displayed page numbers count in the reverse order of the pages in the page sequence.

In FOs where page sequences that have different page progression directions are mixed, the page numbers advance in each direction.

Property	`axf:reverse-page-number`
Value	`true` | `false`
Initial	`false`
Applies to	fo:page-number, fo:page-number-citation, fo:page-number-citation-last

If `axf:reverse-page-number` is `true`, the page numbers are assigned in reverse order, counting from the last page in the page sequence.

2.11.4 Display column number with page number

When there are multiple columns, the column number as well as the page number can be displayed.

Property	axf:number-type
Value	page | column | page-and-column
Initial	page
Applies to	fo:page-number, fo:page-number-citation, fo:page-number-citation-last

Specifying column for axf:number-type when generating a page number or page number reference displays the column number instead of the page number.

If page-and-column is specified, both the page number and column number are displayed.

Property	axf:column-number-format
Value	<*string*>
Initial	A
Applies to	fo:page-sequence

axf:column-number-format specifies the column number format. When axf:number-type is page-and-column, the format property specifies the page number format.

2.11.5 Physical page number

The <fo:page-sequence> determines the displayed page numbers. For example, if initial-page-number="100" is specified, the page numbers will start from page 100. AH XSL Formatter can provide the physical page number even when page numbers are reset between page sequences.

Property	axf:physical-page-number
Value	true | false
Initial	false
Applies to	fo:page-number, fo:page-number-citation, fo:page-number-citation-last

When axf:physical-page-number is true, the physical page number is displayed and referenced.

If, for example, you want to put physical page numbers outside the printed page, specify `axf:crop-area-visibility="visible"` on the page master and then generate an absolutely positioned container placed in the crop area containing an `<fo:page-number>` with `axf:physical-page-number="true"`.

If you want to get the total number of pages, assign an ID to the last object to be generated and specify `axf:physical-page-number="true"` for the page number reference that refers to that ID, and it will be the total number of pages.

```
<!-- Serial number / total number of pages -->
<fo:block>
  <fo:inline>
    <fo:page-number axf:physical-page-number="true" /> /
    <fo:page-number-citation ref-id="lastPage"
     axf:physical-page-number="true"/>
  </fo:inline>
  ...
</fo:block>
<!-- Last page -->
<fo:block id="lastPage">...</fo:block>
```

2.11.6 Relative page number

You can generate a relative page number, such as "*after n pages*".

`<fo:page-number*>` generates or references the number of a page within a sequence of pages. In AH XSL Formatter, you can get the relative offset to the page containing a specific ID.

Property	`axf:origin-id`
Value	*<idref>*
Initial	`none`
Applies to	fo:page-number, fo:page-number-citation, fo:page-number-citation-last

Specify *<idref>* in `axf:origin-id`, and the generated or referenced page number is the offset to the page of the element with that ID. The generated page number is "*'ref-id page' – 'origin-id page' + 1*", but if 'ref-id page' comes after 'origin-id page', the page number will be 0. With `<fo:page-number>`, the current page is used instead of 'ref-id page'.

`<fo:page-number-citation>` normally generates the page number of the specified *<refid>*, such as in "*For further information, see page 200.*", but by using `axf:origin-id` property, it is also possible to generate a relative page number, such as "*in n pages*".

```
<!-- Normal page reference -->
<fo:block id="here">
  <fo:page-number-citation ref-id="refered-after" />For further inf
ormation, see page
  ...
  <!--  Relative page reference -->
  <fo:page-number-citation ref-id="refered"
    axf:origin-id="here" /> pages.
  </fo:block>
<!--  Target  -->
<fo:block id="refered-after">...</fo:block>
```

Assuming that the `<fo:block>` with `id="referred-after"` is on page 220 and the `<fo:block>` with `id="here"` is on page 200, the relative page reference of the above example is calculated as "*220 - 200 + 1*", and the cross-reference becomes "*See further information in 21 pages*".

Note

axf:origin-id cannot be used to generate cross-references of the form "*n pages before*. The page number is 0 when origin-id occurs after ref-id.

2.11.7 Suppress page number prefix and suffix

When generating a page number cross-reference, you can individually control whether to suppress the page number prefix and suffix that were specified for pages in a page sequence.

With `<fo:folio-prefix>` and `<fo:folio-suffix>` from XSL 1.1, you can specify text to display before or after a page number when generating a page number or page number reference. These can be different for every page sequence.

It can be useful to include the prefix and suffix when referring to pages from another page sequence but suppress them when referring to pages in the current page sequence.

Property	`axf:suppress-folio-prefix`, `axf:suppress-folio-suffix`
Value	`true` | `false`
Initial	`false`
Applies to	fo:page-number, fo:page-number-citation, fo:page-number-citation-last

Specify `axf:suppress-folio-*="true"` when you want to suppress displaying them as part of a page number or page number reference.

2.11.8 Consecutive page references in an index

You can specify how to abbreviate long sequences of page number references in an index.

`merge-sequential-page-numbers` has been extended to control merging consecutive page number references.

If `merge` in the XSL 1.1 specification is specified, multiple page number references, such as "*2, 3, 4, 5, 6, 7*", are grouped like "*2-7*". Even two consecutive pages are grouped, such as "2, 3" grouped as "*2-3*". As the page numbers increase, the number of duplicated characters increases, such as "*1002-1003*".

Property	`merge-sequential-page-numbers`
Value	`merge` | **`merge-f`** | **`merge-ff`** | `leave-separate`
Initial	merge
Applies to	fo:page-index-citation-list
Inherited	yes

Element	`<axf:index-page-citation-range-f-suffix>`
Parent	fo:index-page-citation-list

Element	`<axf:index-page-citation-range-ff-suffix>`
Parent	fo:index-page-citation-list

With `merge-sequential-page-numbers="merge-f"`, "*2, 3*" is displayed as "*2f.*" when there are 2 consecutive page numbers. For three or more consecutive page numbers, it works the same as `merge`.

`merge-sequential-page-numbers="merge-ff"` displays three consecutive page numbers as "*2ff.*". It works the same as merge-f for two or more consecutive page numbers and the same as `merge` for four or more consecutive page numbers.

The content of `<axf:index-page-citation-range-f-suffix>` and `<axf:index-page-citation-range-ff-suffix>` override the "*f.*" and "*ff.*" suffixes, respectively.

merge-sequential-page-numbers	Indexed page numbers	Application result
leave-separate	1000,1001,1002	1000,1001,1002
merge	1000,1001,1002	1000-1002
merge-f	1000,1001	1000f.
merge-ff	1000,1001,1002	1000ff.

Section 2.12 Paragraph start and end

AH XSL Formatter has multiple extensions for styling the start and end of paragraphs.

2.12.1 Initial letters (drop capitals)

Large initial letters beginning the first line of a paragraph add emphasis to the paragraph.

You could just increase the font size of the character at the beginning of the paragraph by specifying an `<fo:inline>` with a larger font size. However, for the character to sink into the paragraph, you have to calculate and handle multiple aspects based on the height of each line and the relationship with the text that follows the initial letter. `axf:initial-letters` allows you to specify initial letters based on Initial Letters in [Figure 2.12-1] (p.177).

Property	`axf:initial-letters`
Value	`normal` | [[<*number*> | <*length*>] <*integer*>] ? | [<*number*> | <*length*>] `&&` [`drop` | `raise`]] [`adjacent` | <*integer2*>]] ?
Initial	`normal`
Applies to	fo:block

`axf:initial-letters` makes it simple to create initial letters. This property allows you to decolate letters like below:

1. Specify the size of the initial using either the number of lines or a height. A height is converted into the corresponding, possibly non-integer, number of lines.

   For example, if `3` is specified, the initial letters will be sized to fit from the cap-height of the first line to the baseline of the third line.

   The *dcem* unit is based on this size, so dcem cannot be used to specify the height.
2. Specify the number of lines to sink the initial letters. Alternatively, specify either drop or raise. Use drop for a normal drop capital that completely sinks into the paragraph, or use raise for a raised initial that rises up from the baseline on the first line.

   Add `adjacent` or <*integer2*> to specify how to indent the lines following the lines of a dropped initial.

   When the block contains fewer lines than are needed for a dropped initial, the first lines in the next block will also be indented to accommodate the dropped initial.

Drop capital: The upper edge of the enlarged initial is aligned with the height of the first line of text and the lower edge to the baseline of the second (or subsequent) line.

Raised capital : The lower edge of the enlarged initial is aligned with the baseline of the first line.

Sunken capital: The upper edge of the enlarged initial is higher than the first line, and the lower edge is aligned with the baseline of the second (or subsequent) line.

`axf:initial-letters="3 drop"` (Drop)

`axf:initial-letters="3 raise"` (Raised)

`axf:initial-letters="3 2"` (Sunken)

Figure 2.12-1 Initial letters examples

Property	axf:initial-letters-color
Value	*<color>*
Initial	the value of the *<color>* property
Applies to	fo:block
Inherited	yes

axf:initial-letters-color lets you set the color of the initial letters instead of inheriting their color from the block.

Property	axf:initial-letters-width
Value	auto | length | *<percentage>*
Initial	auto
Applies to	fo:block
Inherited	yes
Percentages	refer to the width of containing reference area

Property	axf:initial-letters-text-align
Value	start | center | end | left | right
Initial	end
Applies to	fo:block
Inherited	yes

axf:initial-letters-width specifies the width to use for the initial letters. If auto, it is calculated from the character string or image comprising the initial letters. When the specified width is wider than the value determined by axf:initial-letters-width="auto", the initial letters can be aligned within that width using axf:initial-letters-text-align.

Note

An initial image can be used instead of text. If so, the axf:alttext property is useful if you want a screen reader to read equivalent text for the image in the output PDF.

axf:initial-letters property is ignored in some contexts. Refer to the manual for other precautions.

Property	axf:initial-letters-first-line-head-height
Value	cap-height | x-height | auto
Initial	auto
Applies to	fo:block
Inherited	yes

You can control how a dropped initial aligns with the first line of text. `axf:initial-letters-first-line-head-height` allows you to specify where in the first line the drop cap height should be aligned.

`cap-height` aligns the initial to the cap-height, `x-height` aligns it to the x-height, and `auto` determines it automatically. `auto` aligns to x-height when the next letter is a lowercase small-caps letter or when the number of lines of the dropped initial is equal to the specified number of lines to sink, otherwise it aligns to cap-height.

If `axf:initial-letters="2 2"` and there are two lines in the block, it is x-height.

2.12.2 Move text relative to initial letter

You can adjust the indent of the lines `adjacent` to the initial letter and also of the lines following the initial letter.

When axf:initial-letters includes adjacent, all lines in the block are indented so that the initial letter protrudes from the block. This is called a *hanging capital* [Figure 2.12-2] （p.181）.

```
<!-- Hanging capital -->
<fo:block axf:initial-letters="4 drop adjacent">
  Lorem ...
</fo:block>
```

<*integer2*> instead indents the specified number of lines after the initial letter. For a paragraph with six lines, when `axf:initial-letters="3 drop 1"` is specified, the three lines adjacent to the initial letter plus the fourth line are all indented, but the following lines are not.

注

If all the lines of a paragraph are indented because of an initial letter, depending on the settings of the next paragraph, this paragraph could look pushed in. You can avoid this by specifying `start-indent="-1dcem"` for this paragraph or by increasing the indent of the following paragraphs.

Property	`axf:initial-letters-end-indent`
Value	*<length>+*
Initial	`0pt`
Applies to	fo:block
Inherited	yes

`axf:initial-letters-end-indent` specifies the space at the end side of the initial letter. This determines the beginning of the text adjacent to the initial letter. When multiple values are specified, successive values apply to successive lines.

The last value is used for the corresponding line plus any subsequent lines that are also adjacent to the initial. If the initial capital is 4 lines high, and `1em 3em 2em` are specified, the first line will be 1em from the initial letter, the second line, 3 em, the third line, 2 em, and the fourth line will also be 2 em.

A negative value moves the beginning of the line into the area of the drop capital [Figure 2.12-3]（p.181）.

```
<!-- Put the text into the initial letter  -->
<fo:block axf:initial-letters="4 drop"
  axf:initial-letters-end-indent="-0.25dcem -0.3dcem 0">
  Zorem ...
</fo:block>
```

Figure 2.12-2 Hanging capital

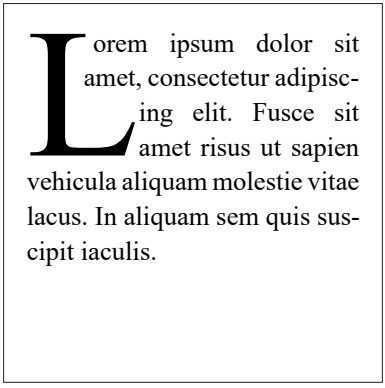


Figure 2.12-3 Text moved into the initial letter

2.12.3 Punctuation marks preceding initial letters

You can control the processing when a paragraph with initial letters begins with punctuation marks.

Property	`axf:initial-letters-leading-punctuation`
Value	`normal` | `hide` | [<*length*> || <*color*>]
Initial	`normal`
Applies to	fo:block
Inherited	yes

`axf:initial-letters-leading-punctuation` specifies the font size and color of any punctuation marks beginning the initial letters.

Property	`axf:initial-letters-leading-punctuation-position`
Value	`normal` | [[`hang` | `intrude`] && <*length*>] ?]
Initial	`normal`
Applies to	fo:block
Inherited	yes

Property	`axf:initial-letters-leading-punctuation-shift`
Value	`normal` | [[`baseline` | `before` | `middle` | `after`] || <*length*>]
Initial	`normal`
Applies to	fo:block
Inherited	yes

`axf:initial-letters-leading-punctuation` specifies the font size and color of any punctuation marks beginning the initial letters.

`axf:initial-letters-leading-punctuation-position`	**Definition**
`normal`	Do nothing.
`hang`	Any preceding punctuation protrudes on the start side of the initial letters. This is different from `axf:hanging-punctuation` because all the preceding punctuation marks protrude.
`intrude`	Intrude into the initial letters.

With a setting other than `normal`, the alignment position and any offset from that position can be specified.

axf:initial-letters-leading-punctuation-shift	Definition
`normal` | <*length*>	Align with the baseline in horizontal layout and at the center in vertical layout. If only <*length*> is specified, it is the amount to shift from normal.
`baseline` <*length*>	Align with the baseline of the last line.
`before` <*length*>	Align with the before side of the initial letter.
`middle` <*length*>	Align with the center of the initial letter.
`after` <*length*>	Align with the after side of the initial letter.

Note that `baseline` indicates the baseline of the last adjacent line, not of the initial letter, and `middle` indicates the center of the initial letter.

An example is shown in [Figure 2.12-4] (p.184).

```
<!-- Align the preceding punctuation marks with the position and si
ze of the text on the first line -->
<fo:block ...
  axf:initial-letters="4 drop"
  axf:initial-letters-leading-punctuation="1em"
  axf:initial-letters-leading-punctuation-position="hang"
  axf:initial-letters-leading-punctuation-shift="before">
  “Extensible Stylesheet Language ..., ...
</fo:block>
```

In this example, the preceding punctuation mark aligns with the position of the text on the first line.

`axf:initial-letters-leading-punctuation="1em"` changes the font size of the mark from the font size of the initial letter to the font size of the paragraph text. `axf:initial-letters-leading-punctuation-position="hang"` pushes the punctuation mark out from the block, and `axf:initial-letters-leading-punctuation-shift="before"` aligns the mark with the before (top) side of the initial letter.

Figure 2.12-4 Punctuation mark aligned with the position and size of the text on the first line

2.12.4 Minimum number of characters in the last line of a paragraph

You can specify the minimum number of words or characters that must remain on the last line of a paragraph.

The `widows` property specifies the minimum number of lines to be placed after a page break.

However, `widows` does not prevent a paragraph from having only one word or character on the last line.

Property	`axf:avoid-widow-words`
Value	`false` | `true` | [<*length*> || <*percentage*>]
Initial	`auto`
Applies to	fo:block
Inherited	yes
Percentages	refer to the width of containing block

`axf:avoid-widow-words` specifies the minimum value for the number of words or characters in the last line of a paragraph. If `true` is specified, the last line will not be one word (one character in CJK).

When `axf:line-break="bpil"` is specified, if a <*length*> or <*percentage*> value is specified, that width will be left on the last line. If both <*length*> and <*percentage*> are

specified, the smaller one will be used. When `axf:line-break` is not `bpil`, a width is equivalent to specifying `true`.

Note

The property might not always work as specified due to its interaction with other breaks. The effect does not appear when the line length is not too short or when the preceding line does not have enough space to trim.

`axf:line-break="bpil"` does not apply to CJK, etc., so even if *<length>* or *<percentage>* are specified, that text will not be changed from "two or more characters are required for last line".

Exclude punctuation marks from CJK widow words

When `axf:avoid-widow-words="true"` is specified, the last line may still contain only one non-punctuation character, such as "す。". This is because CJK punctuation marks are included in the character count.

Property	`axf:avoid-widow-words-cjk-punctuation`
Value	`auto` | `true` | `false` | *<string>*
Initial	`auto`
Applies to	fo:block
Inherited	yes

Option Property	`avoid-widow-words-cjk-punctuation`
Value	`true` | `false` | *<string>*
Initial	`false`
Applies to	`<formatter-settings>`

When the `axf:avoid-widow-words-cjk-punctuation` property is `true` (or `axf:avoid-widow-words-cjk-punctuation` property is `false` and the `avoid-widow-words-cjk-punctuation` option setting is true), characters classified as CL (Close Punctuation) or CP (Close Parenthesis) in UAX #14: Line Breaking Properties (such as "”" (U+2019), '’" (U+201D), punctuation marks, close-parenthesis, etc.) are excluded from the count.

If *<string>* is specified, each character in the string is also excluded from the count. For example, if specified as `avoid-widow-words-cjk-punctuation="'なのです'"`, this will prevent the last line from being just "*なのです。*".

2.12.5 Indent of paragraph beginning a page or column

You can specify a different text indent for the first line of paragraph if at the beginning of a page or column.

Property	`axf:text-indent-if-first-on-page`
Value	*<length>* | *<percentage>* | `auto`
Initial	`auto`
Applies to	fo:block, fo:block-container
Inherited	yes
Percentages	refer to the logical width of the containing block

The `axf:text-indent-if-first-on-page` value is used as the `text-indent` property value when the first line of a block is at the beginning of a page or column.

```
<!-- Same axf:text-indent-first-on-page and text-indent
for all paragraphs except headings -->
<fo:block ...>Heading</fo:block>
<fo:block ...
  axf:text-indent-if-first-on-page="2em"
  text-indent="4em">
    Lorem ipsum ...
</fo:block>
...
```

[Figure 2.12-5] (p.187), `axf:text-indent-if-first-on-page="2em"` and `text-indent="4em"` are specified for all the blocks except for the header block. In the left column of the figure, the header block, containing "*Heading*", is at the top, so the next block is indented 4 em with `text-indent="4em"`. The first line in the middle column is a continuation of the previous paragraph, so it is not indented. The first line of the right column is the beginning of the paragraph, so it is indented 2 em because of the `axf:text-indent-if-first-on-page="2em"`.

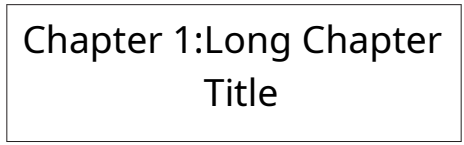


Figure 2.12-5 `axf:text-indent-if-first-on-page` example

2.12.6 Alignment of the first line

Specify the alignment of the first line of the block.

XSL 1.1 defines the `text-align-last` property that controls the alignment of the last line but it does not define a property that controls the alignment of the first line.

Property	`axf:text-align-first`
Value	`relative` | `start` | `center` | `end` | `justify` | `inside` | `outside` | `left` | `right`
Initial	`relative`
Applies to	fo:block
Inherited	yes

`axf:text-align-first` specifies the alignment of the first line of a block.

If `relative` is specified, do nothing. Other values behave the same as the `text-align` property.

`axf:text-align-first` also applies to a line where the last character of the previous line is LINE FEED (U+000A).

In a single-line block, `axf:text-align-first` takes precedence over `text-align-last`.

Note

For example, when `text-align="justify"` and `text-align-last="right"` are specified for a block, if the text to be formatted is one line, that line is treated as the last line and it will be right-aligned because of `text-align-last="right"`.

If specified as `text-align-last="justify"`, it will be justified, but in a block with the same settings and with two or more lines, the last line will be justified. If `axf:text-align-first="justify"` is specified, you can specify as "*justify when there is one line, justify the normal line when there are more lines, and right-align the last line*".

It is also useful for formatting long tables of contents and index items.

2.12.7 Justification of the last line

Whether or not to perform justification on the last line can depend on the content. If `justify` is specified to align the text, the text is justified. If justify the last line when the number of characters is small, it may be difficult to read.
In AH XSL Formatter, you can control whether to justify based on the space available in the last line.

Property	`axf:flush-zone`
Value	`none` | *<length>* | *<percentage>*
Initial	`none`
Applies to	fo:block
Inherited	yes
Percentages	refer to the width of containing block

When the space at the end of the last line is less than or equal to the value specified for `axf:flush-zone`, the last line is treated as if `text-align-last` is `justify`.

If `text-align="justify"` and `text-align-last="left"` are specified, the last line is left-aligned unless the remaining space after the left-aligned text would be less than or equal to the `axf:flush-zone` value, in which case the line will be justified.

When `none` is specified, `axf:flush-zone` has no effect.

```
<!-- axf:flush-zone works -->
<fo:block
  text-align-last="left" text-align="justify"
  axf:flush-zone="30%">
  ...
</fo:block>
<!-- axf:flush-zone does not work-->
<fo:block
  text-align-last="left" text-align="left"
  axf:flush-zone="30%">
  ...
</fo:block>
  <!-- axf:flush-zone does not work-->
<fo:block
  text-align-last="justify" text-align="justify"
  axf:flush-zone="30%">
  ...
</fo:block>
```

Section 2.13 Line breaking

You can control the algorithms used for line breaking, non-starter and non-ending characters in CJK and hyphenation.

Line breaking rules

AH XSL Formatter performs two types of line breaking. It also provides several ways to control non-starter and non-end-of-line characters in CJK.

Property	`axf:line-break`
Value	`auto` | [[`normal` | `strict` | ~~`loose`~~ | ~~`anywhere`~~] || [`line` | `bpil`]]
Initial	`auto`
Applies to	all block-level formatting objects
Inherited	yes

axf:line-break controls the line break algorithm. Most values are from [CSS3-Text] Breaking Rules for Punctuation: the 'line-break' property. `line` and `bpil` have been added mainly for European text. Also, `loose` and `anywhere` are not supported.

`normal` and `strict` apply to CJK.

`axf:line-break`	Definition
`auto`	Same as `normal`.
`line`	Execute line breaking line by line.
`bpil`	Use Knuth-Plass line-breaking algorithm to balance the number of characters in each line.
normal	CJK non-starter characters defined in JIS X 4051 "Formatting rules for Japanese documents" are not treated as non-starter characters.
`strict`	CJK non-starter characters are treated as such.

`xml:lang` or `language`, or the `default-lang` option, specifies the language of a block.

For details on `bpil` and `line` line breaking, see [Line breaking in European languages] (p.190).

For details on specifying the non-starter and non-ending characters in CJK, see the [2.13.9 Non-starter and non-ending characters in CJK] (p.200). For overall details on line breaking control, such as the characters to be processed, see the manual.

2.13.1 Line breaking in European languages

You can control the line breaking algorithm used with European text.

Breaking Paragraphs into Lines (BPIL): This algorithm breaks lines so that lines are approximately the same length. BPIL works well with languages, such as European languages, that use irregular-length words separated by spaces.

When `axf:line-break` is `bpil` or is not specified, BPIL is applied to blocks in the languages specified in the `bpil` option.

[Figure 2.13-1] (p.190) shows the same text formatted when `axf:line-break` is `line` and `bpil`. In the `line example`, the top half contains more words per line, and word spacing differs between the top half and bottom half. In the `bpil` example, word spacing is about the same except for the last line.

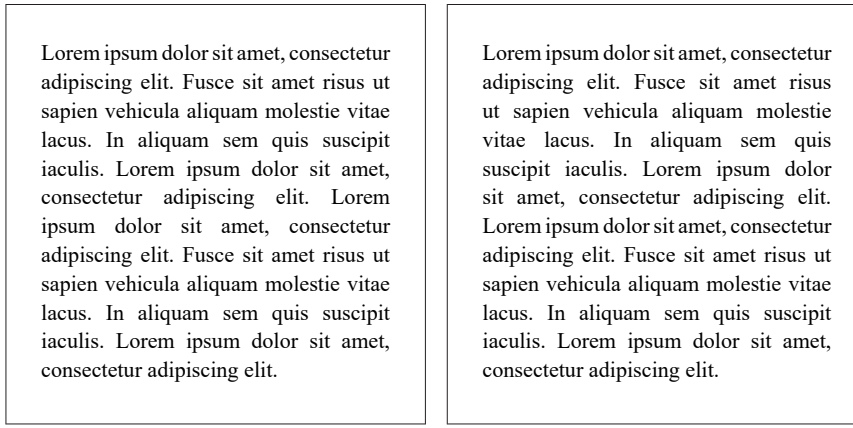


Figure 2.13-1 European processing by `axf:line-break`

BPIL language

Option Property	`bpil`
Value	*<string>*
Initial	`Latn Grek Cyrl Armn Geor`
Applies to	`<formatter-settings>`

The `bpil` option specifies languages or scripts to which to apply BPIL line breaking. When a script is specified, the language that the script represents is set as the target.

Specify languages or scripts separated by spaces. The possible values are:

- Language code (e.g. `en`)
- Language tag (e.g. `en-US`)
- Script (e.g. `Latin`)

If a block is in a language to which `bpil` applies but there is a part that should not use BPIL, it is better to explicitly specify `axf:line-break="line"`.

BPIL

Option Property	`bpil-limit-chars`
Value	<*integer*>
Initial	`50000`
Applies to	`<formatter-settings>`

Option Property	`bpil-minimum-line-width`
Value	<*number*>
Initial	`8`
Applies to	`<formatter-settings>`

Option Property	`bpil-penalty-hyphenation`
Value	<*integer*>
Initial	`100`
Applies to	`<formatter-settings>`

BPIL optimizes line breaking within a paragraph, so a longer paragraph with more characters requires more processing.

The `bpil-limit-chars` option specifies the maximum number of characters to which to apply BPIL. BPIL is not applied to blocks that exceed approximately this number of characters.

The `bpil-minimum-line-width` option specifies the minimum line width, in em, to which to apply BPIL. Less than 1 is considered 1.

When BPIL is applied to a block with `hyphenation="true"`, BPIL handles the hyphenation.

The `bpil-penalty-hyphenation` option, with the range 0 to 1000, influences the likelihood of hyphenation occurring. Hyphenation will occur more frequently as the value approaches 0.

BPIL exclusions

BPIL is not applied, and AH XSL Formatter displays a warning, when formatting a block containing any of:

- Leader (`<fo:leader>`, etc.)
- Float with complex intrusion (BPIL may be applied for simple intrusion)
- Form field
- Bidirectional text
- `axf:indent-here`
- `<axf:tab>` or TAB that treated as `<axf:tab>` by `axf:tab-treatment="preserve"`
- Lines that overflow instead of breaking, because of `wrap-option="no-wrap"` or similar
- Font size or line height is less than or equal to 0
- Area less than or equal to `bpil-minimum-line-width`
- More characters than `bpil-limit-chars`
- Page break to a page with a different size

BPIL is also not applied to the first line of a block that contains `<fo:initial-property-set>`. In addition, BPIL may not be applied on the second and subsequent pages (or columns) of blocks that extend over three pages (or columns).

For these reasons, BPIL has difficulty optimizing columns, especially more complicated ones.

2.13.2 Word breaking

You can specify word breaking rules.

A word here indicates a string of characters with Unicode General Category of L (Letter), M (Mark), or N (Number).

Line breaks within words

The hyphenation rules, etc., normally determine where a word can break, but you can override those breaks.

Property	axf:word-break
Value	normal | keep-all | break-all | ~~break-word~~ | keep-non-spaces
Initial	normal
Applies to	all block-level and inline-level formatting objects
Inherited	yes

axf:word-break is based on the [CSS3-Text] Breaking Rules for Letters: the 'word-break' property property.

When normal, the normal line breaking rules apply.

When keep-all, words are not broken. The rest of the line follows the normal line breaking rules. hyphenate="true" is ignored, which is useful when you don't want to break or hyphenate words.

When break-all and the script is Latn, Cyrl, Grek or Zyyy, words can break anywhere. Other scripts follow the normal line breaking rules.

When keep-non-spaces, lines break only at spaces. This is useful, for example, when words that you want to prevent from splitting contain hyphens.

Line breaks with long words

axf:word-break="break-all" may cause a break at an undesirable point in a word. axf:word-wrap is used for breaking only words that are longer than the line length.

Property	axf:word-wrap
Value	normal | break-word | ~~break-spaces~~
Initial	break-word
Applies to	all block-level and inline-level formatting objects
Inherited	yes

axf:word-wrap is based on the [CSS3-Text] Overflow Wrapping: the overflow-wrap/word-wrap property property. It is useful for words, URLs, etc. that are longer than the line length and are otherwise unbreakable.

If normal, no word will be broken, and a long word will extend beyond the area.

If break-word, a long word is forcibly broken at an appropriate point. As [Figure 2.13-2] (p.194) shows, it can produce better breaks than axf:word-break="break-all".

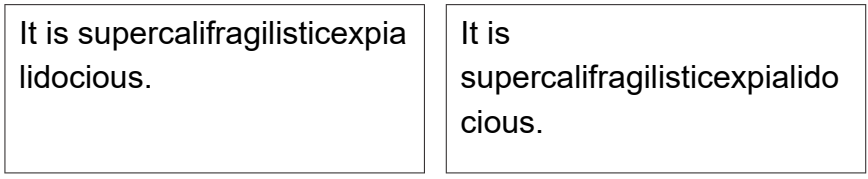


Figure 2.13-2 `axf:word-break` and `axf:word-wrap`

2.13.3 Unbreakable words

The option settings can specify the words you do not want to break.

While it is possible to prohibit breaking in an individual inline element, you can also prohibit a word from breaking anywhere in the entire document.

Option Element	`<unbreakable-words>`
Parent	`<formatter-settings>`

`<unbreakable-words>` is a child of `<formatter-settings>`.

The content of `<unbreakable-words>` is the words that you want to prohibit from breaking, separated by line breaks.

```
<!-- Register words that must not break -->
<formatter-settings>
  <unbreakable-words>
PDF1.3
PDF1.4
...
  </unbreakable-words>
<formatter-settings>
```

2.13.4 Suppressing hyphenation

There are multiple ways to suppress hyphenation.

Suppress hyphenation

`hyphenation-ladder-count` specifies the upper limit of consecutive hyphenated lines. AH XSL Formatter provides more control, such as the number of characters in a word and the remaining length of the line width.

Property	axf:hyphenation-minimum-character-count
Value	<*number*>
Initial	1
Applies to	fo:block, fo:character
Inherited	yes

axf:hyphenation-minimum-character-count specifies the minimum length for a word to be hyphenated as an integer greater than or equal to 1.

Property	axf:hyphenation-zone
Value	none | length | percentage
Initial	none
Applies to	fo:block
Inherited	yes
Percentages	refer to the width of containing block

When axf:hyphenation-zone is over 0, a word will not be hyphenated if the width from the end of the previous word to the end of the line is less than or equal to the specified value.

Invalid when the value is 0 or less.

Property	axf:hyphenate-hyphenated-word
Value	true | false
Initial	true
Applies to	fo:block
Inherited	yes

When axf:hyphenate-hyphenated-word is false, words in the original text that contain hyphens are not further hyphenated. Hyphenated words can break at the existing hyphens. For example, the word "*axf:hyphenate-hyphenated-word*" may be broken after "*axf:hyphenate-*" and after "*hyphenated-*".

When true, a hyphen may be added if the word is split at the end of a line. The recognized hyphens are HYPHEN-MINUS "-" (U+002D), SOFT HYPHEN "" (U+00AD), HYPHEN " ‐ " (U+2010) and NON-BREAKING HYPHEN "-" (U+2011).

Hyphen at the end of page and column

Option Property	hyphenation-keep-mode
Value	word | line
Initial	word
Applies to	<formatter-settings>

The hyphenation-keep-mode option controls the behavior when hyphenation-keep is page or column and the word at the end of the page or column is hyphenated.

When word, only the hyphenated word is pushed to the next page or column.

When line, the line containing the word is pushed out to the next page or column.

When one word in the last line on the page is pushed because of word but the lines on the next page are fewer than widows lines, the whole last line is pushed to the next page or column anyway. While any new last line violates hyphenation-keep, more lines will be pushed. As such, it may not be possible to resolve some combinations of widows and hyphenation-keep.

When a line is pushed, if the last word of the new last line is also hyphenated, that line will be pushed to the next page or column. To avoid this, also use hyphenation-ladder-count to limit the number of consecutive lines ending in a hyphen.

2.13.5 Hyphenation information

You can create a hyphenation exception dictionary to control hyphenation processing.

AH XSL Formatter does not require you to provide hyphenation rules. However, you can add a dictionary of exceptions to standard hyphenation processing. In the GUI, reload and reformat by selecting [**Format**]>[**Reload Hyphenation Dictionary**].

The exception dictionary should be in the hyphenation folder in the AH XSL Formatter installation or in the folder specified by the AHF72_HYPDIC_PATH or AHF72_64_HYPDIC_PATH environment variable, as appropriate.

The dictionary file name is a language code or language tag plus a .xml extension. The language tag may contain either HYPHEN-MINUS "-" (U+002D) or LOW LINE "_" (U+005F); for example, “en_us".

Use a two-letter language code, if it exists; otherwise, use a Terminology code. Also use a two-letter country code, if that exists.

See Manual for details on the priority of the hyphenation exception dictionary and installation folders and environment variables.

Hyphenation exception dictionary

Element	<hyphenation-info>

Element	<hyphen-char>
Parent	hyphenation-info

Element	<hyphen-min>
Parent	hyphenation-info

Element	<exceptions>
Parent	hyphenation-info

Element	<hyphen>
Parent	exceptions

Element	<non-eol-words>
Parent	hyphenation-info

<hyphenation-info> is the root element of a hyphenation exception dictionary. It contains: <hyphen>; <hyphen-char> representing hyphenation characters; <exceptions> specifying the data of the exception dictionary; <hyphen-min> specifying the minimum number of characters before and after the line break position; and <non-eol-words> specifying words that are prohibited from breaking at the end of a line.

When included in the FO document as a child of <axf:hyphenation-info>, the elements must be in the exception dictionary namespace:

http://www.antennahouse.com/names/XSL/Hyphenations

Hyphenation information in FO document

Element	<axf:hyphenation-info>
Property	src, language
Parent	fo:declarations

In addition placing an XML file in a folder, the hyphenation exception dictionary can be defined in an <axf:hyphenation-info> child of <fo:declarations>. src specifies

an external dictionary file or you can define the dictionary directly as a child element. However, only `<axh:exceptions>` is supported as a child of `<axf:hyphenation-info>`.

TeX hyphenation dictionary

AH XSL Formatter incorporates a custom hyphenation process that supports more than 40 languages. It additionally supports TeX hyphenation dictionaries in XML in common with Apache FOP. See Manual and Apache FOP website for more information.

Option Property	`HyphenationOption`
Value	`true` | `false`
Initial	`true`
Applies to	`<formatter-settings>`

If the `HyphenationOption` option is `false`, only TeX dictionaries can be used for hyphenation.

You must provide dictionaries for all languages you want to hyphenate.

Partial TeX hyphenation

Option Property	`hyphenation-TeX`
Value	*<string>*
Initial	empty string
Applies to	`<formatter-settings>`

The `hyphenation-TeX` option is a comma-separated list of the languages to be hyphenated using TeX dictionaries.

2.13.6 Hyphenation exception example

This is an example of hyphenation exception information.

```
<!-- Hyphenation dictionary description -->
<hyphenation-info>
  <hyphen-char value="=" />
  <exceptions>
  as=so=ciate
  ta<hyphen />ble
  </exceptions>
</hyphenation-info>
```

<exceptions> specifies the words and their hyphenation positions. Words are separated by line breaks. Hyphenation positions are marked by either <hyphen> or the character specified in the value property of <hyphen-char>.

2.13.7 Minimum characters before and after hyphenation

You can specify the minimum number of characters left before and after a hyphen.

Minimum characters before and after hyphenation

Property	hyphenation-remain-character-count, hyphenation-push-character-count
Value	<*number*> | **auto**
Initial	**auto**
Applies to	all-elements

hyphenation-remain-character-count specifies the minimum number of characters before a hyphen, and hyphenation-push-character-count specifies the minimum number of characters that are pushed to the next line. AH XSL Formatter adds an auto value. When auto is specified, if <hyphen-min> specifies valid values, those values are used, otherwise 2 is assumed.

<hyphen-min>

Option Property	hyphen-min
Value	false | true
Initial	false
Applies to	<formatter-settings>

When the hyphen-min option is true, <hyphen-min> in the hyphenation exception dictionary specifies the minimum number of characters before and after a hyphen.

Specify the minimum number of characters before and after hyphens

The minimum number of characters at the end and beginning of a line during hyphenation processing can be controlled with the properties of the <hyphen-min> element.

before specifies the minimum number of characters remaining at the end of the line, and after specifies the minimum number of characters to be pushed to the next line.

```
<!-- Minimum number of characters before and after hyphens -->
<hyphenation-info>
  <hyphen-min before="3" after="2" />
</hyphenation-info>
```

2.13.8 Hyphenating uppercase words

You can suppress hyphenation of words composed of uppercase letters.

When `hyphenate` is `true`, the word at the end of the line is broken according to the hyphenation rule.

Property	`axf:hyphenate-caps-word`
Value	`true` | `false`
Initial	`true`
Applies to	fo:block
Inherited	yes

When `axf:hyphenate-caps-word` is `false`, words composed of uppercase letters will not be hyphenated. A word containing a hyphen, such as "*XSL-FO*", is considered multiple words and there may be a line break at the hyphen.

Lowercase characters formatted as small capitals are considered lowercase.

Note

To avoid hyphenating a word in small capitals, such as initial word, enclose the word in `<fo:inline>` element and specify `hyphenate="false"`.

2.13.9 Non-starter and non-ending characters in CJK

You can control the characters that should not appear at the beginning or end of a line in CJK.

The CJK line breaking algorithm in AH XSL Formatter is a modification of UAX #14: Line Breaking Properties. For example, it allows breaking at full-width parentheses.

When `axf:line-break` is `normal`, the non-starter-characters are the Nonstarter characters specified in UAX #14: Line Breaking Properties.

`axf:line-break="strict"` also includes the non-starter Japanese characters specified in JIS X 4051 "Formatting rules for Japanese documents".

Include or exclude non-starter and non-end-of-line characters

You can include or exclude characters as non-starter and non-end-of-line characters.

Property	`axf:append-non-starter-characters`, `axf:append-non-end-of-line-characters`
Value	<*string*>
Initial	empty string
Applies to	fo:page-sequence

Property	`axf:except-non-starter-characters`, `axf:except-non-end-of-line-characters`
Value	<*string*>
Initial	empty string
Applies to	fo:page-sequence

Option Property	`append-non-starter-characters`, `append-non-end-of-line-characters`
Value	<*string*>
Initial	empty string
Applies to	`<formatter-settings>`

Option Property	`except-non-starter-characters`, `except-non-end-of-line-characters`
Value	<*string*>
Initial	empty string
Applies to	`<formatter-settings>`

`axf:append-non-starter-character` specifies additional Nonstarter characters.

`axf:except-non-starter-character` specifies characters to exclude. In a narrow column, it can be used, for example, to exclude small kana characters and Japanese prolonged sound mark as non-starter-characters.

`axf:append-non-end-of-line-character` specifies additional non-end-of-line-characters, and `axf:except-non-end-of-line-character` specifies characters to exclude.

The `append-non-starter-characters` and `append-non-end-of-line-characters` options specify the initial set of additional non-starter and non-end-of-line characters, respectively.

Similarly, the `except-non-starter-characters` and `except-non-end-of-line-characters` options specify additional characters to exclude as non-starter and non-end-of-line characters, respectively.

IDEOGRAPHIC SPACE (U+3000) as non-starter character

Option Property	`non-starter-ideographic-space`
Value	`true` | `false`
Initial	`true`
Applies to	`<formatter-settings>`

Unicode 6.3 and later treats IDEOGRAPHIC SPACE (U+3000) as a non-starter character.

Consider IDEOGRAPHIC SPACE inserted after a Japanese sentence ending with FULLWIDTH QUESTION MARK “？” (U+FF1F) or FULLWIDTH EXCLAMATION MARK “！” (U+FF01). For the text “ありがとう！` ` さようなら！”, where the “ありがとう！” ends the line, if IDEOGRAPHIC SPACE is a non-starter, it cannot come at the start of the line, so ！` ` is sent to the next line, but if “！” is non-starter-character, the “う！` ` is sent to the next line.

When the `non-starter-ideographic-space` option is `true`, IDEOGRAPHIC SPACE is treated as a non-starter-character.

2.13.10 Polish, Hungarian and Czech conventions

Polish, Hungarian, Czech, etc., do not allow a single-letter word at the end of a line.

Option Property	`avoid-orphan-single-word`
Value	`false` | <*string*>
Initial	`pol,hun,ces`
Applies to	`<formatter-settings>`

The `avoid-orphan-single-word` option contains a comma-separated list of language codes of languages for which no single-letter word is placed at the end of the line.

If `>false` or empty string, single-letter words are always allowed at the end of lines.

2.13.11 Line-break after SOLIDUS "/" (U+002F)

You can prevent words from breaking after SOLIDUS

A URL or path retains its meaning even if it breaks after a SOLIDUS, but to break words such as "*m/s*", which combines two units, into "*m/*" and "*s*" is sometimes undesirable.

Property	`axf:abbreviation-character-count`
Value	`auto` | <*number*>
Initial	`auto`
Applies to	fo:block
Inherited	yes

Option Property	`abbreviation-character-count`
Value	<*number*>
Initial	`3`
Applies to	`<formatter-settings>`

`axf:abbreviation-character-count` specifies the number of characters to be left after SOLIDUS as an integer between 0 and 10. When `axf:abbreviation-character-count="3"` is specified for "*m/second*", it may split into "*m/sec*" and "*ond*".

If you specify `0`, this control is not applied. A value greater than 10 is considered 10.

2.13.12 Indent on the fly

You can specify the indentation of the following lines from the middle of the text, not on the block.

Property	`axf:indent-here`
Value	`none` | <*length*>
Initial	none
Applies to	fo:inline

An `<fo:inline>` with an `axf:indent-here` value other than none sets the indent for following wrapped lines. The indent is the position of the `<fo:inline>` plus the length in the property value.

[Figure 2.13-3] (p.204) shows an example of chapter title which has chapter numbering. After line-breaking of the first line, continuous lines are indented to its end of the chapter number label "*Chapter 1:*".

```
<!-- axf:indent-here example -->
<fo:block ...>
  Chapter 1:<fo:inline axf:indent-here="0pt"/>Long Chapter Title
</fo:block>
```

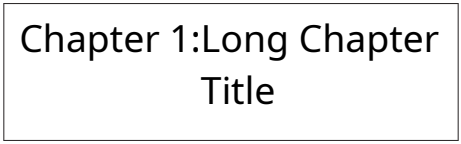


Figure 2.13-3 `axf:indent-here` example

If you wrap the "Chapter 1" with an `<fo:inline>` that specifies `axf:indent-here="0pt"`, the start position of "Chapter" determines the indentation caused by the `axf:indent-here`.

`axf:indent-here` only works on continuous lines. For example, it does not apply after an empty `<fo:block>` forces a line breaking, as shown in [Figure 2.13-4] (p.204).

If you want to break a line at a specific point in the text while also using `axf:indent-here` to cause an indent, you can force the text to wrap by using `<axf:tab>` to fill the remainder of the line.

```
<!-- axf:indent-here does not apply -->
<fo:block ...>
  Chapter 1:<fo:inline axf:indent-here="0pt"/>そこそこに<fo:block/>Lo
ng Chapter Title
</fo:block>
```

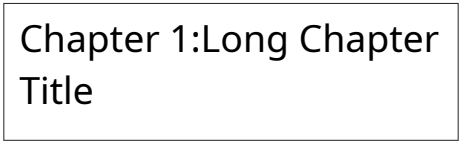


Figure 2.13-4 Where `axf:indent-here` does not apply

2.13.13 SOFT HYPHEN (U+00AD)

You can change the display of SOFT HYPHEN for use with special fonts.

SOFT HYPHEN (U+00AD) is normally visible only at line breaks. However, some fonts, such as for pictographic characters, may assign a glyph to SOFT HYPHEN.

Property	`axf:soft-hyphen-treatment`
Value	`auto` | `preserve`
Initial	`auto`
Applies to	all formatting objects
Inherited	yes

When `axf:soft-hyphen-treatment` is `preserve`, SOFT HYPHEN is always displayed. `auto` enables normal processing.

[Figure 2.13-5] (p.205) shows how `axf:soft-hyphen-treatment` is used. Wingdings is a Windows font with non-standard characters.

```
<!-- Always display SOFT HYPHEN -->
<fo:block>abc&#xAD;xyz</fo:block>
<fo:block axf:soft-hyphen-treatment="preserve"
  font-family="Wingdings">abc&#xAD;xyz</fo:block>
```

Figure 2.13-5 Normal processing (left) and processing with SOFT HYPHEN displayed (right)

Section 2.14 Paragraph placement and baseline grid

You can control the baseline grid and whether, based on its page position, a block is displayed or hidden.

2.14.1 Baseline grid

You can adjust the placement of text, figures and tables relative to a baseline grid.

Baseline grid: The feature to align the arrangement of lines and blocks to the assumed grid on the page is called a baseline grid.

AH XSL Formatter can be set a baseline grid based on font information.

With care, the following controls are possible:

- Constant line pitch to make the document easier to read.
- Lines in adjacent columns on the same baseline to improve the appearance.
- Lines on the front and back of a page aligned to improve the appearance.
- No unintentional gaps before the first line and after the last line of the page.
- Text block that exactly fits an integer number of lines.

Property	`axf:baseline-grid`
Value	`normal` | `none` | `root` | `new`
Initial	`normal`
Applies to	block-level formatting objects, fo:flow and fo:static-content / block containers

`axf:baseline-grid` specifies the baseline grid.

`axf:baseline-grid`	Definition
`normal`	Neither sets nor clears the baseline grid.
`none`	Clears the baseline grid.
`root`	Sets the baseline grid using the font and line height specified for the root element.
`new`	Sets a new baseline grid. The new baseline grid is established by using the font and line height specified for the current element.

While `axf:baseline-grid` can be specified on a block level object, it is possible, for example, to set the baseline grid for the body region with

axf:baseline-grid="root" on the <fo:flow>, and not use the baseline grid in a formula block with axf:baseline-grid="none".

axf:baseline-grid is not inherited. When axf:baseline-grid="new" is specified in the parent, if the child element does not also set a baseline grid, then the baseline grid set in the parent element is used as if axf:baseline-grid="normal" is specified.

You can also specify the alignment to the existing baseline grid of a figure that does not contain text or a block that has a different grid.

Property	axf:baseline-block-snap
Value	none | [auto | before | after | center] || [border-box | margin-box]
Initial	auto border-box
Applies to	block-level elements

axf:baseline-block-snap specifies how the current block is aligned with its parent's baseline grid.

axf:baseline-block-snap	Definition
none	Do not align blocks to the baseline grid.
auto	before at the beginning of the page or column, after at the end, center otherwise.
before	Align the before-side edge of the block with the text-before-edge line on the baseline grid.
after	Align the after-side edge of the block with the text-after-edge line on the baseline grid.
center	Center the block between the text-before-edge and text-after-edge lines on the baseline grid.

You can also specify whether to use a margin-box or a padding-box as the edge of the block.

2.14.2 Hide a block at the top of a page or column

You can control the display of a block at the top of a page or column.

When formatting a sequence of multiple blocks, which block is placed at the top of each new page or column is not known until formatting time. However, whether to display a block when it is at the top of a page or column can be controlled.

Property	axf:suppress-if-first-on-page
Value	true | false | unless
Initial	false
Applies to	fo:block, fo:block-container

When axf:suppress-if-first-on-page is true, the block is not shown when it would appear the top of a page or column.

When false, this property has no effect.

unless will omit the block when it would appear at a place other than the top of the page or column.

The suppressed block actually exists as a zero-height block, and id cross-references to it and so on are still valid.

Note

When there are multiple columns, a block with span="all" creates a column break. axf:suppress-if-first-on-page="true" is useful when you want to use this block as a text separator but do not want the separator to appear at the top of a page.

Section 2.15 Overflow processing

You have precise control when text overflows a fixed area.

2.15.1 Overflow processing

AH XSL Formatter provides extensions for overflow handling.

This includes extension properties for finer control of the selected overflow handling method.

When placing text, once you reach the limit of the available area, text placement resumes from the next area.

Sometimes, however, there are featured areas or pages, such as a table cell or an item in a brochure, and you may want to keep the text within that area.

If you stop lines from breaking, or if you insert more text than can be placed in a fixed-height block, overflow will occur.

Overflow can occur in both the block progression direction [Figure 2.15-1] (p.210) and the inline progression direction.

`overflow` specifies how to handle overflow when it occurs.

AH XSL Formatter extends the values defined in the XSL 1.1 specification.

Property	`overflow`
Value	`visible` | `hidden` | ~~`scroll`~~ | `error-if-overflow` | `repeat` | **`replace`** | **`condense`** | `auto`
Initial	`auto`
Applies to	all block-level formatting objects
Inherited	yes

In addition to standard processing, overflowing text can be modified (`replace`) or the text can be condensed (`condense`).

The `scroll` value is not supported.

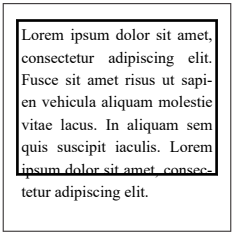


Figure 2.15-1 Overflow in the block progression direction (`overflow="visible"`)

2.15.2 Overflow error reporting

Control the error conditions and processing during overflow processing.

Report an error with `overflow="auto"`

You can specify `overflow="error-if-overflow"` to report an error when an overflow occurs. However, it may be difficult to specify this in every place where overflow may occur.

Option Property	`error-if-overflow`
Value	`true` | `false`
Initial	`false`
Applies to	`<formatter-settings>`

If the `error-if-overflow` option is `true`, overflow occurring for a block with `overflow="auto"` will generate an error.

Limits for overflow processing

The `overflow` property may specify that when overflow occurs, text should be kept within the area by reducing its font size or condensing it by other means.

However, condensing it too much may affect readability. If you want to generate an error when the display is unreadable, you can set overflow limits that generate an error when they are exceeded.

Property	axf:overflow-limit
Value	<*length*>{1,2}
Initial	Depends on system defaults
Applies to	block formatting elements
Inherited	yes

Property	axf:overflow-limit-block
Value	<*length*>
Initial	Depends on system defaults
Applies to	block formatting elements
Inherited	yes

Property	axf:overflow-limit-inline
Value	<*length*>
Initial	Depends on system defaults
Applies to	block formatting elements
Inherited	yes

Option Property	overflow-limit-block
Value	<*length*>
Initial	0pt
Applies to	<formatter-settings>

Option Property	overflow-limit-inline
Value	<*length*>
Initial	0pt
Applies to	<formatter-settings>

`axf:overflow-limit-block` sets the limit of the block progression direction, and `axf:overflow-limit-inline` sets the limit value in the inline progression direction.

`axf:overflow-limit` is a shorthand. If there is only one value, it applies to both the block progression direction and the inline progression direction. If there are two

values, the first value is for the inline progression direction and the second is for the block progression direction.

The `overflow-limit-block` and `overflow-limit-inline` options set the system default values for the block progression direction and the inline progression direction, respectively. When `overflow="error-if-overflow"` is specified, AH Formatter reports a level 2 error for an overflow that exceeds the values specified by `*overflow-limit*`, and level 1 error for one that does not.

For details on the error output, see the Manual.

2.15.3 Replace overflowing text

When a block overflow occurs in the block progression direction, you can replace the text of the block.

Property	`axf:overflow-replace`
Value	*<string>*
Initial	empty string
Applies to	all block-level formatting objects
Inherited	yes

When `overflow` is `replace`, `axf:overflow-replace` specifies the replacement text. Only simple text can be specified.

Note

If you want to change just the end of the text, consider using `axf:text-overflow` with `overflow="hidden"`.

2.15.4 Condense overflowing content

When the overflow occurs, you can specify how and by how much to condense the content as well as the alignment of the last line. When `overflow="condense"`, overflowing text will be condensed.

Property	axf:overflow-condense
Value	[font-size | font-stretch | line-height | letter-spacing]+ | auto | none
Initial	auto
Applies to	all block-level formatting objects, inline-level formatting objects
Inherited	yes

axf:overflow-condense specifies the condensing method to use [Figure 2.15-2] (p.214).

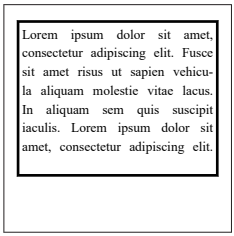
axf:overflow-condense	Block progression direction	Inline progression direction
font-size	Adjust font size.	
font-stretch	Adjust font width.	
line-height	Adjust line height.	Invalid
letter-spacing	Adjust space between characters. May cause characters to overlap or reverse their displayed positions.	
auto	font-size	font-stretch
none	Do not condense. Can also be specified for inline elements. This setting is not inherited.	

When multiple methods are specified, they are tried in order. If, for a given condensing sequence, the axf:condense-limit-* for the first method is exceeded but the content still does not fit in the area, the next method is tried starting from that state.

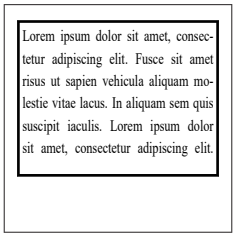
For example, for axf:overflow-consense="line-height font-stretch font-size", firstly, the line height of the block's lines is reduced to fit the content into the area. However, if axf:condense-limit-line-height="1.1" is also specified, the line height will not be reduced below 1.1. If the content then does not fit, AH XSL Formatter will try to condense further by adjusting the font width, still with line-height="1.1".

axf:overflow-condense is inherited. To partially disable this setting, specify none on a descendant element to exclude it from being condensed. You can specify none on inline elements, although this value is not inherited.

When a descendant of an inline container or a block container specifies an absolute value for a property that could be modified when condensing the container's content, if condensing is performed, that property value will not change.



axf:overflow-condense="font-size"



axf:overflow-condense="font-stretch"

Figure 2.15-2 overflow="condense"

2.15.5 Condensing limits

You can specify the lower limit and fallback behavior for each condensing method.

Property	axf:overflow-condense-limit-font-size
Value	none | [<*length*>[visible | hidden | error-if-overflow | repeat]] ?]
Initial	none
Applies to	all block-level formatting objects
Inherited	yes

axf:overflow-condense-limit-font-size specifies the minimum font size resulting from axf:overflow-condense="font-size".

Numbers less than or equal to 0 are invalid.

none is assumed to have specified 0.

Property	axf:overflow-condense-limit-font-stretch
Value	none | [[<*length*> | <*percentage*>][visible | hidden | error-if-overflow | repeat]] ?]
Initial	none
Applies to	all block-level formatting objects
Inherited	yes
Percentages	refer to the font size

axf:overflow-condense-limit-font-stretch specifies the minimum font width resulting from axf:overflow-condense="font-stretch". 1 is equivalent to 100 %.

Property	axf:overflow-condense-limit-line-height
Value	none | [<*number*>[visible | hidden | error-if-overflow | repeat]] ?]
Initial	none
Applies to	all block-level formatting objects
Inherited	yes

axf:overflow-condense-limit-line-height specifies the minimum line height resulting from overflow-condense="line-height". <*number*> is the proportion of the font size. If less than 0 is specified, it is regarded as 0.

Property	axf:overflow-condense-limit-letter-spacing
Value	none | [<*length*>[visible | hidden | error-if-overflow | repeat]] ?]
Initial	none
Applies to	all block-level formatting objects
Inherited	yes

axf:overflow-condense-limit-letter-spacing specifies the minimum space between characters resulting from axf:overflow-condense="letter-spacing".

Because <*length*> specifies the amount to condense, it is usually negative. If none is specified, it is regarded as -1em.

axf:overflow-condense-limit-* property can, in addition, include a fallback for when the lower limit is reached. The visible, hidden, error-if-overflow and repeat values from the overflow property are allowed. The fallback is ignored when axf:overflow-condense includes a subsequent condensing method.

2.15.6 Last line alignment after condensing

You can specify the alignment of the last line of condensed content.

text-align-last specifies the alignment of the last line of a block. When condensing occurs, the last line can contain a lot of text. It is possible to justify the last line if it has been condensed.

Property	axf:condensed-text-align-last
Value	false | true | justify | auto
Initial	false
Applies to	all block-level formatting objects
Inherited	yes

Option Property	condensed-text-align-last
Value	false | true | justify
Initial	false
Applies to	<formatter-settings>

`axf:condensed-text-align-last` specifies whether to perform `text-align-last="justify"` when the text is condensed.

If `false`, do nothing, and if `true`, equivalent to `text-align-last="justify"`.

If `justify`, `text-align-last="justify"` will be specified only when `text-align="justify"` is also specified on the block with `overflow="condense"`.

When `auto`, the `condensed-text-align-last` option value is used.

```
<fo:block overflow="condense" text-align="left" text-align-last="le
ft"
  <fo:block axf:condensed-text-align-last="true">
    ... <!-- During condensing process, text-align-last="justify"--
>
  </fo:block>
  <fo:block axf:condensed-text-align-last="justify">
    ... <!-- During condensing process, text-align-last="left" -->
  </fo:block>
</fo:block>
<fo:block overflow="condense" text-align="justify" text-align-last=
"left"
  <fo:block axf:condensed-text-align-last="justify">
    ... <!-- During condensing process, text-align-last="justify" -
->
  </fo:block>
</fo:block>
```

Note

When condensing causes `text-align-last="justify"`, other related settings, such as `axf:flush-zone`, also affect the result.

2.15.7 Overflow in the inline progression direction

Overflow in the inline progression direction occurs when text that cannot wrap, such as an object with `wrap-option="no-wrap"`, overflows its area [Fig. 2.15-3] (p.193). [Figure 2.15-3] (p.217) .

Property	`axf:text-overflow`
Value	`clip` | `ellipsis` | <*string*>
Initial	`clip`
Applies to	block level elements

For use with `overflow="hidden"`, `axf:text-overflow` specifies how to display the end of text that overflows [Figure 2.15-4] (p.218). `axf:text-overflow` is valid only when `overflow` is `hidden`.

`axf:text-overflow`	**Definition**
`clip`	Do nothing. As per `overflow="hidden"`, the overflowing text is not displayed.
`ellipsis`	Reduce the width of the displayed text enough to insert HORIZONTAL ELLIPSIS "…" (U+2026) at the end. The same as `clip` when the first character or image overflows.
<*string*>	Reduce the width of the displayed text enough to insert the string at the end.

If the first character or image overflows, `axf:text-overflow` is treated as a `clip`.

Note

If wrapping is prohibited and `overflow="condense"` is specified, the condensing process is applied to overflow in the inline progression direction.

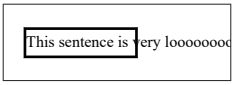


Figure 2.15-3 Overflow in the inline progression direction

Figure 2.15-4 `axf:text-overflow`

2.15.8 Aligning overflowed blocks

You can change the alignment when a block overflows.

Property	`axf:overflow-align`
Value	`normal` | `start` | `end` | `center`
Initial	`normal`
Applies to	block-level formatting objects

`axf:overflow-align` specifies the alignment when the block overflows.

`normal` uses the `text-align` value. `axf:overflow-align` has no effect when `overflow="condense"` is specified.

2.15.9 Block alignment when inline container overflows

You can change the alignment of blocks in an inline container when it overflows. When an inline container breaks across a line, its child blocks may overflow.

Property	`axf:overflow-inline-align`
Value	`normal` | `start` | `end` | `left` | `right`
Initial	`normal`
Applies to	
Inherited	yes

When a line break occurs in the inline container, the position of its block children can be adjusted by specifying `axf:overflow-inline-align`.

2.15.10 Upper limit of keep-together

You can limit the scope for `keep-together` to suppress breaks.

`keep-together` suppresses breaking when a condition is met in the current context. An FO that cannot break may overflow its display area. AH XSL Formatter lets you specify an upper limit for where a break is suppressed.

Limit keep-together for page and column breaks

An FO for which `keep-together.within-page` or `keep-together.within-column` is not `auto` may extend past the footer or the bottom edge of the page.

Property	`axf:keep-together-within-dimension`
Value	`all` | <*length*>
Initial	`all`
Applies to	block level formatting object

`axf:keep-together-within-dimension` specifies the maximum height that is kept together without breaking. Even if `keep-together="always"`, a height that exceeds the limit is treated as if `keep-together="auto"`.

Limit keep-together for line breaks

An FO for which `keep-together.within-line` is not `auto` may overflow its display area because line breaking is suppressed.

Property	`axf:keep-together-within-inline-dimension`
Value	`all` | <*length*>
Initial	`all`
Applies to	block level formatting object

When `keep-together.within-line` is not `auto`, `axf:keep-together-within-inline-dimension` property specifies the maximum width that is kept together without breaking.

When `all` is specified, there is no upper limit.

Section 2.16 Character spacing and alignment

You can precisely control spacing within and between different scripts, such as Japanese and European.

2.16.1 European punctuation marks

You can control the space before and after punctuation marks in European text.

Although European text uses a common set of punctuation marks, you may want to change their spacing based on the language.

Element	<axf:space-start-punctuation>,<axf:space-end-punctuation>
Property	language, space, code
Parent	fo:declarations

Element	<axf:space-between-digit-and-punctuation>
Property	language, space, code
Parent	fo:declarations

Element	<axf:space-between-punctuation-and-digit>
Property	language, space, code
Parent	fo:declarations

Option Element	<space-start-punctuation>, <space-end-punctuation>
Property	language, space, code
Parent	<formatter-settings>

Option Element	<space-between-digit-and-punctuation>
Property	language, space, code
Parent	<formatter-settings>

Option Element	`<space-between-punctuation-and-digit>`
Property	`language`, `space`, `code`
Parent	`<formatter-settings>`

You can specify the space around punctuation marks in European text. Use `<axf:space-start-punctuation>` for the space before a punctuation mark, `<axf:space-end-punctuation>` for the space after punctuation marks, `<axf:space-between-digit-and-punctuation>` for the space between a number and a following punctuation mark, and `<axf:space-between-punctuation-and-digit>` for the space between a punctuation mark and a following number. `<axf:space-between-*>` take precedence over `<axf:space-*-punctuation>`.

`language` specifies the language code; `space`, the space to allocate; and `code`, the target punctuation mark.

Property	`space`
Value	*<number>* | *<length>* | *<percentage>* | `figure` | `punctuation` | `thin` | `hair`
Initial	N/A
Applies to	axf:space-end-punctuation, axf:space-start-punctuation, axf:space-between-digit-and-punctuation, axf:space-between-punctuation-and-digit, space-end-punctuation, space-start-punctuation, space-between-digit-and-punctuation, space-between-punctuation-and-digit
Percentages	refer to the font size

space specifies the space to use. As well as a *<number>*, *<length>* or *<percentage>*, you can specify the keyword for a space character: figure is the width of FIGURE SPACE (U+2007), punctuation is the width of PUNCTUATION SPACE (U+2008), thin is the width of THIN SPACE (U+2009) and hair is the width of HAIR SPACE (U+200A).

Negative values are invalid.

The following example changes the space around QUESTION MARK "?" (U+003F) in French (`xml:lang="fr"`) [Figure 2.16-1] (p.222). The markup uses the same characters and spaces for French and English (`xml:lang="en"`), but the formatted output shows different space before the character.

```
<!--  Change the space around "?" in French -->
<fo:declarations>
  <axf:space-start-punctuation code="?" space="1 div 3" language="f
r"/>
```

```
</fo:declarations>
...
<fo:page-sequence ... xml:lang="en" >
  <fo:flow ...>
    <fo:block>...
      In English: What is a question mark?
      In French:
     <fo:inline font-style="italic" xml:lang="fr">
       Qu'est-ce qu'un point d'interrogation?
     </fo:inline> OK?
    </fo:block>
  </fo:flow>
</fo:page-sequence>
```

... In English: What is a question mark? In French: *Qu'est-ce qu'un point d'interrogation ?* OK?

Figure 2.16-1 French question mark space

2.16.2 Space between Japanese and European text

You can specify whether to insert a space between Japanese and European text and its width.

You may sometimes want to insert a space between adjacent Japanese and European text.

Property	`axf:text-autospace`
Value	`none` | [`ideograph-numeric` || `ideograph-alpha` || `ideograph-parenthesis`] | `auto`
Initial	`auto`
Applies to	all block-level and inline-level formatting objects
Inherited	yes

Option Property	`text-autospace`
Value	`true` | `false`
Initial	`true`
Applies to	`<formatter-settings>`

Property	`axf:text-autospace-width`
Value	*<length>* | *<percentage>* | `auto`
Initial	`auto`
Applies to	all block-level and inline-level formatting objects
Inherited	yes
Percentages	refer to the font size

Option Property	`text-autospace-width`
Value	*<length>* | *<percentage>*
Initial	`25%`
Applies to	`<formatter-settings>`
Percentages	refer to the font size

`axf:text-autospace`	**Definition**
`ideograph-numeric`	Insert a space between an ideographic character (kana or kanji) and a non-ideographic number.
`ideograph-alpha`	Insert a space between an ideographic character and a non-ideographic letter.
`ideograph-parenthesis`	Insert a space between an ideographic character and a non-ideographic parenthesis. Does not insert a space between an ideographic character and a non-ideographic close-parenthesis or between a non-ideographic open-parenthesis and an ideographic character.

`axf:text-autospace` specifies the character classes between which to insert a space, and `axf:text-autospace-width` specifies the width.

"*Ideographic characters*" are in the Unicode general category Lo (other characters including syllables and ideographs). This includes hiragana, katakana, Chinese Bopomofo, Hangul Jamo with code positions from U+3041 to U+31FF and CJK unified ideographs including extension and compatibility characters.

"*Non-ideographic numbers*" are in the Unicode general category Nd (decimal) and are not full-width. Includes DIGIT ZERO to DIGIT NINE "9" (U+0039) and script-specific decimal digits.

"*Non-ideographic letter*" is a character in the Unicode general category Lu (uppercase), Ll (lowercase), Lt (digraph with uppercase first) or Lm (modifier). These include LATIN CAPITAL LETTER A, LATIN SMALL LETTER A "a" (U+0061), LATIN CAPITAL LETTER D WITH

SMALL LETTER Z WITH CARON “Dž” (U+01C5), LATIN CAPITAL LETTER A WITH GRAVE “À” (U+00C0) and so on.

“*Non-ideographic parentheses*” refers to parentheses that are not full-width. These include LEFT PARENTHESIS “(” (U+0028), RIGHT PARENTHESIS “)” (U+0029), LEFT CURLY BRACKET “{” (U+007B), RIGHT CURLY BRACKET “}” (U+007D) and so on.

When `axf:text-autospace="auto"` is specified, the value from the `text-autospace` option or the [**Format Option Setting Dialog**] of the GUI is used. When `text-autospace="true"` is specified, “`ideograph-numeric ideograph-alpha`” is used. When `text-autospace="false"` is specified, it is considered `none`.

2.16.3 Trim full-width punctuation marks

You can trim white-space from Japanese punctuation marks that are at the beginning and end of lines next to a full-width character.

Fixed-width fonts use the same width for punctuation marks and normal Japanese characters. With AH XSL Formatter, you can control whether to trim white-space from punctuation marks.

Beginning, end and adjacent full-width punctuation marks

You may sometimes want to trim the space of the full-width punctuation marks [Figure 2.16-2] (p.224), depending on where they are displayed.

It is possible to trim the white-space from a full-width open-parenthesis at the beginning of a line, from full-width punctuation marks at the end of a line and trim the white-space from full-width punctuation marks next to full-width characters where the space looks wide.

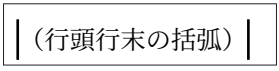


Figure 2.16-2 Text with full-width punctuation marks

Property	`axf:punctuation-trim`
Value	`none` | [[`start` | `start-except-first`] || [`end` | `allow-end` | `end-except-fullstop`] || `adjacent`] | `all` | <*string*> | `auto`
Initial	`auto`
Applies to	all block-level and inline-level formatting objects
Inherited	yes

Option Property	`punctuation-trim`
Value	`true` | `false`
Initial	`true`
Applies to	`<formatter-settings>`

`axf:punctuation-trim` specifies when to trim full-width punctuation marks in Japanese. It is ignored for proportional fonts.

`start` and `>start-except-first` set when to trim marks at the beginning of a line [Figure 2.16-3] (p.226), and `*end*` values set when to trim marks at the end of a line. [Figure 2.16-4] (p.226).

`adjacent` allows you to trim adjacent marks [Figure 2.16-5] (p.226).

If `none` is used, no full-width punctuation marks are trimmed.

`axf:punctuation-trim`	Definition
`start`	Trim full-width punctuation marks at the beginning of the line to half-width.
`start-except-first`	Trim full-width punctuation marks at the beginning of the line to half-width, except for the first line and the line immediately after the forced line break.
`end`	Trim full-width punctuation marks at the end of the line to half-width when the alignment is `right` or `justify`.
`allow-end`	When text alignment is `right` or `justify`, if the full-width punctuation mark located at the end of the line fits in the line, treat it as full-width and if it does not fit, trim it to half-width.
`end-except-fullstop`	When text alignment is `right` or `justify`, full-width punctuation marks except IDEOGRAPHIC FULL STOP "。" (U+3002) and FULLWIDTH FULL STOP "．" (U+FF0E) located at the end of the line are trimmed to half-width.
`adjacent`	Trim the space between full-width punctuation marks and full-width characters in Japanese.
`all`	Trim all full-width parentheses, middle dots and punctuation marks to half-width.
<string>	Among the characters trimmed to half-width by `all`, trim only the specified characters.
`auto`	Same as none when the `punctuation-trim` option is `false`, otherwise same as `start end adjacent`.

The punctuation-trim option also affects axf:text-justify-trim="auto".

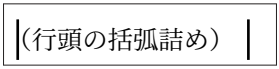


Figure 2.16-3　Trimming full-width punctuation marks at the beginning of a line

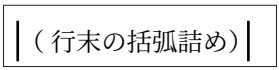


Figure 2.16-4　Trimming full-width punctuation marks at the end of a line

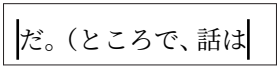


Figure 2.16-5　Trimming space between adjacent full-width punctuation marks

Trimming width of adjacent full-width punctuation marks

axf:punctuation-spacing specifies the amount of space when axf:punctuation-trim is adjacent.

Property	axf:punctuation-spacing
Value	*<length>* | *<percentage>* | auto
Initial	auto
Applies to	all block-level and inline-level formatting objects
Inherited	yes
Percentages	refer to the font size

Option Property	punctuation-spacing
Value	*<length>* | *<percentage>*
Initial	50%
Applies to	<formatter-settings>
Percentages	refer to the font size

The punctuation-spacing option specifies the amount of space to trim when axf:punctuation-spacing is auto.

2.16.4 Trimming

When the text does not fit in the available area, you may want to trim the space between characters so that the text does fit.

Property	axf:text-justify-trim
Value	none | [punctuation || punctuation-except-fullstop || punctuation-except-middledot || [kana | ideograph] || inter-word] | auto
Initial	auto
Applies to	all block-level and inline-level formatting objects
Inherited	yes

Option Property	text-justify-mode
Value	4 | 5
Initial	5
Applies to	<formatter-settings>

axf:text-justify-trim affects a wider range of characters than axf:punctuation-trim.

axf:text-justify-trim="auto" leaves the trimming to AH XSL Formatter. When the text-justify-mode option is 4, or when axf:avoid-widow-words is valid, auto is treated as ideograph and inter-word. The punctuation-trim option also applies when axf:text-justify-trim is auto.

axf:text-justify-trim	Definition
punctuation	Trim Japanese full-width parentheses, middle dots and punctuation marks.
punctuation-except-fullstop	Trim Japanese full-width parentheses, middle dots and punctuation marks except IDEOGRAPHIC FULL STOP and FULLWIDTH FULL STOP.
punctuation-except-middledot	Trim Japanese full-width parentheses and punctuation marks except KATAKANA MIDDLE DOT “・” (U+30FB), FULLWIDTH COLON “：” (U+FF1A) and FULLWIDTH SEMICOLON “；” (U+FF1B).
kana	Trim hiragana and katakana.
ideograph	Trim kanji, hiragana and katakana.
inter-word	Trim spaces between European words.

2.16.5 Letter-spacing side

You can specify where space is added when letter-spacing adds space between letters.

Property	axf:letter-spacing-side
Value	both | start | end
Initial	both
Applies to	inline-level formatting objects
Inherited	yes

both places half the space on either side of each character, start places the space on the start side and end places the space on the end side.

2.16.6 Space CJK based on character count

A fixed space between characters is common in CJK text. You can control the space between characters based on the number of characters.

This simplifies placing text in certain contexts, such as in a title.

Property	axf:auto-letter-spacing
Value	[none | <*length*> | <*percentage*>]] *
Initial	none
Applies to	inline-level formatting objects
Percentages	refer to the font size

axf:auto-letter-spacing specifies the space to add between characters for individual character counts starting at two characters.

Note

text-align, text-align-last and axf:letter-spacing-side apply to the spaces added by axf:auto-letter-spacing.

2.16.7 Justification spacing

You can control where space is added in justified text.

Property	axf:text-justify
Value	auto | inter-word | inter-character | distribute
Initial	auto
Applies to	all block-level and inline-level formatting objects
Inherited	yes

axf:text-justify	Definition
inter-word	Adjust space between words. Useful for European text.
inter-character, distribute	Adjust space between characters. Useful for Japanese. distribute accepted for compatibility.
auto	Depends on the script. Uses inter-character for kanji, hiragana and katakana, Kashida processing for Arabic and inter-word for others.

Both axf:text-justify="inter-character" and letter-spacing limit kerning and ligature processing.

2.16.8 Kashida

You can specify the relative length of a Kashida in Arabic.

In Arabic, a Kashida (line extending from the end of a letter) functions as the space between words.

Property	`axf:text-kashida-space`
Value	*<percentage>* | `auto`
Initial	`auto`
Applies to	all block-level and inline-level formatting objects
Inherited	yes

Option Property	`text-kashida-space`
Value	*<percentage>*
Initial	`100%`
Applies to	`<formatter-settings>`

`axf:text-space-kashida` specifies the proportion of white-space to use for Kashida. Specify a value from 0 % to 100 %.

0 % does not insert Kashida, and 100 % inserts Kashida as much as possible. A value outside the range is invalid.

When `axf:text-kashida-space` is `auto`, the `text-kashida-space` option value is used.

2.16.9 Tab alignment

You can specify alignment on tab stops in AH XSL Formatter.

In the XSL 1.1 specification, TAB is treated as just a white-space character, and it is either merged with adjacent white-space or left as is by specifying `white-space-collapse`.

Note

Lines containing `<axf:tab>` are forced to `text-align="start"`.
Tab alignment may not work as expected for text that requires BIDI processing.

Element	<axf:tab>
Property	axf:tab-align
Parent	fo:block, fo:inline

Property	axf:tab-align
Value	auto | start | end | left | right | decimal | <*string*>
Initial	auto
Applies to	axf:tab

Property	axf:tab-stops
Value	[[<*tab-align*>] ?[<*length*> | <*number*>]] *[<*tab-align*>] ? eol]] ?]!
Initial	8
Applies to	all block-level formatting objects
Inherited	yes

axf:tab-align and axf:tab-stops control alignment of an <axf:tab>.

If you want to align individual <axf:tab>, specify axf:tab-align on each <axf:tab>. When <*string*> is specified, the first character of the string is aligned on the tab position. decimal is equivalent to axf:tab-align=".".

When axf:tab-align="auto", axf:tab-stops on the block FO specifies multiple tab alignments at once. In addition to the axf:tab-align values, <*length*>, <*number*> and eol (which aligns to the end of the line) can be specified.

```
<!-- Define multiple tab intervals with tab-stops -->
<fo:block axf:tab-stops="eol">
  <axf:tab tab-align="auto"/>
</fo:block>
```

Overlap in tab-aligned text

Property	axf:tab-overlap-treatment
Value	auto | ignore-tab | next-tab
Initial	auto
Applies to	fo:block, fo:character
Inherited	yes

Option Property	tab-overlap-treatment
Value	ignore-tab | next-tab
Initial	ignore-tab
Applies to	<formatter-settings>

axf:tab-overlap-treatment specifies the processing when tab-aligned text overlaps.

ignore-tab shows the text immediately after the text that overlapped the tab position, and next-tab sends the text to the next tab stop.

When axf:tab-overlap-treatment is auto, the tab-overlap-treatment option value is used.

2.16.10 TAB (U+0009) processing

When TAB (U+0009), not <axf:tab>, appears in the FO, you can specify whether to treat it as a tab alignment target or as white space.

Property	axf:tab-treatment
Value	<*number*> | preserve
Initial	4
Applies to	fo:block, fo:character
Inherited	yes

axf:tab-treatment specifies the treatment of TAB.

When preserve, TAB is treated as <axf:tab>, and axf:tab-stops determines its effect.

When <*number*> is specified, it is treated as that number of SPACE(U+0020) characters. white-space-treatment and white-space-collapse apply to the replacement characters.

2.16.11 Fixed-width space glyphs

You can specify how to handle fixed-width space glyphs.

Many fonts do not provide glyphs for fixed-width white-space characters, such as EM SPACE(U+2003) and THIN SPACE(U+2009). If the fallback when these spaces are included in the text is to show an alternative glyph, the glyph may not be the correct width.

Option Property	`fixed-width-space-treatment`
Value	`false` | `true` | `always`
Initial	`true`
Applies to	`<formatter-settings>`

The `fixed-width-space-treatment` option allows you to specify how to treat of fixed-width spaces. The following table shows the affected characters. The units are em.

Character	**em**
EN QUAD(U+2000)	$\frac{1}{2}$
EM QUAD(U+2001)	1
EN SPACE(U+2002)	$\frac{1}{2}$
EM SPACE	1
THREE-PER-EM SPACE(U+2004)	$\frac{1}{3}$
FOUR-PER-EM SPACE(U+2005)	$\frac{1}{4}$
SIX-PER-EM SPACE(U+2006)	$\frac{1}{6}$
FIGURE SPACE(U+2007)	Width of DIGIT ZERO
PUNCTUATION SPACE(U+2008)	Width of FULL STOP "." (U+002E)
THIN SPACE(U+2009)	Width of `thin-space-width` option
HAIR SPACE(U+200A)	Width of `hair-space-width` option
MEDIUM MATHEMATICAL SPACE(U+205F)	$\frac{4}{18}$

When `false`, nothing is done and a blank or alternative glyph is displayed.

When `true`, inserts a space when the font does not have the target glyph. If the font includes a glyph, that will take precedence.

When `always`, the space is always inserted even if the font has the target glyph.

Option Property	`thin-space-width`
Value	*<number>*
Initial	`0.2`
Applies to	`<formatter-settings>`

Option Property	`hair-space-width`
Value	*<number>*
Initial	`0.1`
Applies to	`<formatter-settings>`

The `thin-space-width` and `hair-space-width` options specify the widths of THIN SPACE and HAIR SPACE, respectively.

These widths are used when the `fixed-width-space-treatment` option is `true` or `always`.

2.16.12 Exclude NO-BREAK SPACE (U+00A0) from justification

You can stop justification applying to NO-BREAK SPACE.

NO-BREAK SPACE is used to join words that should be kept on the same line. When `text-align="justify"` and similar specify that the line is justified, AH XSL Formatter adjusts the space between words in European text. You can control whether the adjustment includes NO-BREAK SPACE.

Property	`axf:justify-nbsp`
Value	`true` | `false`
Initial	`true`
Applies to	all formatting objects
Inherited	yes

When `true`, justification applies to NO-BREAK SPACE. When `false`, NO-BREAK SPACE will have the width of a space.

Section 2.17 Character formatting

You can control formatting, Unicode normalization, character transformation, and formatting direction change independently of font features.

2.17.1 Number formatting

`format` has been extended.

The additional tokens that can be specified are described in the manual.

A string that is not a recognized token is used to generate the values. How the string is interpreted depends on the characters used.

Control the extension and transformation of number formats

You can specify a format to use for numbers.

Property	`axf:number-transform`
Value	`none` | `kansuji` | `kansuji-if-vertical` | <*list-style-type*> | <*string*>
Initial	`none`
Applies to	all block-level and inline-level formatting objects
Inherited	yes

When `axf:number-transform` is specified on a block or inline element, number sequences in the text are transformed using the specified format.

When `none`, no transformation is applied.

When `kansuji`, number sequences are transformed to Japanese numerals. `axf:kansuji-*` specify the details. `kansuji-if-vertical` applies `kansuji` only in vertical writing.

<*list-style-type*> is: a style defined by `<axf:counter-style>`; a predefined counter style implemented in AH XSL Formatter; or `symbols()`.

<*string*> behaves the same as a character string specified for `format`.

Japanese numeral conversion and grouping

There are multiple ways to write numbers using Japanese numerals: one-to-one correspondence; adding characters such as 十, 百 and 千 to indicate the number of digits; and adding the character for a large number, such as 万 for 10,000, in the middle of long

numbers. AH XSL Formatter can insert *万*, *億*, *兆*, *京*, *垓*, *杼*, *穣*, *溝*, *澗*, *正*, *載* and *極*, in sequence, every four digits.

Extension properties control the appearance of Japanese numerals when `axf:number-transform` is `kansuji`.

Property	`axf:kansuji-style`
Value	`simple` | `grouping` | `readable`
Initial	`simple`
Applies to	all block-level and inline-level formatting objects
Inherited	yes

`axf:kansuji-style` specifies the style of Japanese numerals.

`simple` converts each digit in sequence into one Japanese numeral.

`grouping` inserts "*万億兆京垓杼穣溝澗正載*極" as grouping characters in addition to performing the `simple` conversion. Also, when all the numbers below any grouping are zero, they are discarded from the formatted number.

`readable` performs the grouping conversion and, to make more readable numbers, also inserts *十* to indicate tens, *百* to indicate hundreds, and *千* to indicate thousands. For huge numbers that would require grouping characters above *極*, those digits are converted using the simple one-to-one conversion method.

`axf:kansuji-style`	12340	6700000
`simple`	*一二三四〇*	*六七〇〇〇〇〇*
`grouping`	*一万二三四〇*	*六七〇万*
`readable`	*一万二千三百四十*	*六七十万*

Characters for Japanese numerals

Property	`axf:kansuji-letter`
Value	`kansuji` | `latin` | <*string*>
Initial	`kanji`
Applies to	all block-level and inline-level formatting objects
Inherited	yes

`axf:kansuji-letter` specifies the characters used as Japanese numerals with `axf:number-transform="kansuji"`.

The characters in `axf:kansuji-letter="<string>"` are assigned as digits from 0 to 9, in order. If fewer than ten characters, the missing characters are assigned from 〇一二三四五六七八九.

If `kanji` is specified, Japanese numerals will be used. This is equivalent to `axf:kansuji-letter="〇一二三四五六七八九"`.

If `latin` is specified, full-width Arabic digits are used. This is equivalent to `axf:kansuji-letter="０１２３４５６７８９"`.

For example, if you want to use traditional Japanese numerals, specify `axf:kansuji-letter="零壱弐参肆伍陸柒捌玖"`.

Letters for grouping Japanese numerals

Property	`axf:kansuji-grouping-letter`
Value	*<string>*
Initial	empty string
Applies to	all block-level and inline-level formatting objects
Inherited	yes

`axf:kansuji-grouping-letter` specifies the grouping letters used with `axf:number-transform="kansuji"` when `axf:kansuji-style` is `grouping` or `readable`.

The specification method is the same as `axf:kansuji-letter="<string>"`.

The empty string is equivalent to `axf:kansuji-grouping-letter="十百千万億兆京垓杼穣溝澗正載極"`.

2.17.2 Custom counter style

You can define counters as in CSS3 Counter Styles Level 3.

Element	`<axf:counter-style>`
Property	`name`, `system`, `negative`, `prefix`, `suffix`, `range`, `pad`, `fallback`, `symbols`, `additive-symbols`
Parent	fo:declarations

The `<axf:counter-style>` child of `<fo:declarations>` defines a counter style. `name` is the counter name used when specifying the counter in the FO, and `system` is the counter's system type. To extend or modify an existing counter, use `system="extends <counter style name>"`.

`negative` specifies the prefix and suffix strings for negative values. The first <*symbol*> is the prefix and the second, if present, is the suffix.

`prefix` specifies a prefix string, and `suffix` specifies a suffix.

`pad` specifies the minimum length of the result and the symbol to be used to pad it.

`range` specifies the range over which the counter style can be applied, and `fallback` specifies a fallback counter style to use when the value is outside the range or when the style cannot be applied for some other reason.

Specify `symbols` when you want to use symbols for the counter, or specify `additive-symbols` if defining a weighted counter symbol.

`name` cannot be default, `decimal` or `none`.

See the manual for details.

Weighted counter symbol: For example, Roman numerals, where more I symbols are added to make numbers from 1 to 3, the symbol IV is 4, V is 5, and so on.

Note

`prefix` and `suffix` are ignored when a counter style is used to format a number with `axf:number-transform`.

2.17.3 Normalize characters and specify exclusions

You can control Unicode character normalization and specify exclusions from normalization.

There are four types of Unicode normalization: Canonical Decomposition (NFD); Canonical Decomposition followed by Canonical Composition (NFC); Compatibility Decomposition (NFKD); and Compatibility Decomposition followed by Canonical Composition (NFKC). NFC denotes composition (C), but in reality it is decomposition followed by composition.

Property	`axf:normalize`
Value	`auto` | `none` | `nfc` | `nfkc` | `nfd` | `nfkd`
Initial	`auto`
Applies to	all block-level and inline-level formatting objects
Inherited	yes

Property	`normalize`
Value	`none` | `nfc` | `nfkc` | `nfd` | `nfkd`
Initial	`nfc`
Applies to	all block-level and inline-level formatting objects
Inherited	yes

Property	`axf:normalize-exclude`
Value	`full-composition-exclusion` | `none`
Initial	`full-composition-exclusion`
Applies to	all block-level and inline-level formatting objects

`axf:normalize` specifies the Unicode normalization form.

When `axf:normalize` is `auto`, the `normalize` option value is used.

If `axf:normalize-exclude="full-composition-exclusion"` is specified, the composition exclusion characters are not normalized.

Note

A composition exclusion character is decomposable, but it is not composed from its decomposition in normalization forms NFC and NFKC. Composition exclusions are described in UAX #15.

script-specific exclusions: Characters with a canonical decomposition from a script that usually does not use the composed forms.

Post composition version exclusions: Canonically decomposable characters added after Unicode 3.0.

Singleton exclusions: Characters that have a canonical decomposition to a different single character are automatically excluded from composition.

Non-Starter Decompositions: Characters, such as COMBINING GREEK DIALYTIKA TONOS (U+0344), for which either the character or the first character of its canonical decomposition cannot be the starting character of a combining character sequence.

The kanji in CJK Compatibility Ideographs Supplement (U+2F800–U+2FA1D) are all singletons.

2.17.4 Uppercase, lowercase, full-width and half-width conversions

You can convert characters to uppercase or lowercase and between full-width and half-width forms.

Property	`text-transform`
Value	`none` | [[`capitalize` | `uppercase` | `lowercase` | **`capitalize-lowercase`**] || [**`fullwidth`** | **`fullwidth-if-vertical`**] || **`fullsize-kana`**]
Initial	`none`
Applies to	all formatting objects
Inherited	yes

`text-transform`	**Definition**
`capitalize-lowercase`	Converts the initial letter to uppercase and the following letters to lowercase in European. Uppercase words are also converted to “uppercase + lowercase”.
`fullwidth`	Converts characters from U+0021 to U+007E (displayable ASCII characters other than spaces) to full-width.
`fullwidth-if-vertical`	Applies `text-transform="fullwidth"` only in vertical writing, such as `writing-mode="tb-rl"`.
`fullsize-kana`	Converts small hiragana such as HIRAGANA LETTER SMALL TU “っ” (U+3064) and HIRAGANA LETTER SMALL YU “ゆ” (U+3085) to normal hiragana such as HIRAGANA LETTER TU “つ” (U+3065) and HIRAGANA LETTER YU“ゆ” (U+3086). Only full-width characters are converted.

Note

`text-transform` converts characters to different code points. Use `font-variant` to change the glyph without changing the code point.

`axf:text-replace` is useful when replacing one character with another character.

2.17.5 Text replacement

You can replace runs of text in AH XSL Formatter.

Property	axf:text-replace
Value	none | [<*string*> <*string*>]+
Initial	none
Applies to	all block-level and inline-level formatting objects
Inherited	yes

axf:text-replace specifies pairs of some target text to be replaced and its replacement text. If the text is not in pairs or the text to be replaced is an empty string, no text is replaced.

The target text is found by a simple comparison that will not match text across a line break. White-space is matched after being processed by white-space-treatment and related properties.

Note

The first matching target text is used. Also, the replacement is done only once. When specifying multiple target texts that begin with the same string, specify the longer text first.

2.17.6 Character orientation in vertical writing

You can control the orientation of text in vertical writing, including making it upright or horizontal.

Upright text

When using multiple types of characters in vertical writing, there are several ways to display characters that have only horizontal glyphs, such as the European text in mixed Japanese-European sentences. A word can be tilted sideways, or each character or several characters can be rotated 90 degrees to be upright.

As well as rotating the characters, the space before and after the characters is controlled.

Property	axf:text-orientation
Value	mixed | upright | sideways-rl | sideways | none | auto
Initial	auto
Applies to	all block-level and inline-level formatting objects
Inherited	yes

Option Property	text-orientation-mode
Value	5 | 6 | 7
Initial	6
Applies to	<formatter-settings>

`axf:text-orientation` is based on [CSS3 Writing-Mode].

`mixed` rotates alphanumeric characters, such as European languages, 90 degrees clockwise.

`sideways` and `sideways-rl` rotate all characters 90 degrees clockwise.

`upright` sets alphanumeric characters to the same height as Japanese full-width characters and renders each character upright. UAX#50:Unicode Vertical Text Layout: Unicode Vertical Text Layout discusses which characters should be upright in vertical writing, and there is a corresponding explanation in Mixed Vertical Orientation (MVO). AH XSL Formatter implements MVO with some modifications.

`none` gives a result similar to `mixed` with AH XSL Formatter. Which characters to rotate is based on compatibility with older versions of AH XSL Formatter.

Some fonts have glyphs for vertical writing. Commonly used characters are punctuation marks, parentheses and small hiragana and katakana. When `upright`, punctuation marks such as parentheses are not rotated. When `sideways-*` is specified, these characters are also rotated.

When `axf:text-orientation` is `auto`, the `text-orientation-mode` option value is used. When that is `7`, it is considered `mixed`, and when it is `6` or less, it is considered `none`.

Note

Upright characters are processed with `axf:word-break="break-all"` and `hyphenate="false"`. That is, line breaks are not suppressed between any characters, and hyphens are not inserted at the breaks.

Horizontal-in-vertical composition

You can use `axf:text-orientation` to make each character upright, but it cannot rotate several horizontal characters to fit in the height of one character in vertical writing.

Formatting characters horizontally in vertical writing is called "horizontal-in-vertical composition." This can be done using the XSL 1.1 specification by, for example, using an inline container with a changed writing mode. However, it can be complicated to change the processing depending on the number of characters.

Property	`axf:text-combine-horizontal`
Value	`none` | `all` | [`digits` <*integer*> || `alpha` <*integer*> || `alphanumeric` <*integer*>]
Initial	`none`
Applies to	all block-level and inline-level formatting objects
Inherited	yes

`axf:text-combine-horizontal` controls automatic horizontal-in-vertical composition.

`axf:text-combine-horizontal`	Definition
`none`	Do not use automatic horizontal-in-vertical composition.
`all`	All of the text of the element is set horizontally.
`digits` <*integer*>	Consecutive digits less than or equal to the number of digits specified by *<integer>* are set horizontally.
`alpha` <*integer*> *<integer>*	Consecutive alphabetic characters less than or equal to the number of characters specified by *<integer>* are set horizontally.
`alphanumeric` <*integer*>	Consecutive alphanumeric characters (0-9, A-Z, a-z) less than or equal to the number of characters specified by *<integer>* are set horizontally.

`axf:text-combine-horizontal` is useful when, for example, you want to output a lot of text as horizontal-in-vertical composition in a document and specify it all at once.

2.17.7 Example: Replace some characters with characters for vertical writing

In vertical writing, AH XSL Formatter automatically changes the glyphs for Japanese parentheses and small kana. However, it does not automatically replace some special characters specific to vertical writing.
The following example shows `axf:text-replace` replacing some characters in the text with the characters for vertical writing [Figure 2.17-1] (p.246).

```
<!--  Replace some characters in horizontal text with vertical tex
t - -->
<fo:page-sequence writing-mode="tb-rl">
  <fo:flow ...>
    ...
    <fo:wrapper
      axf:text-replace="離れ離れ 離れ&#x3034;&#x3035;
      トントントン トン&#x3033;&#x3035;&#x3033;&#x3035;
      さまざま さま&#x3034;&#x3035;
      かくかく かく&#x3033;&#x3035;
      しかじか しか&#x3034;&#x3035;">
      <fo:block>離れ離れになる</fo:block>
        <fo:block>トントントン</fo:block>
        <fo:block>さまざまな試み</fo:block>
        <fo:block>かくかくしかじか</fo:block>
    </fo:wrapper>
  </fo:flow>
</fo:page-sequence>
```

The replacement is done from the preceding text. If you want replacements for *トントントントン*, *トントントン* and *トントン*, specify the text pairs in order from the longest *トントントントン*.

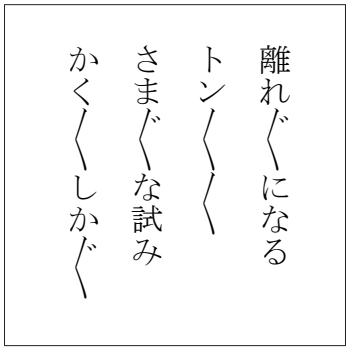


Figure 2.17-1　Character replacement in vertical text

Section 2.18 Font

You can specify ligature processing, glyph changes using font feature, and language-specific processing.

2.18.1 Font-width

`font-stretch` has been extended to allow font width to be specified as a <*percentage*> or <*number*>.

Property	`font-stretch`
Value	`normal` | `wider` | `narrower` | `ultra-condensed` | `semi-condensed` | `semi-expanded` | `expanded` | `extra-expanded` | `ultra-expanded` | <***percentage***> | <***number***>
Initial	`normal`
Applies to	all formatting objects
Inherited	yes
Percentages	the font size

<*number*> is treated as (<*number*> ×100 %). For example, to display CJK full-width characters as half-width, specify `font-stretch="0.5"`.

If you want to use the OpenType feature to select half-width glyphs, specify it with `font-variant`.

Property	`font-stretch-mode`
Value	`5` | `6`
Initial	`6`
Applies to	font-settings

The `font-stretch-mode` option controls whether to use a dedicated font, such as a condensed font, when `font-stretch` changes the font width. If the value is `6`, when font-stretch is a keyword such as condensed, the keyword influences font selection. Additionally, if font-stretch is `extra-condensed`, the condensed font will be compressed. If the value is `5`, the `font-stretch` value does not influence font selection.

2.18.2 Synthetic bold

You can specify the bold weight to use with fonts that do not have bold in their font family.

When you specify a bold font weight in AH XSL Formatter, such as `font-weight="bold"`, the bold font from the font family is used if it is available. For fonts that do not have bold weights, the bold font is synthesized by thickening the normal font by the amount determined by the system.

Option Property	`bold-ratio`
Value	<*number*>
Initial	`1.0`
Applies to	`<font-settings>`

The `bold-ratio` option specifies how thick to make a synthesized bold font. The font is displayed with the specified <*number*> times thicker.

If the value is 1.0 or is 0.0 or less, bolding is not synthesized.

2.18.3 Default font size and color

You can specify the default font size and color.

Option Property	`default-color`
Value	<*#RRGGBB*>
Initial	`#000000`
Applies to	`<formatter-settings>`

Option Property	`default-font-size`
Value	<*length*>
Initial	`10pt`
Applies to	`<formatter-settings>`

The `default-color` option specifies the color to use when the color is not known. The value can be in the *<#RRGGBB>* format, a color name, `rgb()`, or `rgb-icc()`. RGB components must be included if `rgb-icc()` is used.

The `default-font-size` option specifies the font size to use when the font size is not known.

2.18.4 Default CJK language for unified ideographs

Some characters, such as CJK unified ideographs, are used with multiple CJK languages. You can specify the language to use when the CJK script is not defined.

Option Property	`default-CJK`
Value	*<string>*
Initial	*empty*
Applies to	`<formatter-settings>`

The `default-CJK` option specifies the default language code for CJK characters. When nothing is specified, it is determined from the operating environment. Japanese is assumed in operating environments other than CJK.

This can be specified in the GUI as [**Format**]>[**Format Option Setting**]>[**Default CJK Language**].

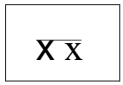
2.18.5 Font size adjustment by font name

`font-size-adjust` specifies a consistent x-height. This is used, for example, when formatting a mixture of fonts.

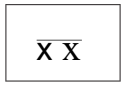
In XSL 1.1, `font-size-adjust` is a *<number>* that is a proportion of the x-height of the first choice font. AH XSL Formatter extends this so that a font name can be specified.

Property	`font-size-adjust`
Value	`none` | *<number>* | *<string>*
Initial	`none`
Applies to	all formatting objects
Inherited	yes

The named font provides the x-height to use. If the specified font does not exist, or if the font does not have x-height information, it is treated as `none`. [Figure 2.18-1] (p.250) shows the effect of specifying `font-size-adjust="Times New Roman"` for an Arial *x* and a Times New Roman .



none



'Times New Roman'

Figure 2.18-1 font-size-adjust

2.18.6 Kerning

You can control kerning.

There are combinations of letters, such as “V” and “A”, where their shape makes it look like there is a wide space between them. Pair kerning is a kerning process that works for these specific character combinations.

Property	axf:kerning-mode
Value	none | pair | auto
Initial	auto
Applies to	all block-level and inline-level formatting objects
Inherited	yes

Option Property	pair-kerning
Value	true | false
Initial	auto
Applies to	<formatter-settings>

axf:kerning-mode	Definition
none	Do not process kerning.
pair	Do pair kerning processing.
auto	Depends on system defaults.

When axf:kerning-mode is auto, the pair-kerning option controls whether to perform pair kerning. When true is specified, pair kerning is performed.

letter-spacing determines overall letter spacing adjustment. However, axf:kerning-mode does not work when letter spacing is specified.

Also, setting kern or vkrn in axf:font-feature-settings has priority over axf:kerning-mode.

2.18.7 Glyphs and features

You can specify ligature processing and glyphs.

Property	`font-variant`
Value	`normal` | **none** | [<*font-variant-caps*> || <*font-variant-numeric*> || <*font-variant-alternates*> || <*font-variant-ligatures*> || <*font-variant-alternates*> || <*font-variant-position*> || <*font-variant-east-asian*>]
Initial	`normal`
Applies to	all elements
Inherited	yes

AH XSL Formatter extends `font-variant` to support font features from [CSS3-Fonts] Overall shorthand for font rendering: the font-variant property Overall shorthand for font rendering: the `font-variant` property. You can use OpenType font features to specify, for example, old-style numbers and old Japanese fonts.

For fonts that do not have a feature, `small-caps` and `all-small-caps` are emulated and displayed, and all other unsupported features are ignored.

When `none` is specified, <*font-variant-ligatures*> is `none`, and no other `font-variant` values are considered to be specified.

Note

If you specify `titling-caps` for a heading but the font does not support it, the text will be displayed using normal glyphs.

`font-variant` has multiple independent components. Details of each setting will be explained in individual sections.

2.18.8 Capital letters in European text

You can specify the small capitals and related features.

Small capitals (small-caps) are uppercase letters that typically are the height of the x-height of lowercase letters. They are used, for example, following an uppercase initial letter so that there is no height difference with the main text (lowercase letters).

Petite-caps are usually even smaller than small-caps and, like small-caps, look like capitals.

Uppercase glyphs are usually intended to be used with lowercase letters. The text in European headings may consist of only uppercase letters, but normal uppercase letters can give the impression of being too bold.

Titling-caps are uppercase letters designed for headings.

Value	Value Composition
<font-variant-caps>	`small-caps` | **`all-small-caps`** | **`petite-caps`** | **`all-petite-caps`** | **`titling-caps`** | **`unicase`**

Specify `small-caps` for small capitals glyphs. In OpenType, this is the `smcp` feature.

`small-caps` affects only lowercase letters, while `all-small-caps` displays all text, including uppercase letters, as small capitals. In OpenType, this is the `c2sc` feature (or `smcp` feature).

Specify `petite-caps` for petite caps glyphs. In OpenType, this is the `pcap` feature.

Specifying `all-petite-caps` changes all glyphs to petite-caps, similarly to `all-small-caps`. This is the `c2pc` feature (or `pcap` feature).

`unicase` changes uppercase letters to small capitals and lowercase letters to normal glyphs. The OpenType feature is `unic`.

`titling-caps` specifies titling-cap glyphs. The OpenType feature is `titl`.

Small-caps emulation size

Small capital glyphs are normally used for `small-caps` or `all-small-caps`. When the font does not have glyphs for small capitals, small-caps are emulated by displaying reduced-size uppercase glyphs.

Option Property	`small-caps-emulation-size`
Value	*<percentage>*
Initial	`70%`
Applies to	`<formatter-settings>`
Percentages	the font size

The `small-caps-emulation-size` option specifies this reduction ratio. *<percentage>* is based on the current font size.

Option Property	`small-caps-emulation-x-height`
Value	`true` | `false`
Initial	`true`
Applies to	`<formatter-settings>`

When emulating small capitals, you can emulate a more suitable size by using the x-height and cap-height information of the font rather than a uniform ratio setting.

When `small-caps-emulation-x-height` is `true` and the font defines x-height and cap-height, the x-height/cap-height ratio is used when emulating small capitals. If the font does not have these values, the `small-caps-emulation-size` value is used.

Always emulate small capitals

Option Property	`small-caps-emulation-always`
Value	`true` | `false`
Initial	`false`
Applies to	`<formatter-settings>`

When the `small-caps-emulation-always` option is `true`, small capitals are always emulated regardless of font information.

2.18.9 Glyphs for numbers, ordinals and fractions

You can specify the glyph type for numbers.

Value	Value Composition
<font-variant-numeric>	<*numeric-figure-values*> || <*numeric-spacing-values*> || <*numberic-fraction-values*> || `ordinal` || `slashed-zero`
<*numeric-figure-values*>	`lining-nums` | `oldstyle-nums`
<*numeric-spacing-values*>	`proportional-nums` | `tabular-nums`
<*numeric-fraction-values*>	`diagonal-fractions` | `stacked-fractions`

`lining-nums` are number glyphs with the same height, whereas `oldstyle-nums` are numbers with differing heights like the lowercase alphabet.

Digits can have different widths with `proportional-nums`, but not with `tabular-nums`.

The uniform width of numbers with `tabular-nums` means that numbers in a table are aligned and easier to compare.

`diagonal-fractions` draws a diagonal line between the denominator and the numerator, whereas `stacked-fractions` draws a horizontal line between the denominator and the numerator.

`ordinal` displays an ordinal glyph, such as "1st", and `slashed-zero` draws a diagonal line through the center of "0" to make it easier to distinguish from LATIN CAPITAL LETTER O "O" (U+004F).

2.18.10 Ligatures

You can control which scripts can form ligatures.

Ligatures for scripts

Property	`axf:ligature-mode`
Value	`none` | [`latin` || `kana` || `jamo`] | `all` | `auto`
Initial	`auto`
Applies to	all block-level and inline-level formatting objects
Inherited	yes

Option Property	`latin-ligature`
Value	`true` | `false`
Initial	`true`
Applies to	`<formatter-settings>`

Option Property	`jamo-ligature`
Value	`true` | `false`
Initial	`true`
Applies to	`<formatter-settings>`

`axf:ligature-mode` specifies the targets for ligature creation. When `none`, no ligatures are formed. If the font does not include ligatures, none can be formed.

`axf:normalize` controls whether to use Unicode canonical composition to also form some ligatures.

axf:ligature-mode	Definition
latin	Form ligatures when the script is Latn, Grek or Cyrl.
kana	Form the kana ligatures in JIS X 0213:2004 plus ligatures of COMBINING KATAKANA-HIRAGANA SEMI-VOICED SOUND MARK (U+309A).
jamo	Form ligatures of the Hangul Jamo from U+1100 to U+11FF.
all	Equivalent to "latin kana jamo".
auto	Form kana ligature. Other ligatures depend on option settings.

When axf:ligature-mode is auto, option settings control whether to form latin and jamo ligatures. If the latin-ligature option is true, latin ligatures are formed, and if the jamo-ligature option is true, jamo ligatures are formed.

Ligature types

There are several types of ligatures. axf:ligature-mode controls which languages can form ligature. font-variant can enable or disable different types of ligature.

Value	Value Composition
<common-lig-values>	common-ligatures | no-common-ligatures
<histrical-lig-values>	historical-ligatures | no-historical-ligatures
<discretionary-lig-values>	discretionary-ligatures | no-discretionary-ligatures
<contextual-values>	contextual | no-contextual

common-ligatures enables common ligatures such as fi and ffi.

historical-ligatures enables historical ligatures such as tz in German.

discretionary-ligatures enables any additional ligatures designed by the type designer.

contextual enables ligatures that change the characters used depending on the context, such as the connected parts of cursive characters.

Specifying no-* disables that type of ligature.

[Figure 2.18-2] (p.256) shows a *<common-lig-values>* example.

```
<!-- common-lig-values -->
<fo:block font-variant="common-ligatures">difficult</fo:block>
<fo:block font-variant="no-common-ligatures">difficult</fo:block>
```

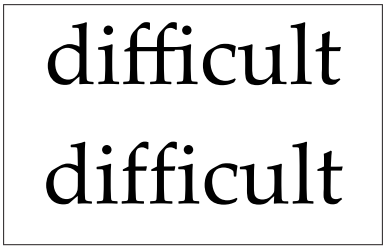


Figure 2.18-2 Using common-ligatures

2.18.11 Decorative glyphs

You can use `font-variant` to enable the use of decorative or alternate glyphs.

Variant characters in CJK are handled by <*font-variant-east-asian*>.

Value	Value Composition
<***font-variant-alternates***>	`historical-forms` | `stylistic()` | `swash()` | `ornament()` | `annotation()` | `styleset()` | `character-variant()`

Function Name	`stylistic()`, `swash()`, `ornament()`, `annotation()`
Argument Value	<*number*>

Function Name	`styleset()`, `character-variant()`
Argument Value	<*number*>`#`

`historical-forms` specifies historical glyphs. In OpenType, this is the `hist` feature.

Values other than historical-forms, take a <*number*> argument corresponding to the OpenType <*feature-index*>. To specify a stylistic glyph, specify <*feature-index*> for `stylistic()`. In OpenType, this is the `salt` feature.

`swash(`<*feature-index*>`)` selects a glyph with a more flowing shape. In OpenType, this is the `cswh` feature.

`ornament(`<*feature-index*>`)` selects a symbolic glyph such as an ornament or fleuron. It is used for bullet points and paragraph start symbols. In OpenType, this is the `ornm` feature.

`annotation(`<*feature-index*>`)` selects a glyph such as a circled number or mirror writing. In OpenType, this is the `nalt` feature.

`styleset(`<*feature-index*>`+)` specifies to use one or more sets of stylistic glyph replacements that are predefined in the font. In OpenType, this is the `ss` feature.

`character-variant(`<*feature-index*>`)` selects a specific variant glyph. In OpenType, this is the `cv` feature.

2.18.12 Glyphs for superscripts and subscripts

You can use `font-variant` to enable specific glyphs for superscripts and subscripts.

Value	Value Composition
<font-variant-position>	`super` | `sub`

`super` selects glyphs for superscripts and `sub` selects glyphs for subscripts.

2.18.13 Half-width and ruby glyph variants and traditional or simplified CJK characters

You can use `font-variant` to specify CJK variants and half-width characters.

Value	Value Composition
<font-variant-east-asian>	*<east-asian-variant-values>* || *<param.east-asian-width-values>* || `ruby`
< east-asian-variant-values >	`jis78` | `jis83` | `jis90` | `jis04` | `hojo-kanji` | `nlckanji` | `simplified` | `traditional`
<east-asian-width-values>	`full-width` | `half-width` | `third-width` | `quorter-width` | `proportional-width`

`jis78`, `jis83`, `jis90` and `jis04` specify variant glyphs defined in JIS 1978, JIS 1983, JIS 1990 and JIS 2004, respectively.

[Figure 2.18-3] (p.257) shows JIS 1978 variant characters used for part of the text.

```
<!-- Specify font-variant="jis78"to the part of the text -->
<fo:block font-fmaily="'Noto Serif CJK Jp'"
  >繋という字は JIS78 では<fo:inline
  font-variant="jis78">繋</fo:inline>と書く。</fo:block>
```

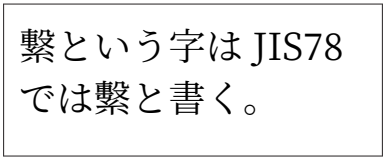


Figure 2.18-3 Specify `font-variant="jis78"` to the part of the text

In addition, `hojo-kanji` selects supplementary kanji , and `nlckanji` selects NLC kanji. `ruby` selects glyphs designed for ruby.

For Chinese fonts, `simplified` selects Simplified Chinese, and `traditional` selects Traditional Chinese.

`full-width`, `half-width`, `third-width`, `quarter-width` and `proportional-width` specify CJK character width.

2.18.14 OpenType font features

You can directly specify OpenType font features such as GSUB and GPOS.

OpenType fonts can store a variety of formatting-related features in the font itself. AH XSL Formatter uses these features during formatting, but `axf:font-feature-settings` allows users to directly specify them.

Property	`axf:font-feature-settings`
Value	`normal` | *<feature-tag-value>*#
Initial	`normal`
Applies to	all elements
Inherited	yes

`normal` does nothing.

The format of *<feature-tag-value>*, which specifies an OpenType GSUB/GPOS tag, is:

Value	Value Composition
<feature-tag-value>	*<string>*[*<integer>* | `on` | `off`]?

<string> is a four-character OpenType tag name consisting of ASCII characters from SPACE (U+0020) to TILDE "~" (U+007E). Tags with fewer than four characters are padded with SPACE.

<integer> is an integer greater than or equal to `0` that represents the valid/invalid status of the glyph's selection index or feature. `1` or `on` enables the feature, whereas `0` or `off` disables it. If omitted, `1` is assumed.

`ccmp`, `locl`, `rlig`, `mark` and `mkmk` are always valid regardless of the `axf:font-feature-settings` value.

Note

`axf:font-feature-settings` takes precedence over `font-variant`, `axf:ligature-mode` and `axf:kerning-mode`.

2.18.15 WOFF and WOFF2 font, etc.

You can add web fonts and other fonts that are not on the font path of the formatting environment to the available fonts.

Element	`<axf:font-face>`
Property	`src`, `font-family`, `font-style`, `font-weight`
Parent	fo:declarations

`<axf:font-face>` specifies the font to add.

`src` specifies the font resource to add.

`font-family` is the font name to use when selecting the added font in the FO.

`font-style` is the font style to use when selecting the font. The value is `normal` or `italic`.

`font-weight` is the font weight to use when selecting the font. The value is `normal`, `bold` or a number from 1 to 1000.

Property	`src`
Value	[<*uri-specification*>[`format(` <*font-type*>`)`]]? | <*font-face-name*>]#
Initial	N/A
Applies to	axf:font-face

<*uri-specification*> is `url()` containing the URI. The data scheme can be used. `format()` explicitly specifies the font type. If omitted, the type is determined by from the extension or the media type of the data scheme. Allowed values are `truetype`, `opentype`, `woff` and `woff2`.

<*font-face-name*> is `local()` containing the name of a local font, which can be the font family name from another `<axf:font-face>`.

2.18.16 Oblique skew

You can change the skew of synthesized oblique fonts.

Option Property	`oblique-skew`
Value	<*number*>
Initial	`0`
Applies to	`<formatter-settings>`

<number> specifies the skew angle when the font style is `oblique` or `backslant`. 0 or less is regarded as the system default value.

Note

Whenever the font family style is `italic` and the font does not have italics, it is skewed by the system default value and this option setting is ignored.

If you want to skew at a specific angle, for example, you can specify `axf:transform="skew()"` on the block to pseudo-transform the text.

`oblique-skew` is valid for GUI, PDF, PostScript and XPS output.

2.18.17 Font family syntax check

XSL 1.1 specifies that a font family that contains whitespace should be quoted by enclosing the name in either APOSTROPHE "'" (U+0027) or QUOTATION MARK '"' (U+0022). In AH XSL Formatter, you can control whether to adhere to this restriction.

Option Property	`family-name-syntax`
Value	`strict` | `auto` | `loose`
Initial	`auto`
Applies to	`<formatter-settings>`

When the `family-name-syntax` option is `loose` or `auto`, a font family name that does not match the strict rules may be acceptable.

When `strict` is specified, a name that does not conform is treated as an error.

2.18.18 Script-specific generic fonts

AH XSL Formatter sets the default generic fonts based on the OS. The generic fonts can be changed in [**Format Option Setting...**] in the GUI as well as in the option settings.

Option Element	`<script-font>`
Property	`script`, `serif`, `sans-serif`, `cursive`, `fantasy`, `monospace`, `fallback`
Parent	`<font-settings>`

`<script-font>` has a `script` property that specifies the script code, and `serif`, `sans-serif`, `cursive`, `fantasy` and `monospace` properties that specify the generic fonts for that script code. `fallback` specifies the font to use when any generic font is not available.

Specifying an empty string for `script` sets the default generic fonts.

If you want to specify *源ノ明朝* font for `serif` and *源ノ角ゴシック* font for `sans-serif` in Japanese text, describe it as follows:

```
<!--  Japanese generic font -->
<formatter-config>
  ...
  <font-settings>
    <script-font script="Jpan"
      serif="源ノ明朝" sans-serif="源ノ角ゴシック"/>
    ...
  </font-settings>
</formatter-config>
```

2.18.19 Font fallback and error reporting

AH XSL Formatter provides option settings to control font fallback processing when a specified font or glyph is not found. There are separate controls for fallback processing and error reporting level.

The options that set an error reporting level do not report an error when the value is `0`. A value of `1` or more will be reported as that error level.

Option Property	`auto-fallback-font`
Value	`true` | `false`
Initial	`true`
Applies to	`<font-settings>`

The `auto-fallback-font` option controls whether to look for fallback fonts when no glyph is found in the specified font family. AH XSL Formatter searches for fallback fonts when `true` is specified.

Option Property	`fallback-glyph`
Value	`0` | `1` | `2` | `3` | `4`
Initial	`1`
Applies to	`<font-settings>`

The `fallback-glyph` option sets the error level when `auto-fallback-font` is `true` and font fallback occurs.

Option Property	missing-font
Value	0 | 1 | 2 | 3 | 4
Initial	1
Applies to	<font-settings>

The missing-font option sets the error level when a specified font is not found.

Option Property	missing-glyph
Value	0 | 1 | 2 | 3 | 4
Initial	1
Applies to	<font-settings>

The missing-glyph option sets the error level when the glyph of a specified character is not found in the specified font family or fallback font.

Option Property	missing-glyph-all
Value	true | false
Initial	false
Applies to	<font-settings>

Errors from missing-glyph and fallback-glyph are usually reported only once for each character. If missing-glyph-all is true, every missing glyph is reported.

Report emulated italics and small-caps

When font-style is italic but there is no corresponding font, AH XSL Formatter mechanically skews the font to emulate italics.

When font-variant is small-caps but there are no small-capital glyphs, AH XSL Formatter emulates small capital using uppercase glyphs.

Option Property	emulated-italic
Value	0 | 1 | 2 | 3 | 4
Initial	0
Applies to	<font-settings>

The emulated-italic option sets the error level when an italic face is emulated.

Option Property	emulated-small-caps
Value	0 | 1 | 2 | 3 | 4
Initial	0
Applies to	<font-settings>

The emulated-small-caps option sets the error level when small capitals are emulated.

2.18.20 Substitute font family

Sometimes a font specified in the FO does not exist in the environment in which AH XSL Formatter is running. Instead of either changing the FO or using font fallback, it is possible to specify an alternative font in the option settings.

Element	<font-alias>
Property	src, dst
Parent	font-settings

font-alias describes a font family alias.

src specifies the font name used in FO, and dst specifies its replacement.

For example, when the FO includes "font-family="Futura"" but AH XSL Formatter is used in an environment that does not have the "Futura" font, you can replace the corresponding part with the "Times New Roman" font. Below shows the example of replacing the font:

```
<!-- FO -->
<fo:block font-family="Futura">...</fo:block>
```

```
<!-- Option Setting -->
<font-settings><font-alias src="Futura" dst="Times New Roman">
</font-settings>
```

Note

Replacements are not chained: the replacement of src with dst does not continue if the dst value is specified for the src property of another <font-alias>.

Section 2.19　Ruby and emphasis marks

Finely control ruby and emphasis marks.

Ruby is a small character that is attached to the side of the text to indicate reading or meaning. The target to which ruby is added is called its base character. For more information about ruby, see the following references:

For more information about ruby, see the following references:

- 日本語組版処理の要件 （W3C 技術ノート 2012 年 4 月 3 日）
- JIS X 4051『日本語文書の組版方法』
- CSS Ruby Layout Module Level 1 W3C Working Draft, 5 August 2014
- Ruby Annotation W3C Recommendation 31 May 2001 （Markup errors corrected 25 June 2008）

AH XSL Formatter implements ruby according to the 日本語組版処理の要件 （W3C 技術ノート 2012 年 4 月 3 日） .

2.19.1　Ruby markup

By marking up ruby using extension elements, detailed adjustment is possible.

If you want to reproduce ruby fusing only the XSL 1.1 specification, you can, for example, use an inline container and place lines with reduced font size. However, with this method, it is very difficult to properly perform line breaking, handle ruby at the beginning and end of lines or adjust the length of ruby and base characters.

Element	`<axf:ruby>`
Property	`axf:ruby-align`, `axf:ruby-position`, `axf:ruby-offset`, `axf:ruby-overhang`, `axf:ruby-limit-overhang`, `axf:ruby-limit-space`, `axf:ruby-small-kana`, `axf:ruby-font-family`, `axf:ruby-minimum-font-size`, `axf:ruby-font-style`, `axf:ruby-font-weight`, `axf:ruby-font-stretch`, `axf:ruby-condense`, `axf:ruby-font-color`
Parent	block-level and inline-level elements

Element	`<axf:ruby-base>`
Parent	axf:ruby

Element	`<axf:ruby-text>`
Parent	axf:ruby

To use ruby, firstly enclose the range to set the ruby with <axf:ruby>, mark up the base character as the content of its <axf:ruby-base> child element, and mark up the ruby as the content of <axf:ruby-text> child element.

Each <axf:ruby-base> and <axf:ruby-text> pair is called a *ruby container*. Multiple ruby containers can be marked up in the same <axf:ruby>. In addition, line breaking is possible between ruby containers.

In the following example, “first base character” plus “first ruby” and “second base character” plus “second ruby” are two ruby containers.

```
...
<axf:ruby><axf:ruby-base>親文字 1 つめ</axf:ruby-base>
  <axf:ruby-text>first ruby</axf:ruby-text>
  <axf:ruby-base>second base character</axf:ruby-base>
  <axf:ruby-text>second ruby</axf:ruby-text>
</axf:ruby>...
```

All ruby-specific properties are specified on <axf:ruby>.

Note

<axf:ruby-text>要素 can only contain text. Therefore, complicated formatting that requires nesting, such as smaller ruby whose base character is also ruby cannot be processed.

2.19.2 Ruby alignment

You can specify whether to align ruby as justified or solid setting and to align it with the start or end in vertical layout..

Property	axf:ruby-align
Value	auto | [[space-between | space-around] || [center | start | end | start-if-vertical]]
Initial	auto
Applies to	axf:ruby
Inherited	yes

Option Property	ruby-align
Value	[space-between | space-around] || [center | start | end | start-if-vertical]
Initial	space-around center
Applies to	<formatter-settings>

Ruby alignment can be specified with the axf:ruby-align property.

The space-* values format the ruby as justified, and the remainder format as solid setting. space-around includes spaces before and after the ruby, while space-between does not as it aligns the beginning and end of the ruby with the edges of the base character.

start aligns to the start side, center aligns to the center and end aligns to the end side with solid setting. start-if-vertical behaves the same as start in vertical layout and center in horizontal layout.

Also, if two values are specified, such as space-around center, when the ruby is only one character, it will be aligned in the second specified direction.

For example, if you ordinarily justify ruby with spaces before and after, but if for one ruby character, you want the ruby to be start-aligned in vertical layout or center-aligned in horizontal layout specify axf:ruby-align="space-around start-if-vertical".

2.19.3 Ruby position

Specify on which side of the base character to place the ruby and how much the position is moved.

Property	axf:ruby-position
Value	before | after | [[over | under | ~~inter-character~~] || [right | left]]
Initial	before
Applies to	axf:ruby
Inherited	yes

axf:ruby-position specifies where to position the ruby relative to the base character. This property is based on Ruby positioning: the 'ruby-position' property in [CSS3-Ruby].

In horizontal layout, before and over position the ruby above the base character, whereas after and under position the ruby below the base character.

In vertical layout, before and right position the ruby on the right side of the base character, whereas after and left position the ruby on the left side of the base character.

`inter-character` is used for ruby in Taiwanese, but it is not supported by AH XSL Formatter.

Property	`axf:ruby-offset`
Value	*<number>* | *<length>* | *<percentage>*
Initial	`0pt`
Applies to	axf:ruby
Inherited	yes
Percentages	refer to the font size

`axf:ruby-offset` specifies the distance between ruby and its base character.

2.19.4 Emphasis marks

You can place emphasis marks on the side of the Japanese text.

Mainly used in Japanese, *emphasis marks* are small symbols placed next to letters to add emphasis. CJK, Latn, Cyrl, Grek and Zyyy scripts can have emphasis marks.

Emphasis marks can be added to ruby base characters, but not to ruby text.

Emphasis mark symbol

Property	`axf:text-emphasis-style`
Value	`none` | [[`filled` | `open`] || [`dot` | `circle` | `double-circle` | `triangle` | `sesame`]] | *<string>*
Initial	`none`
Applies to	all elements
Inherited	yes

`axf:text-emphasis-style` specifies the symbols and characters to be used as emphasis marks.

You can specify a symbol name, such as `sesame` and `dot`, plus `filled` to select the solid variant or `open` to select the outline variant [Figure 2.19-1] (p.268), or a specific string can be used.

When `none` is specified, no emphasis marks are added.

axf:text-emphasis-style	filled	open
dot	BULLET “•” (U+2022)	WHITE BULLET “ ◦ ” (U+25E6)
circle	BLACK CIRCLE “●” (U+25CF)	WHITE CIRCLE “◯” (U+25CB)
double-circle	FISHEYE “◉” (U+25C9)	“◎” (U+X25CE)
triangle	BLACK UP-POINTING TRIANGLE “▲” (U+25B2)	WHITE UP-POINTING TRIANGLE “△” (U+25B3)
sesame	SESAME DOT “﹅” (U+FE45)	WHITE SESAME DOT “﹆” (U+FE46)

dot filled(Upper), open(Lower)

sesame filled(Left), open(Right)

Figure 2.19-1 Examples of axf:text-emphasis-style

Emphasis mark font

Property	axf:text-emphasis-font-family
Value	[*<family-name>* | *<generic-family>*]#
Initial	empty string
Applies to	emphasis elements
Inherited	yes

Using axf:text-emphasis-font-family to select specific font that is the correct size for emphasis marks, can save you the trouble of adjusting the size.

Property	axf:text-emphasis-color
Value	*<color>*
Initial	currentColor
Applies to	emphasis elements
Inherited	yes

Property	`axf:text-emphasis-font-size`
Value	*<number>* | *<absolute-size>* | *<relative-size>* | *<length>* | *<percentage>*
Initial	`0.5`
Applies to	emphasis elements
Inherited	yes
Percentages	refer to the font size

Property	`axf:text-emphasis-font-style`
Value	`normal` | `italic`
Initial	*empty*
Applies to	emphasis elements
Inherited	yes

Property	`axf:text-emphasis-font-weight`
Value	`normal` | `bold` | `bolder` | `lighter` | `1... 1000`
Initial	*empty*
Applies to	emphasis elements
Inherited	yes

Property	`axf:text-emphasis-font-stretch`
Value	`normal` | `wider` | `narrower` | `ultra-condensed` | `extra-condensed` | `condensed` | `semi-condensed` | `expanded` | `extra-expanded` | `ultra-expanded` | *<percentage>* | *<number>*
Initial	*empty*
Applies to	emphasis elements
Inherited	yes
Percentages	refer to the text-emphasis-font-size

Using `axf:text-emphasis-font-*` properties is the same as for the ruby properties, but they allow emphasis marks to be styled independently from ruby.

Characters to which to add emphasis marks

You can make decisions such as “whether to add emphasis marks to punctuation marks” or “whether to add emphasis marks to European languages.”

Property	axf:text-emphasis-skip
Value	none | [spaces || punctuation || symbols || narrow]
Initial	spaces
Applies to	all elements
Inherited	yes

Character classes specified in axf:text-emphasis-skip do not show emphasis marks [Figure 2.19-2] (p.270). spaces excludes emphasis marks from applying to white-space characters, punctuation excludes punctuation marks, symbols excludes symbols characters and narrow excludes that are not full width (half-width characters.)

```
<!-- Do not add emphasis marks to European and punctuation mark
s  -->
<fo:block>
  <fo:inline axf:text-emphasis-font-size="0.5"
    axf:text-emphasis-font-family="'KentenGeneric'"
    axf:text-emphasis-style="filled dot"
    axf:text-emphasis-font-style="normal"
    axf:text-emphasis-skip="punctuation narrow"
    >この文は<fo:inline font-style="italic"
    >emphasize</fo:inline>されています。</fo:inline>
</fo:block>
```

この文章全体が *emphasize* の対象です。

Figure 2.19-2　Emphasis marks except on punctuation marks and European text

Emphasis marks position

Property	`axf:text-emphasis-position`
Value	`before` | `after`
Initial	`before`
Applies to	all elements
Inherited	yes

`axf:text-emphasis-position` specifies the direction in which the emphasis marks are positioned.

Property	`axf:text-emphasis-offset`
Value	*<number>* | *<length>* | *<percentage>*
Initial	`0pt`
Applies to	all elements
Inherited	yes
Percentages	refer to the font size

`axf:text-emphasis-offset` specifies a relative offset for positioning the emphasis mark. This is useful, for example, to avoid overlapping ruby when using both emphasis marks and ruby.

2.19.5 Ruby examples

You can use the ruby extensions to format different types of ruby.

An example of formatting ruby using ruby extension is shown in [Figure 2.19-3] (p.273) shows an example of formatting ruby using ruby extensions. Start-aligned and center-aligned ruby are concepts from mono-ruby, but are also applied to group ruby here. The names used here may differ from the JLReq terminology.

Name	Definition
Mono-ruby	Place one ruby character in one base character. This can be regarded as group ruby with a single base character.
Group-ruby	Ruby is positioned over multiple base characters.
Jukugo-ruby	Ruby with multiple ruby containers. Line breaking between ruby containers is possible.
One-third sized ruby	The size of ruby in the start–end direction is specified as one-third of the base character.
Start-aligned ruby	Place ruby on the start side.
Center-aligned ruby	Place ruby in the center.
Opposite-side ruby	Place ruby on the after side.
Both sides ruby	Place ruby on both before and after sides of the base character.

Jukugo-ruby markup can be complicated, especially when the ruby is longer than the base character.

```
<!-- Start-aligned ruby -->
<axf:ruby axf:ruby-align="start">
  <axf:ruby-base>齟</axf:ruby-base>
  <axf:ruby-text>そ</axf:ruby-text>
  <axf:ruby-base>齬</axf:ruby-base>
  <axf:ruby-text>ご</axf:ruby-text>
</axf:ruby>
```

```
<!-- Both sides ruby -->
<!-- Specify ruby on the before side -->
<axf:ruby axf:ruby-position="before">
  <axf:ruby-base>
    <!-- Specify ruby on the after side  -->
    <axf:ruby axf:ruby-position="after">
      <axf:ruby-base>薦擧</axf:ruby-base>
      <!-- Content of ruby on the after side -->
      <axf:ruby-text>スヽメアゲ</axf:ruby-text>
    </axf:ruby>
  </axf:ruby-base>
  <!-- Content of ruby on the before side -->
  <axf:ruby-text>センキョ</axf:ruby-text>
</axf:ruby>シ
```

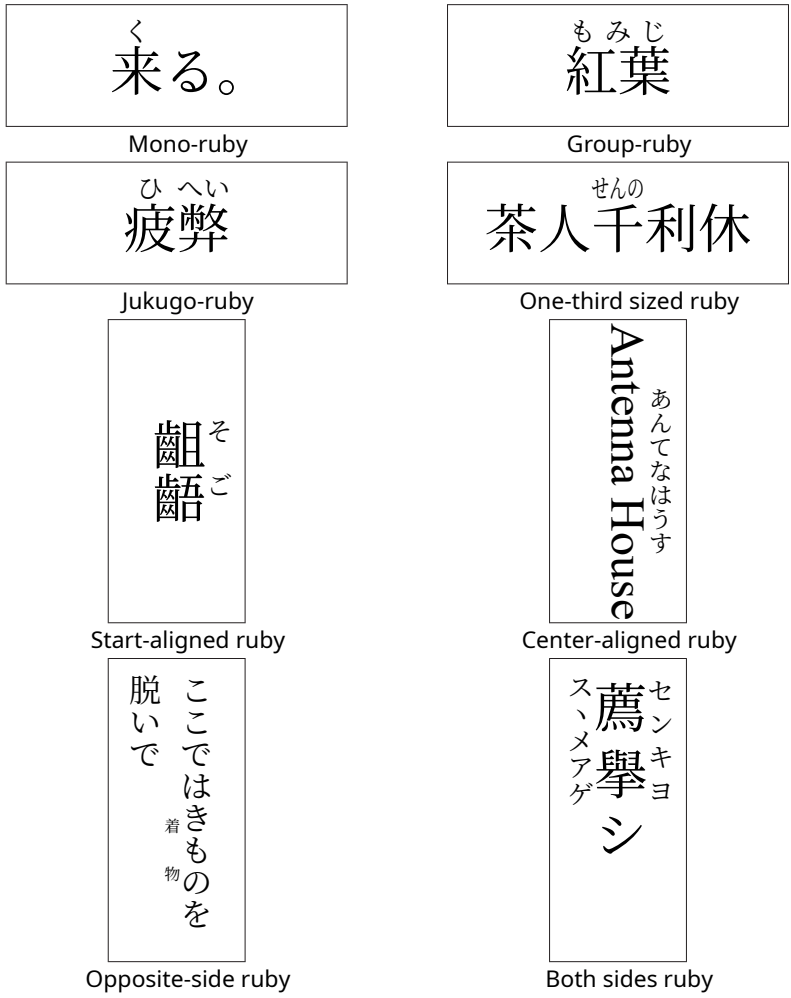


Figure 2.19-3 Ruby examples

2.19.6 Ruby on large base characters

You can use separate font styles for base characters and ruby.

It can be inappropriate for ruby on large text, such as a heading, to inherit its font styles. The ruby shown in [Figure 2.19-4] (p.274) uses explicit settings for its font size, weight and stretch.

Figure 2.19-4　Ruby on large base characters

```
<!-- Specify ruby on large base characters  -->
<fo:block ... letter-spacing="-2pt"
  font-stretch="ultra-condensed"
  font-size="50pt" font-family="sans-serif"
  font-weight="1000">
  <axf:ruby ruby-font-size="10%"
    ruby-font-stretch="normal" ruby-font-family="serif"
        ruby-font-weight="lighter" ruby-offset="3pt">
        <axf:ruby-base>薊</axf:ruby-base>
        <axf:ruby-text>あざみ</axf:ruby-text>
    </axf:ruby>の如く<axf:ruby ...>
        <axf:ruby-base>棘</axf:ruby-base>
    <axf:ruby-text>とげ</axf:ruby-text>
  </axf:ruby>あれば
</fo:block>
```

The initial value of `axf:ruby-font-size` is 50% of the base character, but when the base character is large, the ruby is then too large, so the smaller size is preferable.

`axf:ruby-font-stretch="normal"` prevents the ruby from being condensed along with the base characters.

Because the base characters are large, the position of the ruby is adjusted using `axf:ruby-offset` property.

Note that when lh or em is specified for `axf:ruby-font-size`, they are based on the font size of the base character.

If you do not change the ruby styles, it will be displayed as shown in [Figure 2.19-5] (p.275).

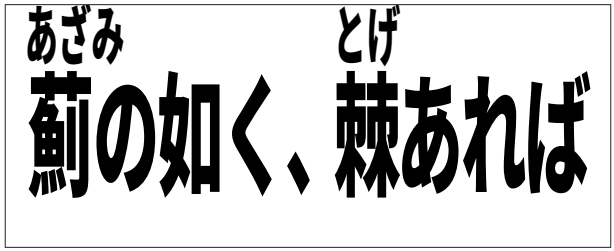


Figure 2.19-5　Default ruby styles

Chapter 3

PDF Extensions

Section 3.1 PDF output

Various PDF features are available.

PDF has gone through multiple versions, but there are also profiles of PDF, such as “PDF/A” and “PDF/X”, tailored for specific purposes. Depending on the output version and profile specifications, some specifications and features may be disabled or not effective. See the manual for details.

3.1.1 PDF Profile

Different PDF versions and profile specifications can be specified output.

Successive PDF versions added, changed or removed features. For example, PDF 1.4 added Tagged PDF. Features that were not removed are still available in later versions.

Version	Release year
1.3	2000
1.4	2001
1.5	2005
1.6	2004
1.7	2008
2.0	2017

Note

Newer versions allow more control over features such as accessibility, security, media support, and so on.

ISO standardizes both PDF versions and the PDF profiles for different applications.

PDF/X: Defined in ISO 15930. Supports reliable interchange of files for printing. Its requirements are "all fonts must be embedded", "support for CMYK and spot colors", "no permission for passwords and print restrictions, etc.", "no permission for links, annotations”, and so on.

PDF/A: Defined in ISO 19005. Supports long-term storage of electronic documents. Its requirements are "all fonts must be embedded", "XMP compliant metadata must be included", "external content reference disallowed", "JavaScript disallowed", "encryption disallowed", and so on.

PDF/UA: Defined in ISO 14289-1. Based on the ISO 32000-1 (PDF 1.7) specification, its purpose is to improve the accessibility of PDF. Its requirements are "Contents

are tagged in logical reading order", "Images with meaning, annotations and formulas must include alternative text", "Must be encrypted without prohibiting accessibility", "The language of the document must be specified", and so on.

Option Property	`pdf-version`
Value	`PDF1.3` | `PDF1.4` | `PDF1.5` | `PDF1.6` | `PDF1.7` | `PDF2.0` | `PDF/X-1a:2001` | `PDF/X-3:2002` | `PDF/X-1a:2003` | `PDF/X-2:2003` | `PDF/X-3:2003` | `PDF/X-4:2010` | `PDF/X-4p:2010` | `PDF/A-1a:2005` | `PDF/A-1b:2005` | `PDF1.4/A-2a:2011` | `PDF1.5/A-2a:2011` | `PDF1.6/A-2a:2011` | `PDF1.7/A-2a:2011` | `PDF1.4/A-2b:2011` | `PDF1.5/A-2b:2011` | `PDF1.6/A-2b:2011` | `PDF1.7/A-2b:2011` | `PDF1.4/A-2u:2011` | `PDF1.5/A-2u:2011` | `PDF1.6/A-2u:2011` | `PDF1.7/A-2u:2011` | `PDF1.4/A-3a:2012` | `PDF1.5/A-3a:2012` | `PDF1.6/A-3a:2012` | `PDF1.7/A-3a:2012` | `PDF1.4/A-3b:2012` | `PDF1.5/A-3b:2012` | `PDF1.6/A-3b:2012` | `PDF1.7/A-3b:2012` | `PDF1.4/A-3u:2012` | `PDF1.5/A-3u:2012` | `PDF1.6/A-3u:2012` | `PDF1.7/A-3u:2012` | `PDF1.5/UA-1:2014` | `PDF1.6/UA-1:2014` | `PDF1.7/UA-1:2014`
Initial	`PDF1.5`
Applies to	`<pdf-settings>`

The `pdf-version` property option setting specifies the PDF version and profile to output. A newer PDF version than the one specified in the `pdf-version` property cannot be embedded.

PDF incompatibility warning

AH XSL Formatter avoids outputting PDF that does not conform to the specified PDF version. However, some incompatibilities are unavoidable, such as “Font that cannot be embedded is specified for PDF/X, which requires font embedding.”

Option Property	error-on-pdfx-fault
Value	true | false
Initial	true
Applies to	<pdf-settings>

The error-on-pdfx-fault option setting controls what to do when PDF/X requirements cannot be met. If true, an error is reported and no PDF is generated. If false, a PDF is generated, but it may not conform to the specified PDF version.

3.1.2 Optimize for Web display

You can output a "linearized PDF" with a structure optimized for viewing PDF in network environments such as on the Web.

Linearizing changes the PDF structure to speed up display both of the first page to be opened and of the pages that may be opened from the opened page.

Option Property	linearlized
Value	true | false
Initial	true
Applies to	<pdf-settings>

When the linearized option setting is true, linearized PDF is output. This takes longer to output than a normal PDF because a normal PDF is generated and then optimized. This can be specified in the GUI by selecting "Fast Web View."

3.1.3 Reduced file size

You can reduce the PDF file size by compressing objects.

Option Property	text-and-lineart-compression
Value	true | false
Initial	true
Applies to	<pdf-settings>

Option Property	`object-compression`
Value	`true` | `false`
Initial	`true`
Applies to	`<pdf-settings>`

When the `text-and-line-art-compression` option setting is `true`, text and line art in PDF output is compressed to reduce the file size. This does not affect any embedded PDF.

Also, in PDF 1.5 or later, when `text-and-line-art-compression="true"`, if the `object-compression` option setting is also `true`, objects in the PDF are compressed.

3.1.4 Password

You can set an owner password to control editing of the output PDF and a user password to control viewing.

Option Property	`owner-password`, `user-password`
Value	*<string>*
Initial	*empty*
Applies to	`<pdf-settings>`

The `owner-password` option setting specifies the PDF owner password, and `user-password` specifies the user password. If neither is specified, no password will be set.

Passwords can be up to 32 ASCII characters in PDF 1.3 to 1.7 or up to 127 ASCII characters in PDF 2.0 and later. A password that contains non-ASCII characters or is too long is invalid, and the password will not be set.

Note

Passwords in the option setting file or `<axf:formatter-config>` are in plain text.

3.1.5 Encryption

When outputting an encrypted PDF, you can specify the encryption method and whether to also encrypt the metadata.

Option Property	encryption-level
Value	40rc4 | 128rc4 | 128aes | 256aes
Initial	128rc4
Applies to	<pdf-settings>

The encryption-level option setting controls key length in encrypted PDF output. 40rc4 is valid for PDF 1.3 to 1.7, 128rc4 is valid for PDF 1.4 to 1.7, 128aes is valid for PDF 1.6 to 1.7, and 256aes is valid for PDF 1.7 and later versions.

Option Property	encrypt-metadata
Value	true | false
Initial	true
Applies to	<pdf-settings>

When the encrypt-metadata option setting is true, the metadata in encrypted PDF is also encrypted. The PDF version must be 1.5 or later.

PDF 1.7 with 256 AES encryption is supported by Adobe Systems™ products as an extension. PDF 2.0 and later support it as standard.

3.1.6 Copying

You can restrict copying of content from the output PDF on a PDF viewer, etc.

Option Property	no-content-copying
Value	true | false
Initial	false
Applies to	<pdf-settings>

When the no-content-copying option setting is true, copying the text and graphics in the output PDF is not allowed. To enable this restriction, the owner-password must also be specified.

3.1.7 PDF layer

You can use layers in the PDF to change how the content is displayed.

In a PDF viewer that supports it, you can switch layers and change the displayed contents based on your settings. Many of the configurable features are available in PDF 1.5 or later.

Property	`axf:layer-settings`
Value	`none` | *<layer-setting>#*
Initial	`none`
Applies to	root element

Value Name	Value Composition
<layer-setting>	*<layer-name>*`[ on | off]] ?[ intent[ view | design] || print[ on | off] || export[ on | off] || lang` *<string>* `preferred]? || zoom` *<number>* *<number>*`]? || locked]] ?`

<layer-settings>	Definition
<layer-name>[on | off]	*<layer-name>* is a *<string>* that specifies the layer name. This name is used in axf:layer as an identifier. on and off supply the default for any of view, print or export that are not specified. If not specified, the layer is regarded as on.
intent[view | design]	If view is specified, it is possible to show or hide the layer in the PDF viewer, which is not possible with design. If omitted, it is regarded as view.
print[on | off]	on specifies that the layer will be printed.
export[on | off]	on specifies that the contents of the layer can be exported (for example, to a raster image format).
lang *<string>* preferred]?	The language of the layer. The layer is displayed when the language specified by *<string>* matches the language of the PDF viewer. When preferred is specified, the layer is displayed even when there is a partial match. The PDF viewer determines whether languages match.
zoom *<number>* *<number>*]?	Minimum and maximum magnifications at which the layer is on. When only one value is specified, it is used as the minimum value.
locked	The layer is locked and editing is disabled. Valid for PDF 1.6 or later.

The axf:layer-settings property on <fo:root> defines the layer names and layer settings. Multiple layers can be defined separated by commas.

Property	axf:layer
Value	none | *<layer-name>*
Initial	none
Applies to	formatting object that generates area

axf:layer specifies the layer in which to place generated areas. *<layer-name>* should match a name in axf:layer-settings.

```
<!-- Display contents are switched for each layer in the
 same PDF -->
<fo:root ...
  axf:layer-settings="
    estimate on intent view view on print on export on l
ang 'ja-JP',
    application off intent view view on print on export
on lang 'ja-JP'
    ">
  ...
  <fo:page-sequence ...>
    <fo:flow ...>
    <fo:block-container absolute-position="absolute"
        axf:layer="estimate"
        top="0.2cm" left="0.5cm">
        ...
      </fo:block-container>
    </fo:flow>
  </fo:page-sequence>
</fo:root>
```

The layer names in `axf:layer-settings` are also used in the output PDF. Also, objects in different layers use their formatted position. Therefore, use absolutely positioned block containers to generate objects that overlap in the same position.

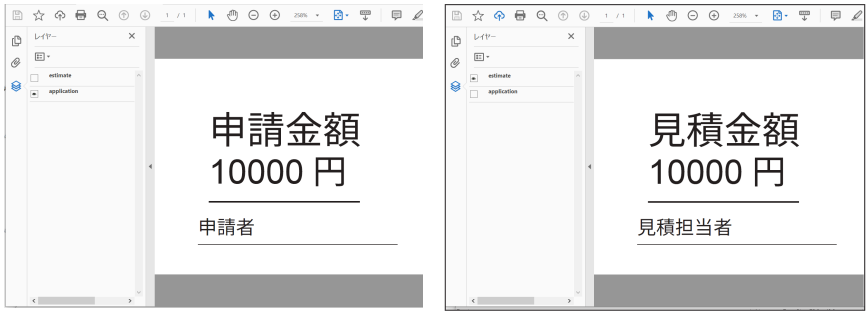


Figure 3.1-1 Display contents are switched for each layer in the same PDF

3.1.8 Pages in reverse order

You can output pages in reverse order.

Option Property	`reverse-page`
Value	`true` | `false`
Initial	`false`
Applies to	`<pdf-settings>`

When the `reverse-page` option setting is `true`, all pages are output in reverse order in the PDF.

After formatting each page, reverse the page order on the PDF.

Note

`axf:reverse-page="true"` in the FO reverses pages in the current page sequence only, while `reverse-page="true"` reverses all pages in the output PDF.

3.1.9 Real values in the PDF

A PDF contains various numerical values, such as information about the content and values used in JavaScript calculations. A too-large number can affect the operation of some PDF viewers.

Option Property	real-value-limit
Value	0 | 1 | 2 | 3
Initial	2
Applies to	<pdf-settings>

Option Property	real-value-limit-modify
Value	true | false
Initial	false
Applies to	<pdf-settings>

The real-value-limit option setting specifies whether to check real values in the PDF and control them so that they do not exceed their limit. Valid with PDF 1.5 or later.

real-value-limit	Definition
0	Do not check.
1	Check that it is within ±32767 (approximately).
2	Check that it is within ±2.14748× 10 9 (approximately).
3	Check that it is within ±9.22337× 10 18 (approximately).

When the output PDF version is 1.4 or less, real-value-limit is considered to be 1 even if 2 or 3 is specified.

If you also want to correct any value that exceeds the limit, specify true for real-value-limit-modify option setting.

This is does not guarantee correct operation of the PDF viewer.

Section 3.2 Document information

You can embed title and author information as metadata in the PDF, and specify to some extent how the PDF will open in the PDF viewer.

In Adobe Acrobat Reader, for example, you can view document information from [**Properties**] [Figure 3.2-1] (p.288).

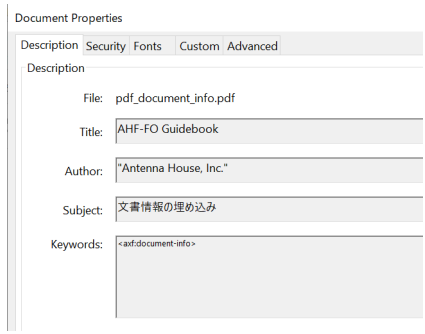


Figure 3.2-1 Document information viewed as “Properties” in Adobe Acrobat Reader™

Element	`<axf:document-info>`
Property	`name`, `value`
Parent	fo:declarations, fo:page-sequence

`<axf:document-info>` is allowed as a child of `<fo:declarations>`.

`name` is the name of one of the document information items described in the following sections. `value` is the value to use. `value` can contain only text.

When outputting multiple volumes, the `<axf:document-info>` in `<fo:declarations>` apply to all volumes. Additionally, one or more `<axf:document-info>` can also be the first child of `<fo:page-sequence>`.

The document information in an `<fo:page-sequence>` overwrites the corresponding `<axf:document-info>` in `<fo:declarations>` for the current volume.

```
<!-- Document information in separate volumes -->
<fo:declarations>
<!-- Document information that applies to all documents  -->
  <axf:document-info name="author" value="Antenna House, Inc" />
     <!-- Separate volume control, here the document information
described in fo:declarations is applied to all files -->
  <axf:output-volume-info document-info-include="all" ... />
```

```
</fo:declarations>
...
<fo:page-sequence ... axf:output-volume-break="true" >
<!-- Applies only to this separate volume  -->
  <axf:document-info name="document-title" value="Volume 1">
  ...
</fo:page-sequence>
<fo:page-sequence ... axf:output-volume-break="true">
<!-- Applies only to this separate volume -->
  <axf:document-info name="document-title" value="Volume 2">
  ...
</fo:page-sequence>
```

3.2.1 Basic document information

Describes the basic document information that can be confirmed in the "Properties" of the PDF viewer.

name	value
document-title	*<Title>*
subject	*<Subtitle>*
keywords	*<Keywords (comma separated)>*
author	*<Author>*

```
<!-- Embedding basic document information -->
<fo:declarations>
  <axf:document-info name="document-title" value="AHF-FO Guidebook
" />
  <axf:document-info name="subject" value="Embedding basic documen
t information" />
  <axf:document-info name="keywords" value="&lt;axf:document-info&g
t;" />
  <axf:document-info name="author" value="Antenna House, Inc." />
...
</fo:declarations>
```

3.2.2 Author and writer of the document

Describes more detailed document information.

name	value
author-title	*<Title of or some keywords about the author>*
description-writer	*<Author of the document description>*

Some PDF viewers do not support displaying detailed document information.

3.2.3 Creation and modification dates

You can explicitly set the creation and modification dates recorded in the PDF.

name	value
createdate	*<Creation date>*
modifydate	*<Modification date>*

If not specified, the current time will be used. in the format is ISO 8601 (JIS X 0301) format, and the following patterns are supported:

Western calendar: `YYYY-MM-DD(Thh(:mm(:ss)?)?)?(Z|±hh(:mm)?)?`

Japanese calendar: `NYY.MM.DD(Thh(:mm(:ss)?)?)?`

Letters have the following definitions:

Letter	Definition
Y	Year. A number from 0 to 9.
N	The Japanese era name. One of 昭,平,令, or S,H,R.
M	Month. A number from 0 to 9.
D	Day. A number from 0 to 9.
T	Delimiter between day and time.
h	Hour of time. A number from 0 to 9.
m	Minute of time. A number from 0 to 9.
s	Second of time. A number from 0 to 9.
-	Delimiter between the parts of the date. Optional.

Letter	Definition
.	Delimiter between the parts of the date. Optional.
:	Delimiter between the parts of the time. Optional.
Z	Time zone. Indicates UTC.
±hh(:mm)	Time zone. Time difference from UTC. ±means either + or –.

If the date and time of creation and modification are both “12:34 on January 2, 2021 AD in Japan time”, this can be specified as follows:

```
<!-- Specify the date and time of PDF creation and modification -->
<axf:document-info name="createdate" value="20210102T1234+09" />
<axf:document-info name="modifydate" value="20210102T1234+09" />
```

2021-01-02T12:34:00+09:00 and 令 03.01.02T12:34 are equally valid.

If the time zone is omitted from a western calendar date, it is considered to be the time zone (local time) of the system on which AH XSL Formatter is installed. The time zone in the Japanese calendar notation is always regarded as Japan time (+09:00). Even if you specify a date that does not exist, for example *Heisei 40*, it will be corrected appropriately. (Heisei is a Japanese era name, and since the era ended in Heisei 30 (2018AD), Heisei 31 and after do not exist.)

When modifydate is older than createdate, it will be aligned to createdate. If the date or the format is incorrect, the output will be corrected when the build finishes. Years earlier than 1970 are invalid in both the Western and Japanese calendars, so even if you specify “明” for Meiji era as N or “1000” for YYYY, it will be ignored and the output PDF will record the current date and time.

3.2.4 Copyright information

You can include copyright information in PDF.

Document information about copyright notices is supported in PDF 1.4 or later.

name	value
copyright-status	Copyright status. Specify either Unknown, Copyrighted or PublicDomain.
copyright-notice	Copyright information text.
copyright-info-url	Copyright information URL

The URL specified for copyright-info-url will not be accessed during processing.

3.2.5 Document information in an XMP file

You can include PDF document information from an external file in XMP format.

PDF 1.4 or later, PDF document information includes an XML specification called Extensible Metadata Platform (XMP).

name	value
xmp	*<URL of XMP file>*

```
<!-- Specify an XMP file that contains the metadata to apply to th
e PDF -->
<fo:declarations>
  <axf:document-info name="xmp" value="common.xmp"/>
</fo:declarations>
```

When outputting PDF, AH XSL Formatter gives priority to other `<axf:document-info>` and specifies AH XSL Formatter in `<pdf:Producer>` and `<xmp:CreatorTool>`.

Creating documents can be more efficient if common document information is maintained as an XMP file.

XMP padding size

XMP padding: Padding added to XMP makes it is possible to edit and extend the XMP even when it is embedded.

Option Property	`xmp-padding`
Value	*<integer>*
Initial	`2000`
Applies to	`<pdf-settings>`

The xmp-padding option setting specifies additional padding to be added to an embedded XMP to assist later in-place editing by other applications. If it is 0 or less, no padding is added. Padding is never added to a read-only XMP that includes . This applies to both an external XMP file and the XMP metadata that is automatically generated when outputting PDF.

The `xmp-padding` option setting specifies additional padding to be added to an embedded XMP to assist later in-place editing by other applications. If it is 0 or less, no padding is added. Padding is never added to a read-only XMP that includes `xpacket end="r"`.

This applies to both an external XMP file and the XMP metadata that is automatically generated when outputting PDF.

3.2.6 Display when PDF opened in viewer

`<axf:document-info>` can specify document information that controls how the PDF is displayed when it is opened in a PDF viewer.

This affects how the PDF is opened, and does not affect subsequent user operations.

Whether it works depends on the PDF viewer.

name	value
`pagemode`	Page mode control. Whether to display the outline or thumbnails, and so on.
`pagelayout`	Page layout control. Whether to display as a two-page spread, two-column display, and so on.
`hidetoolbar`	Toolbar display control. Hide toolbar when `true` is specified. Display with `false`.
`hidemenubar`	Menu bar display control. Hide menu bar when `true` is specified. Display with `false`.
`hidewindowui`	Whether to hide some user interface elements such as scrollbars and navigation controls. Hidden when `true` is specified. Displayed with `false`.
`fitwindow`	Whether the document window is resized to fit the page. Resizes when `true` is specified. Does not resize with `false`.
`centerwindow`	Whether the document window is centered on the screen. Placed in the center when `true` is specified. If `false`, there is no particular control.
`displaydoctitle`	Whether to display the document title as the window title. Displayed when `true` is specified. Hidden with `false`.
`openaction`	A destination to display or an action to perform when the document is opened. A destination has the same *<number-with-fragment>* format as for `internal-destination`. An action is either a named action or JavaScript.

name="openaction"	Definition
value="#Named=Print"	Show print dialog when the PDF is opened.
value=*<number-with-fragment>*	#123 or #page=123 displays the specified page when the PDF is opened.
value="#Named=LastPage"	Show the last page when the PDF is opened.
value="#JavaScript=*<Any JavaScript program>*"	Executes the specified JavaScript when the PDF is opened. Not possible with PDF/A. The JavaScript is ignored when the allow-javascript option setting is false.

3.2.7 Custom properties

You can specify additional PDF properties.

Element	<axf:custom-property>
Property	name, value
Parent	fo:declarations

Items that are not supported values of the name property of the <axf:document-info> element can be embedded with <axf:custom-property>.

Specify the name and value properties in the same way as for <axf:document-info>.

name is case sensitive, and you cannot specify Title, Author, Subject, Keywords, Creator, Producer, CreationDate, ModDate or Trapped.

Section 3.3 PDF Annotations

You can generate PDF annotations using AH XSL Formatter.

PDF annotation is a feature of the PDF specification that allows editable text, and so on, to be associated with a location on a page. In addition to those that can be specified with `axf:annotation-*`, multimedia and links are also PDF annotations. Also, PDF annotations can be used in combination with other annotations.

Different PDF versions support different features. Please see 32000-1:2008 and ISO 32000-2:2017 for details.

3.3.1 Annotation type

You must specify the type of PDF annotation because each type supports different styles and has different features.

Property	`axf:annotation-type`
Value	*<string>*
Initial	empty string
Applies to	all block-level and inline-level formatting objects

`axf:annotation-type` specifies the annotation type, shown below.

`axf:annotation-type`	Definition
`Text`	Text annotation. This is a “text comment” in Acrobat™.
`FreeText`	Free text annotation. This is a “text box” or “text callout” in Acrobat™.
`Stamp`	Stamp annotation.
`FileAttachment`	File attachment annotation.

Note

Pop-up annotation: PDF annotations can be broadly categorized into markup annotations and other types. The `axf:annotation-*` properties In AH XSL Formatter specify several types of markup annotations. Pop-up annotation is used in PDF as children of markup annotations to display the text of the content of markup annotations. Depending on the type of markup annotation, AH XSL Formatter also displays a pop-up annotation when the annotation is specified.

Free text annotation: Text annotations in the PDF appear as sticky notes in Acrobat™. Free text annotations, on the other hand, are annotations that appear to be written directly on the PDF [Figure 3.3-1] (p.296).

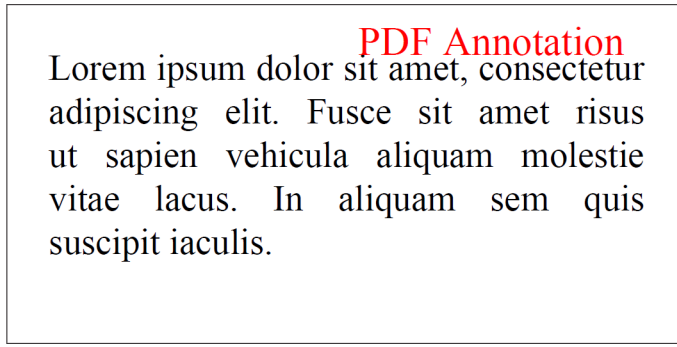


Figure 3.3-1 Free text annotation ("PDF Annotation" in the upper right)

3.3.2 Title and author

You can specify the title and author of a PDF annotation.

Property	`axf:annotation-title`
Value	*<string>*
Initial	empty string
Applies to	all block-level and inline-level formatting objects

Property	`axf:annotation-author`
Value	*<string>*
Initial	empty string
Applies to	all block-level and inline-level formatting objects

`axf:annotation-title` specifies the title, and `axf:annotation-author` specifies the author. `axf:annotation-title` is supported with PDF 1.5 and later.

3.3.3 Creation and modification dates

You can specify the creation date and modification date of a PDF annotation.

PDF annotations store their creation date and modification date information as `/CreationDate` and `/M` values in the annotation dictionary.

These dates can be specified in AH XSL Formatter.

Property	axf:annotation-createdate, axf:annotation-modifydate
Value	<*string*>
Initial	empty string
Applies to	all block-level and inline-level formatting objects

axf:annotation-createdate specifies the annotation creation date, and axf:annotation-modifydate specifies the modification date. The date format is same as for dates in <axf:document-info>.

When axf:annotation-modifydate is older than the axf:annotation-createdate, the axf:annotation-createdate value date is used as the modification date.

If these properties are omitted, the value specified for name="modifydate" in <axf:document-info> will be applied.

3.3.4 File attachment annotations

You can attach a file to a PDF.

In PDF, an annotation on an area on the page can refer to an attached file. Specify axf:annotation-type="FileAttachment" on the object to which the file attachment annotation is added.

Property	axf:annotation-file-attachment
Value	<*uri-specification*>
Initial	empty string
Applies to	all block-level and inline-level formatting objects

axf:annotation-file-attachment specifies the URI of the file to attach.

File attachment annotations are displayed as icons in the PDF.

3.3.5 Background color

You can specify a background color for PDF annotations that have a background.

Property	axf:annotation-color
Value	<*color*> | none
Initial	none
Applies to	all block-level and inline-level formatting objects

axf:annotation-color specify the color used for the background of an annotation.

Depending on the PDF viewer, the color is also reflected in the icon color of PDF annotations that have an icon.

3.3.6 Border style

You can specify some aspects of the borders of free text annotations.

Property	`axf:annotation-border-color`
Value	*<color>*
Initial	axf:annotation-text-color
Applies to	all block-level and inline-level formatting objects

Property	`axf:annotation-border-style`
Value	`solid` | `dotted` | `dashed` | `dot-dash` | `dot-dot-dash`
Initial	`solid`
Applies to	all block-level and inline-level formatting objects

Property	`axf:annotation-border-width`
Value	*<border-width>*
Initial	`0pt`
Applies to	all block-level and inline-level formatting objects

There are no shorthands for `axf:annotation-border-color`, `axf:annotation-border-style` and `axf:annotation-border-width`. There are restrictions on the types of borders that `axf:annotation-border-style` can specify. `double` and `wave` are invalid and will fall back to solid.

3.3.7 Icon

You can specify a unique icon for a PDF annotation.

Property	`axf:annotation-icon-name`
Value	*<string>*
Initial	empty string
Applies to	all block-level and inline-level formatting objects

`axf:annotation-icon-name`specifies the icon to display. Any name can be used, but it depends on the PDF viewer whether the icon can be displayed. There is a possibility that icons that are displayed in Acrobat Reader are not displayed in Firefox. Some annotation types have their own set of standard icon names. See the manual for details.

axf:annotation-type	axf:annotation-icon-name	空のとき
Text	Comment | Help | Insert | Key | NewParagraph | Note | Paragraph	Note
Stamp	Approved | AsIs | Confidential | Departmental | Draft | Experimental | Expired | Final | ForComment | ForPublicRelease | NotApproved | NotForPublicRelease | Sold | TopSecret	Draft
FileAttachment	Graph | PaperClip | PushPin | Tag	PushPin

For example, in text annotations, clicking the icon displays the content and title specified by the `axf:annotation-*` properties.

3.3.8 Styling free text annotations

You can specify styles for free text annotations.

Property	axf:annotation-text-color
Value	<*color*>
Initial	depends on user agent
Applies to	all block-level and inline-level formatting objects

Property	axf:annotation-font-family
Value	<*string*>
Initial	depends on user agent
Applies to	all block-level and inline-level formatting objects

Property	`axf:annotation-font-size`
Value	*<absolute-size>* | *<relative-size>* | *<length>* | *<percentage>*
Initial	depends on user agent
Applies to	all block-level and inline-level formatting objects
Percentages	refer to the font size

Property	`axf:annotation-font-weight`
Value	`normal` | `bold`
Initial	`normal`
Applies to	all block-level and inline-level formatting objects

Property	`axf:annotation-font-style`
Value	`normal` | `italic`
Initial	`normal`
Applies to	all block-level and inline-level formatting objects

Property	`axf:annotation-text-align`
Value	`left` | `center` | `right`
Initial	`left`
Applies to	all block-level and inline-level formatting objects

The PDF viewer sets the default style.

3.3.9 Pop-up and stamp annotation size

You can specify the size of pop-up and stamp annotations.

Property	`axf:annotation-width`, `axf:annotation-height`
Value	*<length>* | `auto`
Initial	`auto`
Applies to	all block-level and inline-level formatting objects

`axf:annotation-width` and `axf:annotation-height` specify the width and height of pop-up and stamp annotations. PDF annotations are horizontal, so `axf:annotation-width` is always horizontal and `axf:annotation-height` is vertical. These properties do not apply to other annotation types.

Note

It is possible to rotate and place the horizontal annotations.

3.3.10 Position

You can position an annotation relative to the object that it annotates.

The default position of pop-up, icon and stamp annotations is the start position of the area. If it is a block, it will be the block start position.

Property	`axf:annotation-position-horizontal`, `axf:annotation-position-vertical`
Value	*<length>*
Initial	`0pt`
Applies to	all block-level and inline-level formatting objects

The `axf:annotation-position-*` property changes the displayed position of the PDF annotation. `axf:annotation-position-horizontal` specifies the horizontal offset and `axf:annotation-position-vertical` property, the vertical offset.

For example, if you want to display a pop-up on the right edge of a paragraph block, specify the length of the line in `axf:annotation-position-horizontal`.

Note

When the size of the area is unknown and, for example, you want to display the annotation at the bottom of a paragraph, this might be difficult to achieve with the `axf:annotation-position-vertical` property.

When a block with `axf:annotation-position-vertical="100vh"` spans across multiple pages, the PDF annotation is placed 100vh from the start of the block on the page.

If you just want to change the display position, create an empty area with `keep-with-previous` after the end of the area you want to annotate, and add the annotation to that area.

3.3.11 Operations on PDF annotations

You can specify permissions for the PDF annotation in the output PDF.

Property	`axf:annotation-flags`
Value	*<string>*
Initial	empty string
Applies to	all block-level and inline-level formatting objects

Users can rewrite the contents of PDF annotations with PDF viewer or editing software. `axf:annotation-flags` controls which of these operations are allowed.

In addition, it controls whether an annotation can be printed.

`axf:annotation-flags`	**Definition**
`Invisible`	Hide the annotation if it is an unknown annotation type.
`Hidden`	Do not display or print.
`NoView`	Do not display, but include in print output (depending on other flags).
`ToggleNoView`	Invert NoView (display if hidden, hide if visible) when an event fires (mouse click, etc.).
`Print`	Include in print output.
`NoZoom`	Limit scaling operations.
`NoRotate`	Limit rotation operations.
`ReadOnly`	Do not respond to operations such as mouse clicks. Does not affect display or printing.
`Locked`	The PDF annotation itself becomes inoperable. Changes to the contents, such as form field values, are permitted.
`LockedContents`	Do not allow the contents to be modified.

3.3.12 Text content

You can specify the text of an annotation.

Property	axf:annotation-contents
Value	<*string*>
Initial	empty string
Applies to	all block-level and inline-level formatting objects

axf:annotation-contents specifies the displayed contents of text, free text, stamp and file attachment annotations. It can also provide alternative text for links generated by <fo:basic-link>.

For a pop-up annotation, the contents of axf:annotation-title, axf:annotation-author and axf:annotation-contents are displayed in combination.

3.3.13 Annotations in embedded PDF

Option Property	import-annotation-types
Value	<*string*>
Initial	empty string
Applies to	<pdf-settings>

Annotation types that can be specified in import-annotation-types `import-annotation-types`

`All`, `Text`, `Link`, `FreeText`, `Line`, `Square`, `Circle`, `Polygon`, `PolyLine`, `Highlight`, `Underline`, `Squiggly`, `StrikeOut`, `Stamp`, `Caret`, `Ink`, `Popup`, `FileAttachment`, `Sound`, `Movie`, `Screen`, `3D`, `RichMedia`, `Other`

The import-annotation-types option setting specifies the types of annotation to be retained in an embedded PDF. Multiple types can be specified by separating them with spaces. Specify All when embedding all annotations, and specify Other when embedding types of annotations other than those listed above.

Note

The types of PDF annotations that can be retained when you embed a PDF are different from the ones that you can create when you generate a PDF with AH XSL Formatter.

3.3.14 Use case: Attach source data to a table

Add a file attachment annotation to the table and specify the size and position of the annotation and its metadata.

If you do not specify the size of a PDF annotation, its size matches the area of the object to which it is attached. For example, if you annotate an `<fo:table>`, an icon indicating a file attachment annotation will be displayed throughout the table.

The following example adds the source data to a table as a file attachment annotation:

File attachment annotation

Specify the type of annotation, the file and the icon.

```
<!-- Add file attachment annotation to table -->
<fo:table-and-caption ...
  axf:annotation-type="FileAttachment"
  axf:annotation-file-attachment="url(./table.xslx)"
  axf:annotation-icon-name="PushPin" ...>
    <fo:table-caption>
      <fo:block... >Table</fo:block>
    </fo:table-caption>
  <fo:table ...>
    ...
  </fo:table>
</fo:table-and-caption ...>
```

Position and size

Add a 1cm icon at the right-hand side of the first line of the table caption.

```
<!-- Specify the position and size of PDF annotation  -->
<fo:table-and-caption ...
  axf:annotation-width="1cm" axf:annotation-height="1cm"
  axf:annotation-position-horizontal="100vw - 1cm">
  ...
</fo:table-and-caption>
```

Specify the width, height and position of the file attachment annotation. Since the origin of the object is the origin for the annotation, the right-hand edge can be specified as 100 vw. In the above FO, the 1 cm icon width is subtracted so the icon fits within the area of the table.

Section 3.4 Links

Links in PDF are expressed as annotations called link annotations.

3.4.1 Opening external links

You can specify how external links are opened.

Property	axf:action-type
Value	auto | gotor | launch | uri
Initial	auto
Applies to	fo:basic-link

axf:action-type on <fo:basic-link> controls the action to take for an external link. Only the values relevant to opening a link can be specified in axf:action-type on <fo:basic-link>.

axf:action-type	Definition
gotor	Open as PDF. The linked URI is considered a PDF.
launch	Open as file.
uri	Open as URI.

Option Property	use-launch-for-local-file
Value	true | false
Initial	true
Applies to	<pdf-settings>

With axf:action-type="auto", the use-launch-for-local-file option setting specifies the action. When use-launch-for-local-file is true, the action is “Open file”, and when false, the link is considered a “World Wide Web link”.

3.4.2 Link to PDF

You can specify how the output PDF page will be displayed (its destination type) when it is opened.

You can control both zoom and position.

A destination with both position and magnification is written in the PDF as:

```
[page /XYZ left top zoom]
```

where: *page* is the destination page; *left* is the position from the left edge on the screen; *top* is the position from the top edge; and *zoom* is the display magnification as a percentage value. The value null for left, top or zoom specifies that the current value is used.

A destination where the PDF viewer should adjust the display size to fit the page is written in the PDF as:

```
[page /Fit]
```

In addition to /Fit, there are also settings such as /FitH and /FitV that take parameters to adjust the position.

Property	`axf:destination-type`
Value	<*string*>
Initial	empty string
Applies to	block level formatting object

`axf:destination-type` specifies the destination type. The value is not case sensitive. The default, `axf:destination-type="xyz-top"`, is also used when the character string is invalid.

AH XSL Formatter automatically calculates the distance from the left and top edges of the display position. The `xyz*` values generate *null* values in the PDF as required. If the magnification zoom is not specified, it is considered *null*.

`axf:destination-type`	**Definition**	**PDF destination type**
`xyz` <*number*>] ?	<*number*> specifies the magnification. left and top are null.	`[page /XYZ null null zoom]`
`xyz-top` | <*number*>	<*number*> specifies the magnification. left is null.	`[page /XYZ null top zoom]`
<*number*>	<*number*> specifies the magnification. left is null.	`[page /XYZ null top zoom]`
`xyz-left` <*number*>] ?	<*number*> specifies the magnification. top is null.	`[page /XYZ left null zoom]`
`xyz-left-top` <*number*>] ?	<*number*> specifies the magnification.	`[page /XYZ left top zoom]`
`fit`	Display the entire page at a magnification that fits the window both vertically and horizontally. If the magnification factors	`[page /Fit]`

axf:destination-type	Definition	PDF destination type
	are different, use the smaller magnification and display the page in the center of the window.	
fitv	Display the page height at a magnification that fits the window. AH XSL Formatter calculates the horizontal position.	[*page* /FitV *left*]
fith	Only valid for internal links. Display the page width at a magnification that fits the window. AH XSL Formatter calculates the vertical position.	[*page* /FitH *top*]
fitr	Only valid for internal links. Display in the area scaled to fit the window.	[*page* /FitR *left bottom right top*]
fitb	The entire width of the page border box is displayed to fit the window.	[*page* /FitB]
fitbh	Only valid for internal links. Scale the height of the page border box to fit the window.	[*page* /FitBH *top*]
fitbv	Scale the width of the page border box to fit the window.	[*page* /FitBV *left*]

3.4.3 Use case: Form field link

It is possible to create a button that triggers a link.

```
<!--  Link by push button -->
<axf:form-field field-type="button" ...
  action-type="uri"
  external-destination="https://www.antennahouse.com" />
```

Section 3.5 Multi-volume output

You can output one document as multiple PDF files. You can control volume location and the bookmarks and document information in each volume.

Items to consider when outputting one document as multiple volumes include: document information common to all volumes; individual document information; number of pages; file name of each volume; cross-references to other volumes; and so on.

Also, PDF can contain bookmarks that are independent from the printed table of contents. The `axf:bookmark-*` properties are used to create bookmarks in multi-volume output instead of using `<fo:bookmark>`.

Multi-volume output works with PDF output. It does not work with printing or stream output.

Element	`<axf:output-volume-info>`
Property	`axf:initial-volume-number`, `axf:document-info-include`, `format`
Parent	fo:declarations

`<axf:output-volume-info>` provides overall control when page sequences are output as separate volumes.

3.5.1 Separation into volumes

You can specify whether a page sequence starts a new volume.

Property	`axf:output-volume-break`
Value	`true` | `false`
Initial	`false`
Applies to	fo:page-sequence

When `axf:output-volume-break` is true, the `<fo:page-sequence>` starts a new volume. The first `<fo:page-sequence>` always starts a volume.

3.5.2 File name

You can specify the file name of individual PDF files.

A new volume is started for the first `<fo:page-sequence>` and for any `<fo:page-sequence>` with `axf:output-volume-break="true"`.

Property	axf:output-volume-filename
Value	<*string*>
Initial	empty string
Applies to	fo:page-sequence

`axf:output-volume-filename` specifies the file name of the volume.

3.5.3 Document information

You can specify how to handle document information such as the author name and how the file opens in a PDF viewer.

Property	axf:document-info-include
Value	`first` | `all`
Initial	`first`
Applies to	axf:output-volume-info

`axf:document-info-include` specifies how to handle the PDF document information in `<axf:document-info>` when outputting as separate volumes.

`first` adds document information to only the first volume, whereas `all` adds the information to all volumes.

3.5.4 First volume number

You can specify the volume number of the first volume.

Property	axf:initial-volume-number
Value	<*number*>
Initial	`1`
Applies to	axf:output-volume-info

`axf:initial-volume-number` specifies the volume number of the first volume. Unless `axf:output-volume-filename` is specified, the volume number is added as a suffix to the file name of the PDF output. `format` on `<axf:output-volume-info>` specifies the suffix format. `format` values are the same as for the XSL 1.1 format property.

3.5.5 Bookmarks

You can control the bookmarks when the FO document is output as multiple PDF files.

Property	axf:outline-title
Value	*<string>*
Initial	empty string
Applies to	block-level formatting objects

Property	axf:outline-level
Value	*<number>*
Initial	0
Applies to	block-level formatting objects

Property	axf:outline-group
Value	*<string>*
Initial	empty string
Applies to	block-level formatting objects

`axf:outline-title` specifies the bookmark title. If omitted, the text of the object will be the title. `axf:outline-color` and other properties control text style.

With `<fo:bookmark>`, the elements nested under `<fo:bookmark-tree>` determine the bookmark hierarchy. With multi-volume output, on the other hand, the structure of the elements with `axf:outline-*` determines the hierarchy. `axf:outline-level` is an integer greater than or equal to 0 and specifies the level of a bookmark. A bookmark is not created with the initial value of 0. The highest level is 1, and a larger number indicates a deeper level.

Bookmarks in volumes

Property	axf:bookmark-include
Value	first | all | separate | separate-group
Initial	separate
Applies to	axf:output-volume-info

axf:bookmark-include	Definition
first	Add bookmark information of all separate volumes to the first volume.
all	Add bookmark information of all separate volumes to all separate volumes.
separate	Add each bookmark to each separate volume.
separate-group	In addition to separate, grouped bookmarks are also added to each separate volume.

With axf:bookmark-include="separate"", a bookmark belongs to the volume in which axf:outline-level="1"appears. When a bookmark spans across multiple volumes, it is included in the first of its volumes.

Bookmarks grouped by the axf:outline-group property are included together in the last volume.

In the next FO, the "Heading bunsatsu" bookmark will be included in the bunsatsu2.pdf file.

```
<!-- Add bookmark of "Heading bunsatsu" to bunsatsu2.pdf  -->
<fo:page-sequence axf:output-volume-break="true"
  axf:output-volume-filename="bunsatsu2.pdf">
  <fo:flow ... >
    ...
    <fo:block axf:outline-level="1"
      axf:autline-title="Heading bunsastu">
      Heading bunsastu
    </fo:block>...
  </fo:flow ... >
</fo:page-sequence>
```

3.5.6 External and internal links in bookmarks in separate volumes

You can generate a bookmark that is a link.

Property	axf:outline-external-destination
Value	<*uri-specification*>
Initial	empty string
Applies to	block-level formatting objects

Property	axf:outline-internal-destination
Value	*<ref-id>* | *<number-with-fragment>*
Initial	empty string
Applies to	block-level formatting objects

For external links, specify `axf:outline-external-destination` containing a `<uri-specification>`.

For internal links, specify `axf:outline-internal-destination` containing either *<ref-id>* or *<number-with-fragment>*, which comprises a page number, "#", and a fragment identifier.

3.5.7 Bookmark styles

You can style the color and weight of bookmarks.

With a single PDF output file and `<fo:bookmark-title>` containing `<fo:bookmark>`, the bookmark text is styled using the `color` property. Bookmark text output in separate volumes is styled using these extension properties.

Property	axf:outline-color
Value	*<color>*
Initial	transparent
Applies to	block-level formatting objects

Property	axf:outline-font-style
Value	normal | italic
Initial	normal
Applies to	block-level formatting objects

Property	axf:outline-font-weight
Value	normal | bold
Initial	normal
Applies to	block-level formatting objects

`axf:outline-font-style` can only specify whether the text is italic or not, and `axf:outline-font-weight` property can only specify whether it is bold or not.

These properties are valid with PDF 1.4 and later.

3.5.8 Expansion of bookmarks in separate volumes

You can specify whether to expand lower-level bookmarks when the PDF is opened.

Property	`axf:outline-expand`
Value	`true` | `false`
Initial	`true`
Applies to	block-level formatting objects

When `axf:outline-expand` is `true`, the bookmarks underneath the current bookmark are expanded and displayed when opened in a PDF viewer that supports it.

3.5.9 Use case: Table of contents and index for a multiple volumes

Create table of contents when creating PDF in multi-separate volumes.

The PDF is divided into “上巻” (first volume), “下巻” (last volume) and “索引” (index) volumes [Figure 3.5-1] (p.316).

Specify separate volumes

```
<!-- Specify separate volumes -->
<fo:declarations>
  <axf:output-volume-info
    initial-volume-number="1" format="上下索"
    bookmark-include="separate" />
</fo:declarations>
```

When `format` is a character string other than tokens and symbols, each character is used once, so “上”, “下” and “索” are added to the end of the output PDF file name. Since the text is used character-by-character, you cannot add “索引” (index) to the end of a file name, even if “上下索引” is specified.

With `axf:bookmark-include="separate"`, the first volume contains the bookmarks for the first volume, and the last volume contains the bookmarks for the last volume.

If there are only three volumes, you could specify the file names individually with `axf:output-volume-filename`.

Bookmarks

```
<!-- Specify bookmarks in separate volumes -->
<fo:block-container ...  axf:outline-level="1"
  axf:outline-title="OSDC V7.1 マニュアル 上巻">
    <fo:block ...
    >Office Server Document Converter V7.1 マニュアル</fo:block>
    <fo:block ...>— 上巻 —</fo:block>
    <fo:block ...>インターフェイス編</fo:block>
</fo:block-container>
...

<fo:block ... axf:outline-level="2"
  letter-spacing="0.4em">上巻目次</fo:block>
...
```

When outputting PDF in multiple volumes, the `axf:outline-*` properties specify the bookmarks.

`axf:outline-title` is not specified alongside `axf:outline-level="2"` in the example above, so "上巻目次" (first volume table of contents) is used as the bookmark title.

If you insert spaces between letters in the heading for better formatted output, the spaces will also be inserted in the bookmark. This can be prevented by specifying the `axf:outline-title` property independently, or, alternatively, `letter-spacing` can be used to add space between letters for the purpose of display.

Volume prefix

```
<!-- 上巻 -->
<fo:page-sequence master-reference="PageMaster">
  <fo:folio-prefix>上</fo:folio-prefix>
    <fo:static-content flow-name="xsl-region-after">
        <fo:block ... text-align="center">
      <fo:inline>- </fo:inline>
      <fo:page-number axf:suppress-folio-prefix="true" />
      <fo:inline> -</fo:inline>
    </fo:block>
  </fo:static-content>
...
</fo:page-sequence>
...
  <!-- 下巻 -->
<fo:page-sequence master-reference="PageMaster">
  ...
  <fo:folio-prefix>下</fo:folio-prefix>
```

```
  ...
</fo:page-sequence>
```

`<fo:folio-prefix>` specifies the page number prefix.

When outputting as multiple volumes, it is useful to add a different prefix or suffix to the page numbers in each volume. However, when reading the first volume, it is clear that the page is in the first volume, so `axf:suppress-folio-prefix` is used to suppress the prefix in the page number displayed in the region-after.

Table of contents

```
<!-- Table of Contents layout -->
<fo:block ...>上巻目次</fo:block>
<fo:block ...><axf:tab tab-align="')'"
  />1) コマンドラインインターフェイス
    <fo:leader leader-pattern="dots" leader-length.optimum="100%"/>
    <fo:page-number-citation ref-id="commandIF"
      axf:suppress-folio-prefix="true" />
</fo:block>
```

Since the heading is "上巻目次" (first volume ToC), it is redundant to add "上" to the page numbers, so `axf:suppress-folio-prefix="true"` is used to suppress the prefix.

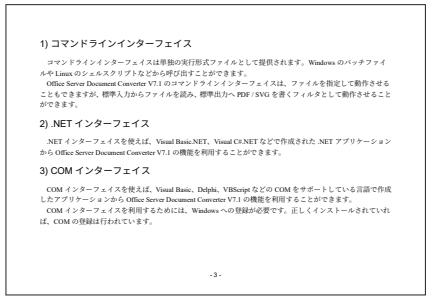
`<axf:tab tab-align="')'"/>` and `<fo:leader>` are for aligning page numbers with different numbers of digits.

Index

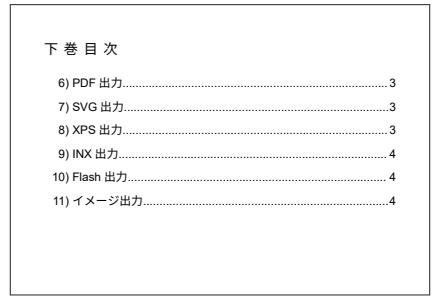
```
<!-- Index layout -->
<fo:block ... >索引</fo:block>
<fo:block-container column-count="2" column-gap="10mm">
  <fo:block text-align-last="justify" space-before="0.5em"
  >.NET インターフェイス<fo:leader leader-pattern="dots" />
        <fo:page-number-citation ref-id="dotnetIF" />
  </fo:block>
  ...
</fo:block-container>
```

Page references in the index include a prefix ("上" and "下") so that it is easy to know whether it is in the first or last volume. `column-count="2"` is specified in `<fo:block-container>`, and the index items are formatted in 2 columns.

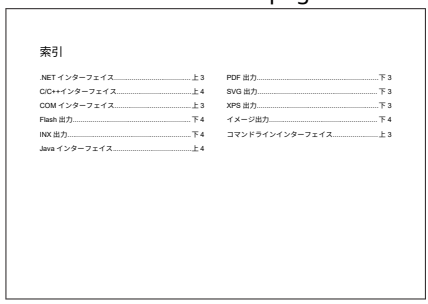
In this example, there are only a few pages, so there is only one line per item, but there are many things to consider, such as consecutive page number references or page number references with a large number of digits in different volumes.



First volume page



ToC of the last volume



Index

Figure 3.5-1　Table of contents and index for multiple volumes

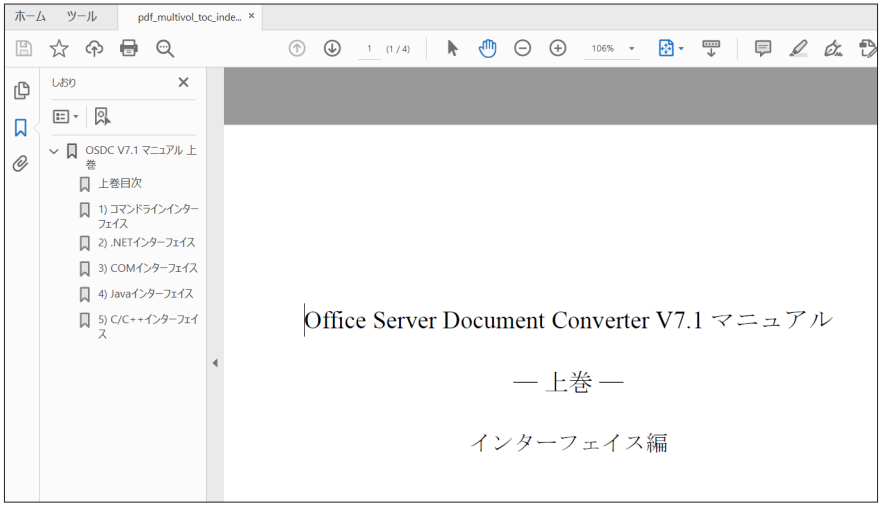


Figure 3.5-2　Bookmarks for one volume

Section 3.6 Embedding multimedia

Media files such as video and audio can be embedded in PDF. In addition, it is possible to embed 3DCG and other formats.

There are two ways to embed multimedia: multimedia annotations and rich media annotations. Multimedia annotations can be included in PDF 1.5 and later, except in PDF/X and PDF/A. However, multimedia annotations are deprecated in PDF 2.0 and later. They cannot be specified in the background-image property.

Rich media annotations are available in PDF 1.7 and later.

3.6.1 Rich media window size

Rich media annotations can display in a separate window. You can specify the size of this window.

The size of the window that opens when `axf:media-treatment="richmedia-windowed"` is specified is determined automatically. Alternatively, the size can be explicitly specified.

Property	`axf:media-window-width`, `axf:media-window-height`
Value	`auto` | *<length>*
Initial	`auto`
Applies to	fo:external-graphic / multimedia objects

`axf:media-window-width` specifies the width, and `axf:media-window-height` specifies the height.

3.6.2 Rich media skin

The PDF viewer provides a 'skin' for the controls for playing rich media annotations. You can control the appearance of this skin.

Property	`axf:media-skin-control`
Value	`none` | `all` | [`play` || `stop` || `forward` || `rewind` || `seek` || `mute` || `volume`]
Initial	`all`
Applies to	fo:external-graphic / multimedia objects

Property	axf:media-skin-auto-hide
Value	true | false
Initial	true
Applies to	fo:external-graphic / multimedia objects

axf:media-skin-control specifies which controls are displayed in the skin.

axf:media-skin-auto-hide controls whether the skin is automatically hidden when rich media annotations are not enabled. It will be hidden automatically with true.

Property	axf:media-skin-color
Value	auto | <*color*>
Initial	auto
Applies to	fo:external-graphic / multimedia objects

axf:media-skin-color controls the skin color. If you specify an invalid value, the default value will be used.

3.6.3 Rich media activation timing

You can control the timing of rich media annotation to be activated or deactivated, such as the video starts playing when the annotation page is displayed.

Property	axf:media-activation
Value	[ExplicitlyActivated | PageOpen | PageVisible]] ? [ExplicitlyDeactivated | PageInvisible | PageClose]] ?
Initial	*empty*
Applies to	fo:external-graphic / multimedia objects

axf:media-activation specifies when to activate or deactivate the rich media annotation. If specifying both, separate them with a space. Each value also has a two-letter abbreviation, as shown in the following table:

axf:media-activation	Timing	Activate/ Deactivate
ExplicitlyActivated, XA	When clicking on the content	Activate
PageOpen, PO	When opening a page that contains content	Activate
PageVisible, PV	When a page containing content is displayed	Activate
ExplicitlyDeactivated, XD	When “Disable Content” is selected from the context menu	Deactivate
PageClose, PC	When closing the page that contains the content	Deactivate
PageInvisible, PI	When the page containing the content is not displayed	Deactivate

The value is not case sensitive. The default, when nothing is specified (`axf:media-activation=""`), is `ExplicitlyActivated ExplicitlyDeactivated`.

Section 3.7 AcroForm

The PDF can contain an AcroForm.

The AcroForm is useful for creating the PDF of a form that can be completed on a PC. The form can be used in various ways, such as adding a digital signature and processing values with JavaScript.

Element	`<axf:form>`
Property	`external-destination`, `axf:field-submit-method`, `axf:field-submit-coordinates`
Parent	page area objects

Element	`<axf:form-field>`
Property	`axf:field-type`
Parent	axf:form, page area objects

An AcroForm consists of form fields that each receives a value, such as text or a checkbox, and a form action that determines what to do with the values entered in the form fields.

If you want to submit or reset the values of form fields created by `<axf:form-field>` —by specifying `submit` or `reset` in `axf:action-type`—create the `<axf:form-field>` as descendants of the `<axf:form>`.

`<axf:form-field>` is an inline element. Use `width` and `height` to specify its size.

Note

The form does not appear in the GUI preview.

3.7.1 Form field

You can create a form field that can be filled in by the user.

Type

Property	`axf:field-type`
Value	`text` | `button` | `checkbox` | `radio` | `listbox` | `combobox` | `signature`
Initial	N/A, a value is required
Applies to	axf:form-field

`axf:field-type` is a required property that specifies the type of form field.

Name

The field name is used, for example, to retrieve the entered value in some way.

Property	`axf:field-name`
Value	*<string>*
Initial	N/A
Applies to	axf:form-field

Property	`axf:field-name-suffix-page-number`
Value	*<string>*
Initial	empty string
Applies to	axf:form-field

`axf:field-name` specifies the name of the form field. When not specified, it is set by AH XSL Formatter.

If `axf:field-name-suffix-page-number` specifies a number format and `axf:field-name` is not an empty string, the current page number, with the format applied, will be appended to the `axf:field-name` value. This is useful when you want to generate a separate form in the `<fo:static-content>` for each page.

Fixed value

Property	axf:field-value
Value	*<string>*
Initial	empty string
Applies to	axf:form-field, axf:form-field-option

axf:field-value specifies the value to use when a check box or radio button is selected.

Font size

Property	axf:field-font-size
Value	font-size | auto | *<length>*
Initial	font-size
Applies to	axf:form-field

axf:field-font-size specifies the font size of the text displayed in the form field. When axf:field-font-size="font-size" is specified, this will be the value specified by the font-size property, and when axf:field-font-size="auto" is specified, it is set by the form field.

```
<!-- font-size="10pt" block container  -->
<fo:block-container font-size="10pt" width="100vw" height="3rlh" >
  <!-- formA のフォントサイズは 10pt -->
  <axf:form-field field-name="formA" field-font-size="font-size" ..
./>
  <!-- formB のフォントサイズは AH XSL Formatter が設定 -->
  <axf:form-field field-name="formB" field-font-size="auto" .../>
</fo:block>
```

Note

axf:field-font-size applies to many field types. Text fields allow more fine-grained control.。

Description

Property	axf:field-description
Value	*<string>*
Initial	empty string
Applies to	axf:form-field

axf:field-description contains a description of the form field. This is displayed as a tool tip in Acrobat [Figure 3.7-1] (p.324).

```
<!-- Add a tooltip description to the input-required fields  -->
<axf:form-field
  ...
  field-required="true"
  field-description="Input required" ... />
```

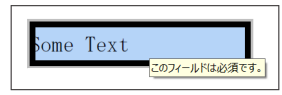


Figure 3.7-1 Tooltip on required fields

3.7.2 Field characteristics

You can make the field read-only or required.

Property	axf:field-flags
Value	*<string>*
Initial	empty string
Applies to	axf:form-field

axf:field-flags specifies various characteristics of the field. Tokens other than Hidden can be specified in individual extensions, but in each case, the setting in axf:field-flags takes precedence. You can specify multiple tokens separated by spaces. It is not case sensitive.

axf:field-flags	Definition
ReadOnly	Read-only field. Same as axf:field-readonly="true".
Required	Required field. Same as axf:field-required="true".
Multiline	A text field may contain multiple lines. Same as axf:field-multiline="true".
Scroll	A text field is scrollable. Same as axf:field-scroll="true".
Password	A text field is intended for entering a password. Same as axf:field-password="true".
Checked	A check box or radio button is checked. Same as axf:field-checked="true".
Multiple	More than one item in the list box may be selected simultaneously. Same as axf:field-multiple="true".
Editable	A combo box includes an editable value. Same as axf:field-editable="true".
Hidden	A hidden field. The size of the field is reserved. Backgrounds and borders are displayed.

Read-only field

Property	axf:field-readonly
Value	true | false
Initial	false
Applies to	axf:form-field

Fields with axf:field-readonly="true" cannot be changed by the user.

Note

This is useful for text fields that store the results calculated by JavaScript from values entered in other form fields.

Required field

Property	`axf:field-required`
Value	`true` | `false`
Initial	`false`
Applies to	axf:form-field

Fields with `axf:field-required="true"` must have a value when the form is submitted. Any visual effects depend on the PDF viewer.

3.7.3 Text field

You can specify the format and style of the text field where the user can enter the text.

A `<axf:form-field>` with `axf:field-type="text"` is a text field.

You can control the layout of the text field.

Alignment

Property	`axf:field-text-align`
Value	`left` | `center` | `right`
Initial	`left`
Applies to	axf:form-field

`axf:field-text-align` specifies the alignment of the entered text.

Text entered by default

Property	`axf:field-default-text`
Value	*<string>*
Initial	empty string
Applies to	axf:form-field

`axf:field-default-text` specifies the default text of the field.

Password field

Property	axf:field-password
Value	false | true
Initial	false
Applies to	axf:form-field

When axf:field-password is true, the text field is a password field. The password field masks the input when filling out the field and prompts the PDF viewer not to save the input in the PDF file.

Property	axf:field-maxlen
Value	<*number*>
Initial	0
Applies to	axf:form-field

axf:field-maxlen specifies the maximum number of characters that can be entered. Since it controls the number of characters that can be entered as a field, it is necessary to adjust the font size with axf:field-font-size, etc., to match text with the apparent edge of the text field.

No limit is set when the specified value is 0 or less or when the length of the axf:field-default-text value is greater than the specified value.

[Figure 3.7-2] (p.328) shows an example of a text field with axf:field-password="true". In the example, the maximum number of characters fits within the area of the text field.

```
<!-- Password field example -->
<axf:form-field axf:field-type="text" axf:field-name="TextPassword"
  axf:field-multiline="false"
  axf:field-default-text=""
  axf:field-password="true" axf:field-required="true"
  axf:field-maxlen="8"

  width="10em" height="1.2em"
  background-color="#ff8" border="1pt red inset" />
```

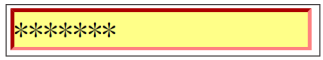


Figure 3.7-2 Password field example

Text field format

You can also use a text field when you want to enter a number or date in the form.

Property	`axf:field-format-category`
Value	`none` | `number` | `percentage` | `date` | `time`
Initial	`none`
Applies to	axf:form-field

Property	`axf:field-format`
Value	`auto` | [[<*string*> | <*number*>][<*string*> | <*number*> | `true` | `false`]] *]
Initial	`auto`
Applies to	axf:form-field

`axf:field-format-category` categorizes the value that can be entered in the text field, and `axf:field-format` specifies the format in more detail.

When `axf:field-format-category` is `number`, `axf:field-format` can contain up to five values, separated by spaces.

Position	Definition	Initial Value
First	The number of digits after the decimal point. When the input value has more digit numbers after the decimal point, the rounded value is displayed.	`2`
Second	How to display separators between digits.	`0`
Third	How to display negative numbers.	`0`
Fourth	Currency symbol, which can be any character string.	`-`
Fifth	Displayed position of the currency symbol string. Displayed before numbers if `true`, displayed after numbers if `false`.	`true`

To control display of separators between digits, specify one of these numbers:

Specification	Display of 1234.56
0	1,234.56
1	1234.56
2	1.234,56
3	1234,56
4	1'234.56

To control display of negative numbers, specify one of these numbers:

Specification	Negative number display
0	No processing.
1	Display in red.
2	Display with parentheses.
3	Display in red with parentheses.

When `axf:field-format-category` is `percentage`, `axf:field-format` can contain up to two values. Similarly to `number`, the first value is the number of digits after the decimal point, and the second is how to display separators between digits.

Since a currency symbol is placed immediately before or after the numerical value, it will be easier to see if you including spaces in the string.

When `axf:field-format-category` is `date` or time, `axf:format` can specify either a format string or an equivalent numerical value.

The following table shows the formats corresponding to numerical `axf:field-format` values when `axf:field-format-category="date"`. The initial value is `0`.

Specification	Format	Display of April 12, 2019 13:07
0	m/d	4/12
1	m/d/yy	4/12/19
2	mm/dd/yy	04/12/19
3	mm/yy	04/19
4	d-mm	12-04
5	d-mmm-yy	12-Apr-19
6	dd-mmm-yy	12-Apr-19

Specification	Format	Display of April 12, 2019 13:07
7	yy-mm-dd	19-04-12
8	mmm-yy	Apr-19
9	mmmm-yy	April-19
10	mmm d, yyyy	Apr 12, 2019
11	mmmm d, yyyy	April 12, 2019
12	m/d/yy h:MM tt	4/12/19 1:07 PM
13	m/d/yy HH:MM	4/12/19 13:07

The following table shows the formats corresponding to numerical `axf:field-format` values when `axf:field-format-category="time"`. The initial value is `0`.

Numerical Value	Character string	Display of 13:07:02
0	HH:MM	13:07
1	h:MM tt	1:07 PM
2	HH:MM:ss	13:07:02
3	h:MM:ss tt	1:06:02 PM

Any format can be specified, but the result is not guaranteed if the format is incorrect. If the formatting is invalid in the output PDF, check the Acrobat version.

Multiline text field

Property	axf:field-multiline
Value	false | true
Initial	false
Applies to	axf:form-field

When `axf:field-type="text"` and `axf:field-multiline="true"` are specified, it is a multiline text field. The number of lines that appear to fit depends on the value of the `axf:field-font-size` property and the height setting.

Property	axf:field-scroll
Value	false | true
Initial	false
Applies to	axf:form-field

If you want to fit multiple lines even though the page does not have enough space, you can specify `axf:field-scroll="true"` to make the text field scrollable.

[Figure 3.7-3] (p.331) shows an example of a multiline text field. `axf:field-maxlen="100"` limits the text to 100 characters. The font size of the field is currently unspecified, and there is a possibility that the input will exceed the area for the text field. Using `axf:field-scroll="true"`, characters will be scrolled when they exceed the area.

```
<!-- Display the scroll bar -->
<axf:form-field axf:field-type="text"
  axf:field-name="TextMultiline"
  axf:field-default-text="Input text here! sample text"
  axf:field-multiline="true" axf:field-scroll="true"
  axf:field-maxlen="100"

  width="10em" height="2.4em"
  background-color="#ff8" border="1pt solid silver" />
```

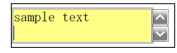


Figure 3.7-3 Scroll bar display

3.7.4 Push button

A field with `axf:field-type="button"` is a push button. It is useful for starting actions and for links.

An extension on the appearance of push buttons. If you want an action to occur when you press the push button, specify a form action.

Property	axf:field-button-layout
Value	auto | icon | caption-below-icon | caption-above-icon | icon-caption | caption-icon | caption-over-icon
Initial	caption
Applies to	axf:form-field

Property	axf:field-button-face
Value	*<string>*
Initial	empty string
Applies to	axf:form-field

Property	axf:field-button-icon
Value	*<uri-specification>*
Initial	empty string
Applies to	axf:form-field

axf:field-button-layout specifies the appearance and position of the button's caption and icon. caption displays the button's text (caption) that is specified in axf:field-button-face. icon displays the button's icon image. Unlike PDF annotations, icons are specified by a *<uri-specification>*.

Property	axf:field-button-face-down, axf:field-button-face-rollover
Value	*<string>*
Initial	empty string
Applies to	axf:form-field

Property	axf:field-button-icon-down, axf:field-button-icon-rollover
Value	*<uri-specification>*
Initial	empty string
Applies to	axf:form-field

You can specify how the button display changes when the push button is pressed (clicked with the mouse). axf:field-button-face-down specifies caption changes and axf:field-button-icon-down specifies icon changes. When the pressed state ends, the original display contents are restored.

The *RollOver* state is when the mouse pointer is over the field. You can change the button display when the mouse pointer is over the push button. axf:field-button-face-rollover specify caption changes and axf:field-button-icon-rollover specifies icon changes.

[Figure 3.7-4] (p.333) shows an example push button. `axf:field-button-layout="caption-below-icon"` places the icon image above the text. `axf:field-button-icon-rollover` switches the icon when the mouse pointer is over the button, and `axf:field-button-icon-down` when the button is pressed. By specifying `border-style="outset"`, you can easily make the button look more prominent than its surroundings.

```
<!-- Push button with caption below the icon -->
<axf:form-field
  axf:field-type="button"
  axf:field-button-layout="caption-below-icon"
  axf:field-button-face="Antenna House, Inc."
  axf:field-button-icon="url('../img/icon.png')"
  axf:field-button-icon-rollover="url('../img/icon-rollover.png')"
  axf:field-button-icon-down="url('../img/icon-down.png')"

  width="15em" height="3em"
  background-color="#eee"
  border="2px silver outset" />
```

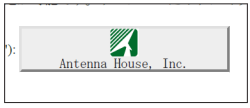


Figure 3.7-4 Push button example

Color change when pressed

By using JavaScript, aspects other than the push button's icon and text can be changed.

```
<!-- Push button to run JavaScript -->
<axf:form-field axf:field-type="button"
  axf:field-name="lock">
  <axf:form-field-event action-type="javascript" name="MouseUp">
    button = this.getField("lock");
    button.fillColor = color.red;
    button.textColor = color.white;
  </axf:form-field-event>
</axf:form-field>
```

The contents of `axf:form-field-event` with `axf:action-type="javascript"` specified are executed when the mouse is released (clicked). When this happens, the field name of the push button is acquired and the field color and the text color are changed.

Note

If you want to switch the display for an odd or even number of clicks, the form action specified by the push button can switch a hidden radio button.

3.7.5 Selection field

You can specify radio buttons, checkboxes, list boxes, combo boxes, and more.

Radio buttons and check boxes

Specify `axf:field-type` as `checkbox` to make a check box, or as `radio` to make a radio button. Use a check box if you want to select multiple items, and use a radio button if you want to select one alternative. The element is empty in both cases. `axf:field-value` specifies the value to use when the field is selected.

To make a set of radio buttons where only one at a time can be selected, specify the same `axf:field-name` value on all of the buttons.

Property	`axf:field-checked`
Value	`false` | `true`
Initial	`false`
Applies to	axf:form-field

Property	`axf:field-checked-style`
Value	`checkmark` | `circle` | `square` | `cross` | `star`
Initial	`checkmark`
Applies to	axf:form-field

`axf:field-checked="true"` makes the field default to the checked state. `axf:field-checked-style` specifies the appearance of a checked item. The shape of the frame does not change.

Note

If you specify `axf:field-checked="true"` for multiple radio button fields in the same set, only one of them will be checked.

```
<!-- Check box specification example -->
<fo:block ...>
  <axf:form-field axf:field-type="checkbox"
    axf:field-name="CheckMe" axf:field-checked="true"
    axf:field-checked-style="checkmark"

    width="1em" border="1px solid silver" /> Check Me!
</fo:block>
<fo:block ...>
 <axf:form-field axf:field-type="checkbox"
  axf:field-name="CheckCross" axf:field-checked="true"
  axf:field-checked-style="cross"... /> ...
</fo:block>
<fo:block ...>
  <axf:form-field axf:field-type="checkbox"
    axf:field-name="CheckStar"
    axf:field-checked="true" axf:field-checked-style="star".../>...
</fo:block>
```

[Figure 3.7-5] (p.335) shows a check box example. Each checkbox is independent because the `axf:field-name` value is different for each. Their `axf:field-checked-style` values are `checkbox`, `cross` and `star`, respectively, and they default to being checked because of `axf:field-checked="true"`.

Figure 3.7-5 Check box example

[Figure 3.7-6] (p.336) shows a radio button example. `axf:field-name="Choice"` is specified for both fields, which places them in the same set. One `axf:field-value` value, `Cat` or `Dog`, becomes the value recorded for `axf:field-name="Choice"`. `axf:border-radius="50%"` is specified to make the frame of the field circular.

```
<!-- Radio button specification example -->
<axf:form-field axf:field-type="radio"
  axf:field-name="Choice" axf:field-value="Cat"
  axf:field-checked="false"
  axf:field-checked-style="circle"

  width="1em" border-radius="50%"
  border="1px solid silver"></axf:form-field> Cat
<axf:form-field axf:field-type="radio" axf:field-name="Choice"
  axf:field-value="Dog" axf:field-checked="false" axf:field-checked
```

```
-style="circle"
  ... ></axf:form-field> Dog
```

Figure 3.7-6 Radio button example

List box and combo box

You can make a list box by specifying `listbox` for `axf:field-type`, or make a combo box (drop-down list) by specifying `combobox`. A list box displays its list items within its area, but when there is not enough space for the items, a scroll bar is displayed.

If you press the initial item displayed for the combo box, which may be blank, the selectable list items will be displayed.

Element	<axf:form-field-option>
Property	field-value
Parent	axf:form-field

Property	axf:field-multiple
Value	false | true
Initial	false
Applies to	axf:form-field

Each <axf:form-field-option> specifies one item in the list box or combo box. They are listed in their order within <axf:form-field>. Values are required for both axf:field-value and the text content of the element.

Property	axf:field-top-index
Value	*<number>*
Initial	1
Applies to	axf:form-field

Property	axf:field-selected
Value	false | true
Initial	false
Applies to	axf:form-field-option

axf:field-top-index specifies the initial selection item (the item selected by default) in the list box or combo box. The numerical value corresponds to the position of an <axf:form-field-option> element. The first element is numbered 1, and axf:field-top-index is considered to be 1 if the specified value is less than 1 or greater than the number of <axf:form-field-option>.

axf:field-multiple specifies whether multiple items in the list box can be selected at once. When true, multiple selections are available.

Property	axf:field-editable
Value	false | true
Initial	false
Applies to	axf:form-field

axf:field-editable specifies whether text in the combo box is editable. When true, it can be edited. It is the text of the <axf:form-field-option> element that is editable, not the value of the axf:field-value property.

[Figure 3.7-7] (p.338) shows list box and combo box examples. By specifying inset for the border, the field can appear recessed from its surroundings.

```
<!--  List box example  -->
<axf:form-field field-type="listbox"
  field-name="ListChoice"
  field-top-index="1"
  field-multiple="true"

  width="10em" height="3em"
  background-color="#eee"
  border="1pt silver inset">
  <axf:form-field-option
    field-value="ja">Japan</axf:form-field-option>
  <axf:form-field-option
    field-value="ko">Korea</axf:form-field-option>
  <axf:form-field-option
    field-value="cn">China</axf:form-field-option>
  <axf:form-field-option
```

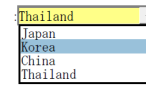
```
    field-value="th">Thailand</axf:form-field-option>
 </axf:form-field>
```

```
<!-- コンボボックスの例 -->
<axf:form-field field-type="combobox"
  field-name="ComboEditable" field-editable="true"
  field-top-index="1"

  width="10em" height="1.2em"
  background-color="#ff8" border="1pt silver inset">
  <axf:form-field-option
    axf:field-value="ja">Japan</axf:form-field-option>
  <axf:form-field-option
    axf:field-value="ko">Korea</axf:form-field-option>
  <axf:form-field-option
    axf:field-value="cn">China</axf:form-field-option>
  <axf:form-field-option
    axf:field-value="th">Thailand</axf:form-field-option>
 </axf:form-field>
```

The `<axf:form-field-option>` in both the combo box and list box examples are the same, but because `axf:field-editable="true"` is specified on the combo box, its content can be edited in the PDF viewer.

List box example



Combo box example

Figure 3.7-7 List box and combo box examples

3.7.6 Digital signature field

`axf:field-type="signature"` specifies a digital signature field. If specified, it is possible to also lock the value entered in the AcroForm.

Property	`axf:field-lock-document`
Value	`auto` | `false` | `true`
Initial	`auto`
Applies to	axf:form-field

`axf:field-lock-document` allows you to control whether to lock the document when it is signed: `auto` causes the PDF viewer to display a user interface such as a dialog asking

whether to lock the document; `false` does not lock the document; and `true`true locks it. A locked PDF cannot be edited.

Document locking with digital signature is valid for PDF 1.7 or later.

```
<!-- Example of digital signature field -->
<axf:form ...>...<axf:form>
<!-- Digital signature field to lock the document -->
<axf:form-field axf:field-type="signature"
  axf:field-name="sig"
  axf:field-lock-document="true"
  background-color="lightblue"
  border="1pt silver solid" width="10em" height="3em"/>
```

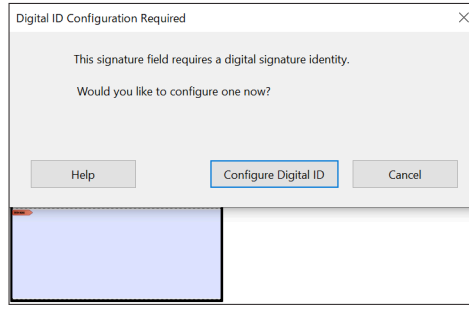


Figure 3.7-8 Digital signature field example

3.7.7 Form actions

You can perform various actions by filling out PDF forms and using JavaScript.

Note

Whether execution of JavaScript is permitted depends on the type of PDF.

Property	`axf:action-type`
Value	`auto` | `goto` | `gotor` | `launch` | `uri` | `reset` | `submit` | `javascript`
Initial	`auto`
Applies to	axf:form-field

Property	axf:field-submit-method
Value	get | post
Initial	get
Applies to	axf:form

Property	axf:field-submit-coordinates
Value	false | true
Initial	false
Applies to	axf:form

The axf:action-type values are auto, goto, gotor, launch and uri for controlling links, and reset, submit and javascript for manipulating the form's values. This section describes the features for value manipulation.

A PDF form on the screen looks like the paper forms that we are used to. The information entered can be stored separately as data, such as simple character strings. In addition, a PDF form is useful when you want to execute JavaScript to, for example, automatically display the total number calculated from the input.

axf:action-type	Definition
reset	Reset the content entered in the form.
submit	Submit the contents entered in the form to the link specified in the external-destination property of the <axf:form> element using HTTP.
javascript	Take action with JavaScript. Specify the JavaScript program in the <axf:form-field-event> element.

When axf:action-type="submit" is specified and axf:field-submit-coordinates="true", the mouse coordinates are included in the submission.

When <axf:form-field-event> is not specified, MouseUp is assumed.

```
<!-- Submit the coordinates of the mouse and child field contents t
o the URL of external-destination - -->
<axf:form
  axf:external-destination="..."
  axf:field-submit-method="post" axf:field-submit-coordinates="true
">
  <fo:block ...>
    <!-- Input text field  -->
```

```
    <axf:form-field axf:field-type="text"
      axf:field-name="InputText" ... width="10em" height="2.4em" ..
. />
  </fo:block>
  ...
  <fo:block ...>
  <!-- Button to send input -->
    <axf:form-field axf:field-type="button"
      axf:field-name="Submit" axf:action-type="submit" />
  </fo:block>
</axf:form>
```

User operation events

Element	<axf:form-field-event>
Property	name, action-type

<axf:form-field-event> is a child of <axf:form-field>. All properties are required.

name specifies the event type, and action-type allows the same values as for <axf:form-field>.

Specify this element when you want to interact with the form field using other than a mouse click. This takes precedence over the action specified in action-type on <axf:form-field>.

name	Definition
MouseUp	The action occurs when a mouse button is released on the form field.
MouseDown	The action occurs when a mouse button is pushed on the form field.
MouseEnter	The action occurs when a mouse cursor enters the form field.
MouseExit	The action occurs when a mouse cursor exits the form field.
OnFocus	The action occurs when a form field gets focus.
OnBlur	The action occurs when a form field loses focus.

For a push button, pressing and releasing the button is MouseUp, and pressing the button is MouseDown. MouseEnter is the action when you hover your mouse over the button, and MouseExit is the action when you move it away from the button.

Note

The state where the mouse is on the push button is “RollOver”.

`OnFocus` and `OnBlur` relate to, for example, a text field being "in focus" when you type a character in it.

Disallow JavaScript execution

Option Property	`allow-javascript`
Value	`true` | `false`
Initial	`true`
Applies to	`<pdf-settings>`

When `allow-javascript` is `false`, the JavaScript specified in `<axf:document-info name="openaction">` and `axf:action-type="javascript"` form fields will be ignored.

3.7.8 Disallow filling and signing

You can disable user input and signature on the entire PDF form.

Option Property	`no-fill-form`
Value	`true` | `false`
Initial	`false`
Applies to	no

`no-fill-form` specifies whether to allow the user to input and sign the entire form of the output PDF file.

If `true`, changes will not be allowed. `owner-password` must be set to enforce this restriction.

This is useful when you want to embed a PDF containing a PDF form, such as an example of a form, but you want to prohibit input to the form.

Section 3.8 Tagged PDF

You can enable Tagged PDF output using option settings, the GUI or command line parameters.

A PDF with an embedded document structure is called a “Tagged PDF”.

If you select Tagged PDF in the AH XSL Formatter output settings, the PDF output will be tagged. You can create a properly structured Tagged PDF based on the FO structure with no further effort.

Option Property	`tagged-pdf`
Value	`true` | `false`
Initial	`false`
Applies to	`<pdf-settings>`

When the `tagged-pdf` option is `truw`, the output PDF will be a Tagged PDF. This setting is ignored for some versions and types of PDF output.

In the GUI, check the [**"Tagged PDF"**] item in [**"Format"**] - [**"PDF Option Setting (P)"**].

For more information on Tagged PDF, see the Manual, ISO 32000-1:2008 and ISO 32000-2:2017.

Tagged PDF is valid in PDF 1.4 or later.

3.8.1 PDF tags

You can explicitly specify the PDF tag for an object. You can also create your own tag mapping.

Alongside automatic tagging by AH XSL Formatter, you can tag specific objects.

Property	`axf:pdftag`
Value	<*string*>
Initial	empty string
Applies to	all formatting objects

`axf:pdftag` specifies the tag name.

Specify the standard PDF tag, such as `axf:pdftag="P"`, or use a non-standard tag.

Option Property	`new-tagging-mode`
Value	`true` | `false`
Initial	`false`
Applies to	`<pdf-settings>`

Option Element	`<tag-role-map>`
Parent	`<pdf-settings>`

Option Element	`<tag-element>`
Property	`tag-base-name`, `tag-derived-name`
Parent	`<tag-role-map>`

Option Property	`tag-base-name`,`tag-derived-name`
Value	*<string>*
Initial	N/A, a value is required
Applies to	`<tag-element>`

When the `new-tagging-mode` option is `true`, if the tag is an empty string (`axf:pdftag="''"`), the element does not generate a tag and is tagged as part of its parent area.

The `<tag-role-map>` option element can map from a non-standard tag name to a standard tag name. In each `<tag-element>` child of `<tag-role-map>`, the tag name in `tag-derived-name` is mapped to the tag name in `tag-base-name`. This base tag name can be further mapped to a different tag by another `<tag-element>` in the `<tag-role-map>`.

Note

PDF/UA prohibits mapping standard tag names to any other tag name. Non-standard tag names must map, directly or indirectly, to a standard tag name.

3.8.2 Tagged PDF and PDF 2.0 or later

PDF 2.0 adds constraints on the parent-child relationship between tags. AH XSL Formatter can warn you when tags violate those constraints.

Option Property	check-tag-relationship
Value	true | false
Initial	false
Applies to	<pdf-settings>

When the check-tag-relationship option is true>, the parent-child relationships are verified when outputting tagged PDF 2.0. AH XSL Formatter generates a warning when the structure is incorrect.

3.8.3 Alternate text for images and links

You can specify alternate text to use when an image cannot be displayed, when a link cannot be displayed or when the PDF is read by a screen reader.

<fo:external-graphic> Alternate text is an accessibility requirement for image elements that do not have text content, such as <fo:external-graphic>. This alternate text should be explicitly specified. When it is not specified, AH XSL Formatter inserts SPACE as the alternate text so that an accessibility check will not show an error.

Property	axf:alttext
Value	<*string*>
Initial	empty string
Applies to	fo:external-graphic, fo:instream-foreign-object, fo:basic-link

axf:alttext specifies alternate text.

axf:annotation-contents specifies alternate text for a link. If this is not specified or is empty, the axf:alttext value will be used.

Depending on the axf:display-alttext setting, when an image does not exist, the axf:alttext value is also displayed as formatted text in its place.

Property	axf:display-alttext
Value	true | false | auto
Initial	auto
Applies to	fo:external-graphic, fo:instream-foreign-object, fo:basic-link

Option Property	display-alttext
Value	true | false
Initial	false
Applies to	<formatter-settings>

When the image specified by the src property is not found, if axf:display-alttext is true, the axf:alttext value will be displayed. If false, an alternate image is displayed.

If axf:display-alttext="auto" is specified, the display-alttext option value is used.

3.8.4 Script for a structural element

In Tagged PDF, structural elements have an associated language setting. AH XSL Formatter supports this feature.

When the language and country properties or the xml:lang attribute specify the language of an object, that language is used as the language of the structural element.

Detailed language settings, especially for documents which mix multiple scripts, can be an effective accessibility support in addition to improving typography.

```
<!--  FO with a mixture of multiple languages -->
<fo:block xml:lang="ja">
「おはよう」はエスペラントで「<fo:inline ...
  xml:lang="eo">Bonan Matenon.</fo:inline>」です。
</fo:block>
```

3.8.5 Headers and cells in the table

You can mark up explicit relationships between a table cell and its headers.

When AH XSL Formatter generates Tagged PDF, each tag is for the current object. In tables, you may want to add relationships between table headers and table cells that go beyond what can be inferred.

Property	axf:scope
Value	auto | row | col | rowgroup | colgroup | both
Initial	auto
Applies to	fo:table-cell

Property	axf:headers
Value	*<idref>*
Initial	*empty*
Applies to	fo:table-cell

axf:scope is set on a header cell, and axf:headers is set on content cell. They correspond to headers and scope in HTML.

When axf:scope is col, it indicates that the header cell applies to following cells in the same column; row, that it applies to following cells in the same row. If number-columns-spanned or number-rows-spanned on the header cell is greater than 1, colgroup or rowgroup indicates that the corresponding cols or rows are collectively headers. both indicates that the header applies to both the column and row that contains it. This is useful on the "header for headers" in the upper-left cell of a table that has both head rows and head columns.

```
<!-- Describe the relationships between cells and headers of table
with column and row headings  -->
<fo:table>
  <fo:table-header>
    <fo:table-row>
      <fo:table-cell axf:scope="col" id="head-heading" >
        ...
      </fo:table-cell>
      <fo:table-cell axf:scope="col" id="head-v1" >
        ...
      </fo:table-cell>
      <fo:table-cell axf:scope="col" id="head-v2" >...</fo:table-ce
ll>
    </fo:table-row>
  </fo:table-header>
  <fo:table-body>
    <fo:table-row>
      <fo:table-cell axf:scope="row" id="head-h" >...</fo:table-cel
l>
      <fo:table-cell axf:headers="head-v1 head-h" >...</fo:table-ce
ll>
      <fo:table-cell axf:headers="head-v2 head-h" >...</fo:table-ce
ll>
    </fo:table-row>
    ...
  </table-body>
</fo:table>
```

3.8.6 Table summary

You can include an equivalent of the `summary` property in the table element of HTML 4.01.

Property	`axf:table-summary`
Value	*<string>*
Initial	*empty*
Applies to	fo:table

`axf:table-summary` contains a simple text summary of the table. It is output in Tagged PDF.

3.8.7 Accurate text extraction

Tagged PDF includes the actual source text from before it is processed for display.

When you want to search in a PDF, it is useful if the PDF contains the original text, before it has been processed for display; that is, without hyphenation or changes caused by text-transform. Text search can also be difficult in complex scripts such as Arabic and Khmer that undergo complex Unicode code point conversion, or Thai, etc. where accent marks are processed using the font's GPOS feature.

Tagged PDF includes the unprocessed text in your PDF as ActualText.

3.8.8 Text access for screen readers

You can restrict whether a screen reader can read text or select and copy the text.

Option Property	`no-accessibility`
Value	`true` | `false`
Initial	`false`
Applies to	`<pdf-settings>`

When the `no-accessibility` option is `true`, the output PDF file restricts text access from screen reader devices. The `owner-password` option must be set when using this restriction.

`no-accessibility` is valid with PDF 1.4 and later.

Chapter 4

Other Extensions

Section 4.1 Printing

Here explains about controls related to the printing process.

4.1.1 Trim size

Specify the distance from the edge of the paper to the trim size. It also controls the drawing of objects that extend outside the trim size.

The trim size is the size specified in the `page-width`, `page-height` or `size` properties. Depending on the printing method, you may need a distance from the outside of this trim size to the edge of the page.

Property	`axf:crop-offset`, `axf:crop-offset-top`, `axf:crop-offset-bottom`, `axf:crop-offset-left`, `axf:crop-offset-right`
Value	<*length*>
Initial	`0pt`
Applies to	fo:simple-page-master

Specify the distance from the outside of the trim size to the physical edge in the `axf:crop-offset*` property.

Specify `axf:crop-offset` if you want to keep the same distance for up, down, left and right, and specify each of the `axf:crop-offset-*` properties if you want to specify them individually. If the `axf:crop-offset` property is specified at the same time as the individual setting, the individual value takes precedence.

If `page-width="182mm"` is specified and `axf:crop-offset-left="14mm"` and `axf:crop-offset-right="14mm"` are set, the physical paper size is 210mm wide.

Area that extends outside the page

Property	`axf:crop-area-visibility`
Value	`hidden` | `visible`
Initial	`hidden`
Applies to	fo:simple-page-master

Specifying `visible` for the `axf:crop-area-visibility` property displays the area that extends outside the trim size or bleed area. When `hidden` is specified, the area is clipped at the trim size or at the bleed area.

4.1.2 Width of the bleed area

Specify the width of the bleed area for cutting off.

When placing images and colors up to the edge of the paper, printing on the paper surface may cause misalignment due to expansion and contraction of the paper or cutting processing. If this happens, the area where images and colors are placed wider than finished size, called bleed area is taken so that there are no gaps in printing due to misalignment.

Property	`axf:bleed`, `axf:bleed-top`, `axf:bleed-bottom`, `axf:bleed-left`, `axf:bleed-right`
Value	<*length*>
Initial	`0pt`
Applies to	fo:simple-page-master

The width of the bleed area outside the finished area in the `axf:bleed*` property. If you do not specify the `axf:printer-marks*` property, the area borders will not be displayed. Specify the top, bottom, left and right widths with the `axf:bleed-top`, `axf:bleed-bottom`, `axf:bleed-left` and `axf:bleed-right` properties.

Specify the same value for the `axf:bleed` property on top, bottom, left and right. If specified at the same time as individual setting, the individual value has priority.

4.1.3 Printer marks

Display registration marks and Japanese printer marks (crop marks and bleed marks).

This is an extension of printer marks that displays the trim size and the position of bleed area on the paper.

Printer marks

Property	`axf:printer-marks`
Value	[[`crop` | `crop-trim`] || [`cross` | `cross-circle` | `cross-registrartion`] || <*uri-specification*>+] | `none`
Initial	`none`
Applies to	fo:simple-page-master

Specify the type of printer mark to be output in the `axf:printer-marks` property. Use crop to display the corner mark, `crop-trim` to display the *corner marks only to the position of the trim size (crop marks)*, `cross` to display the *cross-shaped marks*, `cross-circle` to

display the *cross-shaped marks with concentric circle* and `cross-registration` to display the *registration marks* [Figure 4.1-1] (p.352).

By specifying *<uri-specification>*, it is possible to prepare and output your own printer mark image and color bar. You can specify multiple URIs.

It is also possible to specify as `axf:printer-marks="cross url('./color-bar.svg')"` to output the image of the cross-shaped mark and the color bar at the same time.

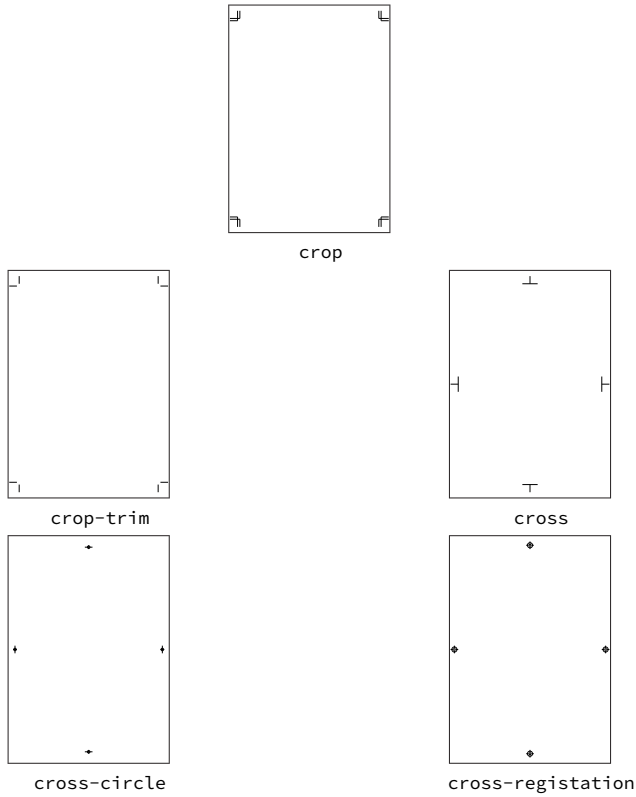


Figure 4.1-1 `axf:printer-marks`

Width of the two-page spread

Property	axf:printer-marks-spine-width
Value	<*length*>
Initial	0pt
Applies to	fo:simple-page-master

When a value larger than 0 is specified for the axf:printer-marks-spine-width property, the center mark of the spine is changed to the spine-width printer mark , assuming that the printing area is spread on the left and right. The left and right page widths are the same and the spine-width printer mark is drawn at the center position.

Printer mark line styles

Property	axf:printer-marks-line-color
Value	<*color*> | auto
Initial	auto
Applies to	fo:simple-page-master

Specify the color of the printer mark line in the axf:printer-mark-line-color property. This is useful when it is specified on the print side or when you want to distinguish the line when the page has black background. When set to auto, registration colors are printed, and when CMYK is used, all four colors are printed with a value of 100%.

Property	axf:printer-marks-line-width
Value	<*length*> | auto
Initial	auto
Applies to	fo:simple-page-master

Option Property	printer-marks-line-width
Value	<*length*>
Initial	0.24pt
Applies to	<formatter-settings>

The printer mark line width can be specified in the `axf:printer-marks-line-width` property. When `auto` is specified, the value specified in the `printer-marks-line-width` property in the option setting is used.

Property	`axf:printer-marks-line-length`
Value	*<length>* | `auto`
Initial	`auto`
Applies to	fo:simple-page-master

Option Property	`printer-marks-line-length`
Value	*<length>*
Initial	`10mm`
Applies to	`<formatter-settings>`

The printer mark length can be specified in the `axf:printer-marks-line-length` property.

When `auto` is specified, the value specified in the `printer-marks-line-width` property in the option setting is used.

The size of the circle is also adjusted with `axf:printer-marks="cross-circle"`.

Margin between the printing area and the printer mark

Property	`axf:printer-marks-zero-margin`
Value	*<length>* | `auto`
Initial	`auto`
Applies to	fo:simple-page-master

Option Property	`printer-marks-zero-margin`
Value	*<length>*
Initial	`3mm`
Applies to	`<formatter-settings>`

Specify the distance between the printing area and the printer mark when the value of the `axf:bleed*` property is 0 in the `axf:printer-marks-zero-margin` property. When `auto` is specified, the value of the `printer-marks-zero-margin` property is used.

4.1.4 Overprint

Specify overprinting to print with overlapping colors.

When arranging objects, there are places where colors overlap, such as background images and text. In printing, it may cause a difference whether putting the colors on top of each other according to this arrangement or remove the colors under the overlap. The method of stacking them is called overprinting. The method of removing the color below is called trapping. If overprint is specified when outputting PDF, the front color and back color of the overlapped part are retained.

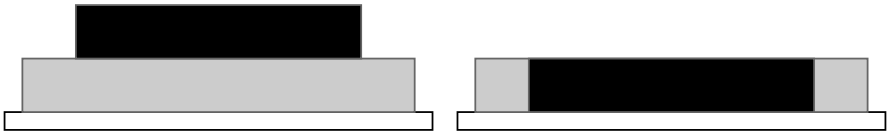


Figure 4.1-2 Illustration of overprint (left) and trapping (right)

Property	`axf:overprint`
Value	`auto` | <*overprint*>`#`
Initial	`auto`
Applies to	all formatting objects
Inherited	yes

Option Property	`overprint`
Value	<*overprint*>`#`
Initial	*empty*
Applies to	`<pdf-settings>`

Value	**Value Composition**
<*overprint*>	<*opcolor*>[[`stroke` || `paint` || [`full` | `nonzero`]] | `none`]] ?
<*opcolor*>	<*color*> | `k100` | `separation` | `all`

Specify the object's overprint to the `axf:overprint` property. Only valid for PDF output.

When `axf:overprint="auto"` is specified, specify the color to which the overprint is applied to the `overprint` property in the option setting. If `overprint="k100"` is specified, all pages of FO will be overprinted.

axf:overprint	dEfinition
<*opcolor*>	Specifies a color you want to apply the overprint.
`stroke`	Applies the overprint for the line stroke.
`paint`	Applies the overprint for the paint.
`full`, `nonzero`	Specifies the operation in case the color ingredient is 0 when applying the overprint. When `full` is specified, the color ingredient is set to 0, when `nonzero` is specified, the color ingredient is not changed. (It is considered no color.) This setting is effective only with CMYK.
`none`	The overprint is not applied.

<*color*>, `k100`, `separation` and `all` excluding transparent color can be specified as <*opcolor*>.

`k100` applies an overprint to `rgb-icc(#CMYK,0,0,0,1)`.

`separation` applies an overprint to the separation color indicated by `rgb-icc(#Separation)`. If this applies, the registration color is not included.

`all` applies the overprint to all non-transparent colors.

When only <*opcolor*> is specified, it is considered that `stroke paint nonzero` is specified at the same time. Specify `none` to exclude the object from overprint.

Multiple overprints can be specified in pairs separating with comma, but when a certain color can be interpreted as multiple settings, the settings will be investigated and the matched setting is adopted in the priority order of <*color*>, `k100`, `separation` and `all`.

The effect of overprint changes depending on the front color and back color settings. When the background color is a separation color, the foreground color with overprint will be almost certainly enabled.

For other details, see the manual.

Note

When embedding a PDF, the source PDF is not processed. If you want to overprint, embed the PDF to which the overprint has been applied in advance.

4.1.5 Disallow printing and printable resolution in PDF

Output a PDF that specifies disallowance of printing and printable resolution. It depends on the PDF viewer whether printing is actually disallowed.

Option Property	printing-allowed
Value	high-resolution | low-resolution | none
Initial	high-resolution
Applies to	<pdf-settings>

Specify whether to print the resulting PDF file or not in the printing-allowed property in the option setting. When none is specified, printing is not allowed. high-resolution allows printing in high resolution. low-resolution only allows printing in low resolution.

low-resolution is valid in PDF 1.4 or later and is considered high-resolution in PDF 1.3.

When limiting the allowance, a value should be set in the owner-password property.

4.1.6 Printer operation in the Windows version

In the Windows version, you can select both sides printing, paper tray and etc. for each page sequence when printing. Whether it is valid or not depends on the printer.

Property	axf:printer-bin-selection
Value	<*string*> | <*integer*>
Initial	depends on the environment
Applies to	fo:page-sequence

Specify the printer tray name or tray number in the axf:printer-bin-selection property. This tray name appears in the printer settings dialog and is obtained from the printer driver. The tray number is also obtained from the printer driver. It is invalid when it does not match what can be obtained by the printer driver.

For example, when pages with different page sizes are mixed, it is possible to print in the size corresponding to each page using this property.

Property	axf:printer-duplex
Value	<*integer*>
Initial	depends on the environment
Applies to	fo:page-sequence

Option Property	SeparatePrinterDuplexJob
Value	true | false
Initial	true
Applies to	<formatter-settings>

Specify a numerical value in the axf:printer-duplex property. Because the value specified in the printer driver is specified, the correspondence of numerical values may differ depending on the printer. For many printers, 1 is one-side printing, 2 is both-sides with long-edge binding, and 3 is both-sides with short-edge binding.

The SeparatePrinterDuplexJob property in the option setting controls whether printing is separated or output in a batch when this print setting is switched. Set true for separating and false for batch printing.

```
<!--  Specifies to use a different tray when printing figures and t
ables with different page sizes  -->
<fo:page-sequence master-reference="A4"
    axf:printer-bin-selection="1"
    axf:printer-duplex="2">
    ...
    <fo:page-sequence master-reference="A3broadtable"
        axf:printer-bin-selection="2">
    ...
    </fo:page-sequence>
</fo:page-sequence>
```

4.1.7 Use case: Add printer marks to the cover including the spine

Create a front cover page including the spine and back cover by adding printer marks.

Create the front cover, spine and back cover as a single page [Figure 4.1-3] (p.359).

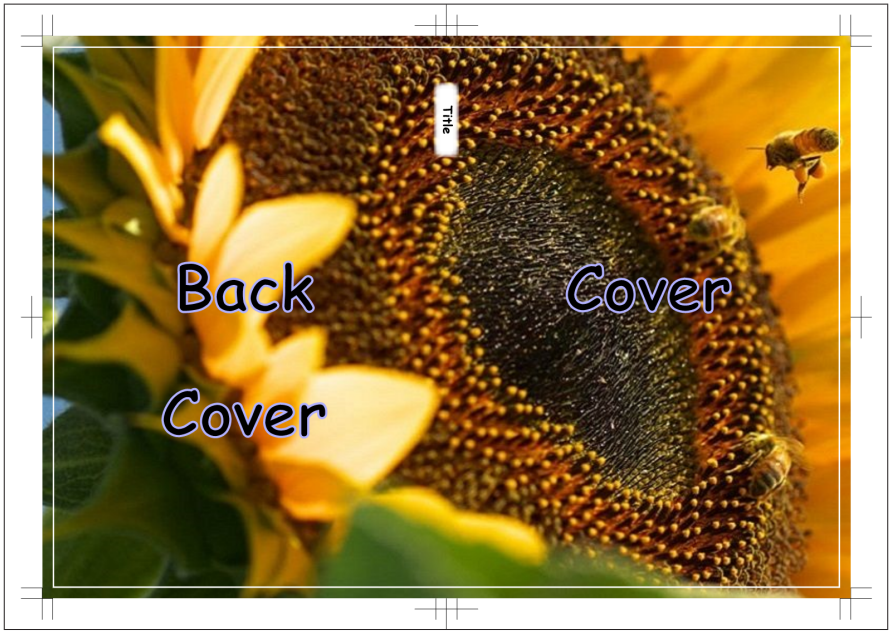


Figure 4.1-3 Add printer marks to the cover including the spine

Specify the page width of the cover page

```
<!-- Cover page master -->
<fo:simple-page-master
  page-height="257mm" page-width="182mm + 182mm + 10mm"
  margin="0mm" ...>
  ...
</fo:simple-page-master>
```

`page-width` specified in the page master is the total value of the front cover width, back cover width and spine width. In order to specify the spine width, it is necessary to know the thickness of the book when completed in advance.

Note

`<axf:spread-page-master>` is a feature that assumes elements to span across two pages. It is not suitable for outputting a cover with a spine as a single page.

Add the printer marks

```
<!-- Bleed area and printer marks  -->
<fo:simple-page-master
  ...
  axf:crop-offset-top="(297mm -257mm) div 2"
  axf:crop-offset-bottom="(297mm -257mm) div 2"
  axf:crop-offset-left="(420mm - 364mm - 10mm) div 2"
  axf:crop-offset-right="(420mm - 364mm - 10mm) div 2"
  axf:bleed="5mm"
  axf:printer-marks-spine-width="10mm"
  axf:printer-marks="crop cross"
  ...>
  ...
</fo:simple-page-master>
```

Specify the area outside the trim size with `axf:crop-offset*`, and specify the bleed area with `axf:bleed*`. Then, specify the display of various printer marks with `axf:printer-marks`. The page size other than the cover is JIS B5 and the long edge is the height, and the page size of the cover is doubled in width and added to the width of the spine. A value is specified for the `axf:crop-offset-*` property so that the physical size is A3 size.

Specify the background that considered the bleed area throughout the page

```
<!-- Specify the background for the page master -->
<fo:simple-page-master ...
  background-image="url(./background.jpg)"
  axf:background-image-resolution="300"
  axf:background-size="cover"
  axf:background-position-horizontal="center"
  axf:background-position-vertical="center"
  axf:background-repeat="no-repeat">
  ...
</fo:simple-page-master>
```

The background is the area that reflects the value specified by `axf:bleed` to the value specified by the `page-width` and `page-height` properties.

Draw the border of the body region

```
<!-- region-body border -->
<fo:region-body
  margin="0"
  border="solid 2pt cmyk(0, 0 , 0 ,0)" />
```

The border of the body region is displayed for the sake of clarify. In the XSL 1.1 specification, the width of `padding` and `border` to `<fo:region-*>` is 0, so the border cannot be drawn, but it is not limited by AH XSL Formatter.

Place the front cover, back cover and spine with block container

```
<!-- Place the front cover, back cover and spine with block contain
er -->
<fo:flow ...>
  <!-- Front cover -->
  <fo:block-container absolute-position="absolute"
    width="182mm" top="33vh" left="182mm + 10mm -2pt">
    <fo:block ...>Cover</fo:block>
  </fo:block-container>
  <!-- Back cover -->
  <fo:block-container absolute-position="absolute"
    width="182mm" top="33vh" left="0 -2pt">
    <fo:block ...>Back <fo:block />
      Cover</fo:block>
  </fo:block-container>
  <!-- Spine -->
  <fo:block-container
    width="10mm" height="257mm"
    absolute-position="absolute" top="0" left="182mm - 2pt">
    <fo:block-container ... reference-orientation="-90">
      <fo:block text-align="center"
      background-color="cmyka(0, 0, 0, 0, 1)"
      axf:box-shadow="1mm 1mm 3pt 1mm cmyka(0,0,0,0,0.8),
        1mm -1mm 3pt 1mm cmyka(0,0,0,0,0.8),
        -1mm 1mm 3pt 1mm cmyka(0,0,0,0,0.8),
        -1mm -1mm 3pt 1mm cmyka(0,0,0,0,0.8)"
      font-size="6mm" line-height="1" ... >
      Title
      </fo:block>
    </fo:block-container>
```

```
  </fo:block-container>
</fo:flow>
```

The text is placed on the front and back covers with the absolute position of the block container. In horizontal writing, the front cover position is the sum of the page width of the back cover and the spine from the left edge. The back cover is the starting point on the left edge. The spine is the page width of the back cover from the left edge.

The reason why it is set to `- 2pt` is to adjust the width of the border of the body region.

For the text direction of the spine, a block container specified `reference-orientation="-90"` is specified as a child of the block container for absolute position. Since the characters are smaller, the background color is specified instead of the outline. A box shadow is specified to blur the border between the title background color and the image.

4.1.8 Use case: Place the index tab including bleed area

Place the background including the bleed area by the absolute position of the block container.

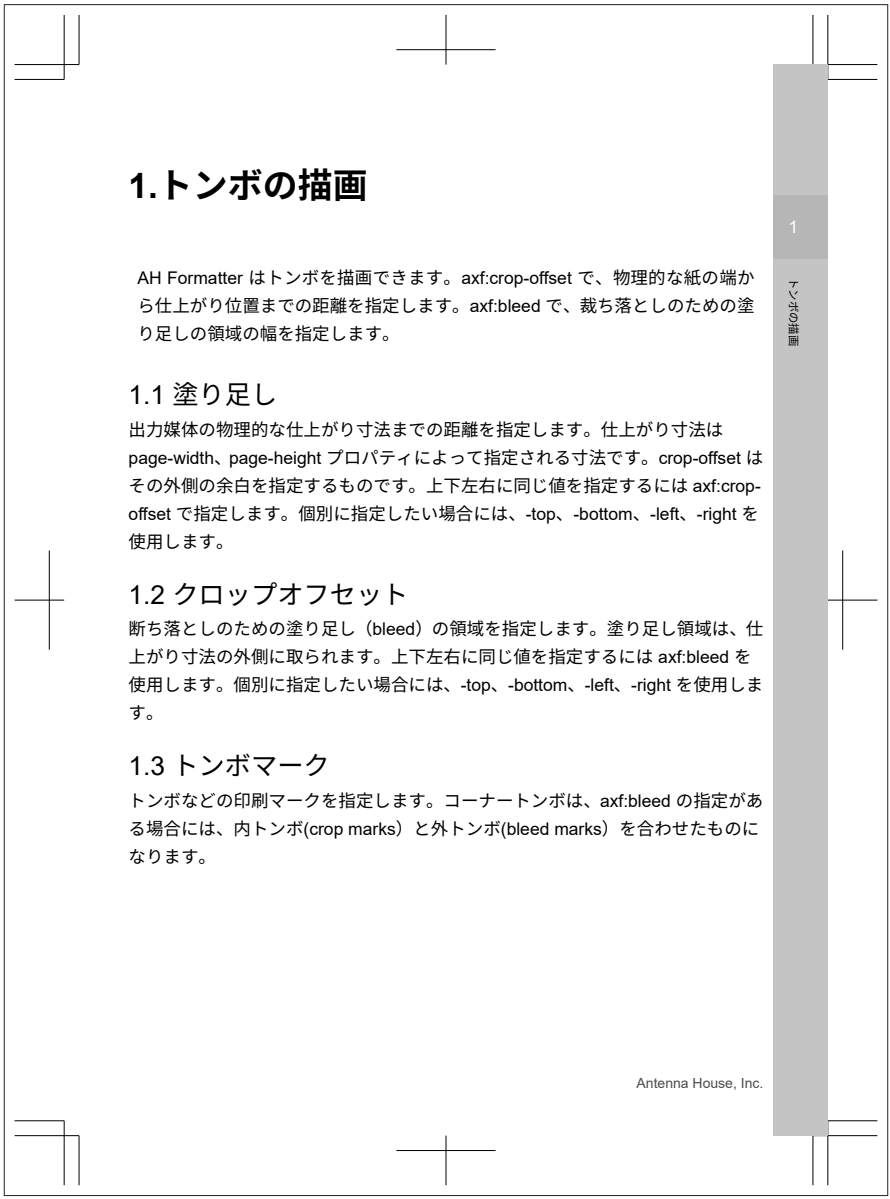


Figure 4.1-4　Place the index tab including bleed area

If it is the background to the page master, the background range will be expanded when `axf:bleed` is specified. No special changes are required for bleed area, except for specifying the image resolution.

However, the header, footer and index tab specified as child of `<fo:region-*>` will be drawn inside `<fo:region-*>` unless otherwise specified, and a blank space will be created at the bleed area.

Specify the block container so that it extends beyond the region, and draw the bleed area [Figure 4.1-4] (p.364).

Specify the printer mark

```
<!-- Specify the printer mark to the page master  -->
<fo:simple-page-master
  page-width="148mm" page-height="210mm"
  margin="0"
  axf:crop-offset-top="15mm" axf:crop-offset-bottom="15mm"
  axf:crop-offset-left="15mm" axf:crop-offset-right="15mm"
  axf:crop-area-visibility="true"
  axf:bleed="3mm"
  axf:printer-marks="crop cross"
  axf:printer-marks-line-color="rgb-icc(#CMYK, 0%, 0%, 0%, 100%)"
  axf:printer-marks-line-length="auto"
  axf:printer-marks-line-width="auto">
...
</fo:simple-page-master>
```

Specify `axf:crop-area-visibility="visible"` in the page master. The printer marks are drawn by specifying the `axf:crop-offset-*`, `axf:bleed` and `axf:printer-marks*` properties.

Specify the region for the index tab

```
<!-- Change region priority for index tab  -->
<fo:region-before ...
  extent="10mm" display-align="before"
  precedence="false" ... />
  <fo:region-end region-name="rgnEndRight"
  extent="10mm" display-align="before"
  reference-orientation="-90" />...
```

`false` is specified for precedence of the `<fo:region-before>` element so that the background of the index tab does not overlap with the header. By this setting, the drawing of `<fo:region-end>` is prioritized.

Absolute positioned index tab with the block container

```
<!-- Move the block container for bleed area -->
<fo:static-content flow-name="rgnEndRight">
  <fo:block-container position="absolute"
    top="-3mm" left="-3mm" right="-3mm"
    height="13mm">
    ...
  </fo:block-container>
</fo:static-content>
```

By `reference-orientation="-90"`, the right side of the page is `top` of the block container, the upper side is `left`, and the lower side is `right`. Because a 3 mm bleed area is specified for the axf:bleed property, it is shifted by 3 mm. Because the origin of the block container is a position that does not include bleed area, a negative value is specified. The height is 13 which adds 3 mm to `<fo:region-end>`.

Index tab styles

```
<!-- Decorate the index tab -->
<fo:block
  line-height="1"
  background-color="rgb-icc(#CMYK, 39%, 8.9%, 0%, 7.5%)">
    <fo:inline-container width="12.8mm"
    height="8mm + 3mm"
    alignment-baseline="text-after-edge">
    <fo:inline-container>
      <fo:block-container ... reference-orientation="90">
        ...
        <fo:inline>1</fo:inline>
        ...
      </fo:block-container>
    </fo:inline-container>
    <fo:inline-container>
      <fo:block font-size="6.5pt">
        <fo:inline baseline-shift="8pt">
          <fo:retrieve-marker retrieve-class-name="index-A" />
        </fo:inline>
    </fo:block>
  </fo:inline-container>
  ...
</fo:block>
```

The number displayed on the index tab should be at most 4 digits, and the chapter header to be referenced should not exceed 1 line. Specify `line-height="1"` to adjust the margins in the inline container. In order to make the chapter numbers upright, `reference-orientation="90"` is specified in the block container and put it back in the same direction as the page. For chapter header references, specify `baseline-shift` to have a margin at the bottom.

If you want to shift the position of the index tab for each chapter, it is sure to change the page sequence and specify a new index tab that has the shifted position.

Section 4.2 Watermark

Insert a watermark on the page.

There are two possible ways when you want to insert a watermark in the output PDF or printed document to specify your rights or restrictions. One is to specify an SVG image as the page background image, and the other is to specify the watermark text in the option settings.

Specify the background image as a watermark

```
<!-- Express the watermark with a background image - -->
<fo:region-body ...
  background-image="watermark.svg"
  background-position-horizontal="center" />
```

Specify the background image in the page master and body region to express the watermark. It is better to use SVG images when you want to display a specific character string repeatedly or when you want to use the same watermark image for different page sizes. It is also possible to specify multiple images in the `background-image` property and add another image that you want to use as the page background after the watermark image.

Specify watermark text in option settings

Option Property	`watermark-text`, `watermark2-text`
Value	*<string>*
Initial	*empty*
Applies to	`<formatter-settings>`

Option Property	`watermark-font-family`
Value	[*<family-name>* | *<generic-family>*]#
Initial	`sans-serif`
Applies to	`<formatter-settings>`

Option Property	watermark2-font-family
Value	[*<family-name>* | *<generic-family>*]#
Initial	the watermark-font-family name
Applies to	<formatter-settings>

Option Property	watermark-font-style, watermark2-font-style
Value	normal | italic
Initial	normal
Applies to	<formatter-settings>

Option Property	watermark-font-weight, watermark2-font-weight
Value	normal | bold | *<integer>*
Initial	normal
Applies to	<formatter-settings>

Because it is a feature to insert text with transparency into the output, it cannot be applied to the type and version of PDF that does not include transparency in the output.

There are two types of watermark text depending on the option settings. The `watermark-*` property controls the watermark text that is inserted diagonally through the page, and the `watermark2-*` property controls the text that is inserted at the bottom of the page.

Specify the text to insert as a watermark in `watermark*-text`. In `watermark-text`, multiple lines are available separated by a line break character reference (`
`).

Only simple text can be specified. Non-simple, complex text includes text that cannot be output in a single font and complex scripts such as Thai and Arabic.

`watermark*-font-weight` The number that can be specified for the watermark*-font-weight property is an integer value of 1 to 1000.

The font family of each watermark text is specified by `watermark*-font-family`. At this time, if the `watermark*-font-family` property is not specified properly for the text, it may not be displayed correctly.

Note

As the number of characters increases, the width is narrowed, so a large number of characters per line affects the readability.

```
<!-- Watermark by option setting -->
<fo:declarations>
  <axf:formatter-config
    xmlns:axs="http://www.antennahouse.com/names/XSL/Settings">
    <axs:formatter-settings
      watermark-text="Antenna House&#10; Watermark"
      watermark-font-family="'Impact'"
      watermark-font-weight="Bold"
      watermark-font-style="italic"
      watermark2-font-family="'Noto Serif,serif'"
      watermark2-text="Watermark 2"/>
  </axf:formatter-config>
</fo:declarations>
```

An example of applying the above settings is shown in [Figure 4.2-1] (p.370). A gray background color is specified to make it easier to see.

Note

The text with the `watermark-text` property is displayed with a white outline, and the text with `watermark2-text` is displayed in gray, so it may not be visible depending on the background color.

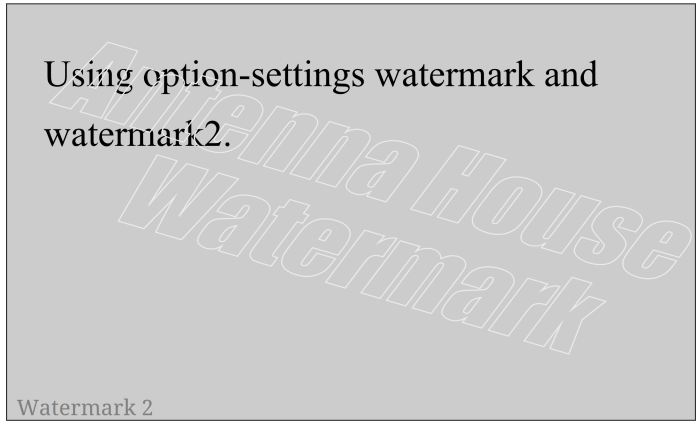


Figure 4.2-1 Watermark by option setting

Section 4.3 Base of a relative URI in the document

Specify the base of the URI specified in the document as a relative path.

Property	`axf:base-uri`
Value	*<uri-specification>*
Initial	empty string
Applies to	all formatting objects
Inherited	yes

For relative URIs in FO, the base URI can be specified in the `axf:base-uri` property. The same process is performed when a value is specified for the `xml:base` property. This is useful when the resource path is different between the production environment and the test environment.

```
<!-- Specify the base of the URI -->
<fo:root ...
  axf:base-uri="https://antennahouse.co.jp/AHF/">
  ...
  <fo:basic-link external-destination="path/to/AHFbook.pdf">
  ...</fo:basic-link>
  <fo:wrapper ... axf:base-uri="/data/Antenna/images/" >
  ...
  <fo:external-image href="uri_sample.pdf" />
  </fo:wrapper>
</fo:root>
```

In the above FO, `https://antennahouse.co.jp/AHF/` is specified in the `axf:base-uri` property in the `<fo:root>` element, and the base of the relative URI is this value.

Because `/data/Antenna/images/` is specified in the `axf:base-uri` property in the `<fo:wrapper>` element, the base URI is `/data/Antenna/images/` in this `<fo:wrapper>` and its child elements.

Section 4.4 Setting options in FO

Element	<axf:formatter-config>
Property	src
Parent	fo:declarations

Option settings can be defined as descendent elements of <axf:formatter-config> or in an external XML file referred to by its src property. Note that option settings are in their own namespace: http://www.antennahouse.com/names/XSL/Settings

Some option settings, such as those related to initialization, are ignored in <axf:formatter-config>. The supported option elements are: <formatter-settings>, <pdf-settings>, <text-settings>, <svg-settings>, <mif-settings>, <ps-settings>, <mathml-settings>, <cgm-settings> and their descendents.

```
<!-- Set options in FO -->
<fo:declarations>
  <axf:formatter-config
     axs="http://www.antennahouse.com/names/XSL/Settings">
    <axs:formatter-settings ... />
    <axs:pdf-settings ... />
</fo:declarations>
```

Disable setting options

Option Property	axf-formatter-config
Value	true | false
Initial	true
Applies to	<formatter-settings>

When the axf-formatter-config option is false, setting options using <axf:formatter-config> in the FO is invalid.

Section 4.5 Error message display

You can stop individual formatting error messages from being displayed on the GUI or command line.

For some errors, such as font fallback or when an external image does not exist, AH XSL Formatter will emit an error message but continue formatting.

Option Property	`no-disp-warnings`
Value	[<*decimal*> | <*hex*>]] *
Initial	*empty*
Applies to	`<formatter-settings>`

The `no-disp-warnings` option lists the decimal or hexadecimal error codes of the error messages you want to suppress.

`no-disp-warnings` suppresses errors for which formatting can continue, but it is invalid for errors that stop formatting.

Section 4.6 Barcodes

When you purchase the barcode generator option, you can generate barcodes of various types. Barcodes can also be combined with other extensions.

Different barcode types have different parameters. See the Manual for details on the types and their parameters. Here are some examples of what you can and cannot do by changing parameters and option settings.

GS1-128/EAN-128 identifier

You can register a new application identifier (AI) or change a registered AI. See the manual for the list of registered AI.

Option Element	`<GS1-128>`
Property	`AI`, `format`
Parent	`<formatter-settings>`

Option Property	`AI`
Value	*<string>*
Initial	*empty*
Applies to	`<GS1-128>`

Option Property	`format`
Value	*<string>*
Initial	*empty*
Applies to	`<GS1-128>`

The `<GS1-128>` option element registers a new application identifier, where `AI` specifies the identifier and format specifies its `format`.

Linear barcode height

Consider changing the line width and height within the range that the scanner can read the linear barcode. Depending on the desired shape, the problem can be solved by adjusting the parameters passed when generating the barcode without deforming the block. Specify the length, width and margin parameters as values with units.

```
<!-- Change only the height of linear barcode bar-->
<fo:block>
  <fo:external-graphic
    src="data:application/vnd.ah-barcode;
    type=EAN; h=5mm; text=auto,4901234567894">
</fo:block>
```

Figure 4.6-1 Linear barcode height

Linear barcode font

Option Property	`barcode-text-font`
Value	[*<family-name>* | *<generic-family>*]#
Initial	`OCRB, monospace`
Applies to	`<font-settings>`

Some linear barcodes display text. There are two ways to change the font used with these. One is to change the `barcode-text-font` option, and the other is to not generate text in the barcode and instead overlay another text. The former is safe and easy to do. The latter allows you to specify the distance to the barcode and rotate the text independently of the barcode, but complex control will be required.

The `barcode-text-font` option specifies the font used for the barcode content that is displayed with a linear barcode, except when the text parameter is `none`.

Rotated barcode

Changing the orientation of its parent block changes the orientation of the generated barcode, just like for figures and tables. Because the text of the barcode content should not be rotated, the text is formatted as a separate block [Figure 4.6-2] (p.376).

```
<!-- axf:transform -->
<fo:block
  axf:transform="translateY(5mm) rotate(90)">
  <fo:external-graphic
    src="data:application/vnd.ah-barcode;
```

```
    type=EAN; text=none, 4901234567894" />
</fo:block>
<fo:block text-align="center"
  axf:transform="translateY(10mm)"
  font-family="OCRB, monospace" >
    4901234567894
  </fo:block>
```

This example uses `<axf:transform>`, but it is also possible to change the orientation by using `<fo:block-container>`.

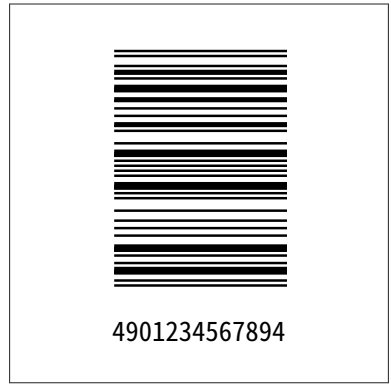


Figure 4.6-2 Barcode text in separate block

List of extended specifications in this book

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Antenna House, Inc.

“Mastering Antenna House XSL Formatter Extensions” ver 1.2 was produced by translating the Japanese version (Japanese: “Antenna House XSL Formatter 拡張仕様使いこなしガイド”) ver 1.1 and modifying the contents partially.

Mastering Antenna House XSL Formatter Extensions

May 18, 2022 1.02 Edition
May 18, 2022 Modified

Author Antenna House, Inc.
Publisher Antenna House, Inc.
Web https://www.antennahouse.com