Practical Formatting Using XSL-FO (Extensible Stylesheet Language Formatting Objects)

Crane Softwrights Ltd.
http://www.CraneSoftwrights.com

+//ISBN 1-894049::CSL::Courses::PFUX//DOCUMENT Practical Formatting Using XSL-FO 2004-02-12 03:30UTC//EN
Fifth Edition - 2004-02-12
ISBN 1-894049-13-6

Practical Formatting Using XSL-FO (Extensible Stylesheet Language Formatting Objects)

Crane Softwrights Ltd.
http://www.CraneSoftwrights.com

Fifth Edition - 2004-02-12 - ISBN 1-894049-13-6

Practical Formatting Using XSL-FO (Prelude) (cont.)

Preface

The main content of this book is in an unconventional style primarily in bulleted form

- derivations of the book are used for instructor-led training, requiring the succinct presentation
  - note the exercises included in instructor-led training sessions are not included in the book
- derivations of the book can be licensed and branded for customer use in delivering training
- the objective of this style is to convey the essence and details desired in a compact, easily perused form, thereby reducing the search for key words and phrases in lengthy paragraphs
- each chapter of the book corresponds to a module of the training
- each page of the book corresponds to a frame presented in the training
- a summary of subsections and their pages is at the back of the book

Much of the content is hyperlinked both internally and externally to the book in the 1-up full-page sized electronic renditions:

- note when using the Acrobat Reader for navigation, the history "back" keystroke sequence is "Ctrl-Left"
- page references, e.g.: Chapter 3 Basic concepts of XSL-FO (page 39)
  - the back-of the book index is hyperlinked to the body of the book
  - the letter references at the bottom of each page are hyperlinked to the index
- construct references are typeset with conventions
  - formatting objects are in monospaced text in brackets, e.g.: `<basic-link>`
  - properties are followed by "=", e.g.: `baseline-shift=`
  - data types are in proportional text in brackets, e.g.: <angle>
- references to sections of the Recommendation are in parentheses, e.g.: (7.13.3)
- external references are in monospaced text, e.g.: `http://www.w3.org/TR/2001/REC-xsl-20011015/xslspec.html`
- chapter references in book summary
- section references in chapter summary
- subsection references in table of contents at the back of the book
- no hyperlinks are present in the cut, stacked, half-page, or 2-up renditions of the material

Practical Formatting Using XSL-FO

Series: Practical Formatting Using XSL-FO

Reference: PFUX

Pre-requisites:

- knowledge of XML syntax

Outcomes:

- exposure to example scripts
- exposure to basic terminology
- exposure to every formatting object

Fifth Edition - 2004-02-12 - ISBN 1-894049-13-6

Paginating structured information

Introduction - Practical Formatting Using XSL-FO

This book is oriented to the XSL-FO stylesheet writer, not the XSL-FO processor implementer

- certain behaviors important to an implementer are not included
- objective to help a stylesheet writer understand the language facilities needed to solve their problem
    - a language reference arranged thematically to assist comprehension
    - a different arrangement than found in the Recommendation itself

First two chapters are introductory in nature

- overview of context of XSL-FO amongst other members of the XML family of Recommendations
- basic flow diagrams illustrate use of XSL-FO
- basic terminology and concepts are defined and explained

Third chapter covers the basics of the area model and page model

- an understanding of the conceptual rendering areas being created by the stylesheet writer
- important to understand these models in order to apply the language features

Fourth through tenth chapters address XSL-FO vocabulary

- all objects are described in detail and their properties are summarized
- significant or important properties are highlighted and described

Paginating structured information (cont.)

Introduction (cont.)

First annex overviews issues of XSLT when working with XSL-FO

- dependencies on XSLT by XSL-FO
- issues to remember when writing XSLT

Second annex covers XSL-FO expressions

- rules when writing expressions
- issues to remember

Third and fourth annexes include object and property summaries derived from the Recommendation

- groupings of objects and properties
- alphabetical lists of objects and properties
- print-oriented summary of all productions

Last annex addresses questions regarding tools

- lists of questions for processor implementers when assessing tool capabilities
- observations and sample incantations working with XSL-FO and XSLT tools

External ZIP file included with the purchase of the book

- all of the complete scripts utilized in the documentation as stand-alone files ready for analysis and/or modification
- sample invocation scripts for Windows environments

Fifth Edition - 2004-02-12 - ISBN 1-894049-13-6

Chapter 1 - Introducing XSL-FO

- Introduction - Contrasting browsed vs. paginated presentations

Contrasting browsed vs. paginated presentations

Consider two media for presenting information to users

- presenting information in a browser window is fundamentally different than presenting the same information in printed form
- browser screens are dynamic
    - navigating around information is the responsibility of the browser
        - in response to user actions
    - the page dimensions can be changed by the reader of the information
        - the width can be modified after the information is presented, requiring the information to be reflowed in the new dimensions
        - the length of the screen is essentially infinite and grows to accommodate the amount of information on the page
- printed pages are static
    - navigating around information is the responsibility of the reader
        - based on cues given in the printed text
    - the page dimensions are fixed by the presenter of the information

Maintaining information separately for two presentations is troublesome

- perhaps the HTML presentation is authored separately from the printed presentation
    - since the navigation tools are different, each authoring tool gives the author a different user interface and authoring options
- the lack of a common source of content introduces maintenance challenges
    - problems when ensuring changes in one are reflected in the other

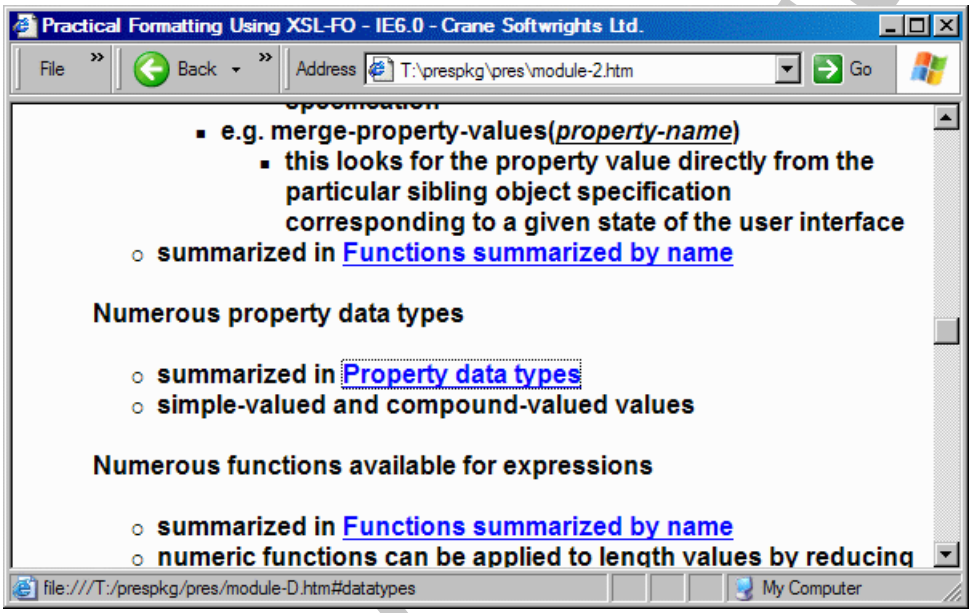
Audiences for browser windows can be different than those for printed information

- a whole constituency of users does not accept reading information from a computer screen
    - will only read information from printed pages
- those who do read lengthy documents from browser windows periodically need a printed rendition
    - the navigation tools on the screen are not translated to the paper form in a browser print request

Contrasting browsed vs. paginated presentations (cont.)

Consider a fragment of training content presented in a browser screen

- hyperlinks are recognized by the underscored text in blue
  - in a black and white print this may be confused with the underscored text at the top of the window that isn't a hyperlink
- a hyperlink with focus exposes the target of the hyperlink in the status line of the window
  - requires interaction with the browser to check each of the possible target locations
- traversing the hyperlink involves interacting with the user agent



Contrasting browsed vs. paginated presentations (cont.)

Consider the training information being authored in HTML for the web presentation:

```
01 <p>Numerous property data types</p>
02 <ul>
03   <li>summarized in <a href="module-D.htm#datatypes">Property
04 data types</a></li>
05   <li>simple-valued and compound-valued values</li>
06 </ul>
07 <p>Numerous functions available for expressions</p>
08 <ul>
09 ...
10 <h2><a name="datatypes">Property data types</a></h2>
```

Of note:

- lines 3-4 show the markup for the highlighted hyperlink
  - the title of the referenced item in line 10 is copied as the "clickable text" of the canvas
  - changing the title requires changing all references to the title
- the entire web page could span dozens of printed pages
  - the hyperlink information is lost
    - no indication of where a target of a hyperlink is found
  - no tools for navigating the collection of pages

Authoring the information in a word processor would require similar duplication of information

- changes to a title requiring changes to all references to the title
- adding of page numbers and page number citations
  - necessitating distinct content from the web-based content

Contrasting browsed vs. paginated presentations (cont.)

Consider an XML representation of the same training material

- we can use any vocabulary of element types and attributes to represent the concepts of training information
- the hyperlink to another location can be an empty element
    - all that is needed is to know what is being referenced not how it is presented

```
<course>
    <title>Practical Formatting Using XSL-FO</title>

<module id="basic">
  <title>Basic concepts of XSL-FO</title>
  <lesson id="vocab">
    <title>Formatting object XML vocabulary</title>
    <frame id="propexp">

      <title>Property value expressions</title>
...
  <point>summarized in <ref idref="funcname"/></point>
</points>
<para>Numerous property data types</para>
<points>
  <point>summarized in <ref idref="datatypes"/></point>
  <point>simple-valued and compound-valued values</point>
</points>
<para>Numerous functions available for expressions</para>
<points>
  <point>summarized in <ref idref="funcname"/></point>
...
    </frame>
  </lesson>
</module>
...
  <frame id="funcname">
    <title>Functions summarized by name</title>
...
  </frame>
...
  <frame id="datatypes">

    <title>Property data types</title>
...
  </frame>
...
</course>
```


Contrasting browsed vs. paginated presentations (cont.)

Of note:

- the same hyperlink is an empty XML element on line 15
    - attribute points to the frame being referenced
- transformation can obtain the title of the referenced section at presentation time
    - only one place to maintain the title
    - guarantees consistency in presentation during maintenance

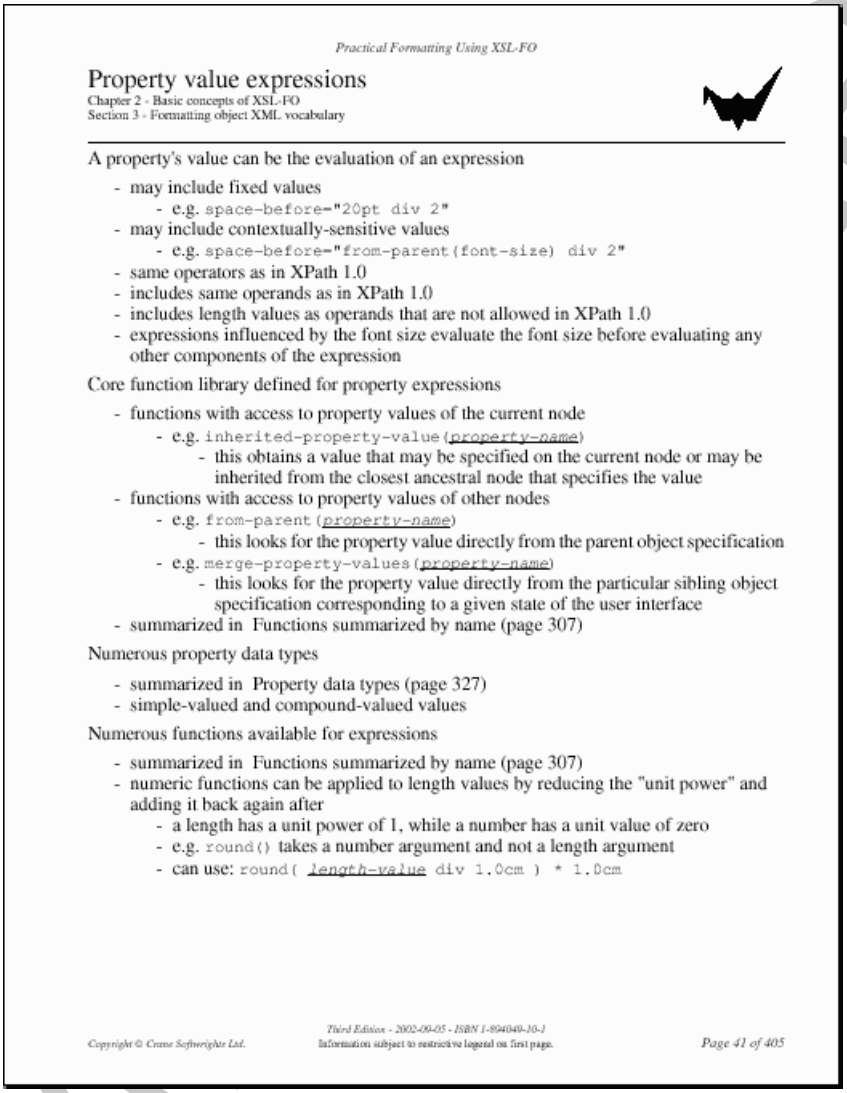
The Extensible Stylesheet Language Transformations (XSLT) is used for presentation and transformation

- rearranges instances of XML information into instances of other vocabularies
- e.g. an XSLT tool reads our XML instances and produces an instance of the HTML vocabulary
    - the stylesheet dictates how the HTML is created with the hyperlink reference for traversal and the clickable content for the canvas

Contrasting browsed vs. paginated presentations (cont.)

Consider the same training information presented in the printed form

- basic presentation of the material is similar to that of the browser presentation
- content is paginated over a number of fixed-size pages
    - navigation tools are different for the collection of pages



Practical Formatting Using XSL-FO

Property value expressions

Chapter 2 - Basic concepts of XSL-FO
Section 3 - Formatting object XML vocabulary

A property's value can be the evaluation of an expression

- may include fixed values
    - e.g. `space-before="20pt div 2"`
- may include contextually-sensitive values
    - e.g. `space-before="from-parent(font-size) div 2"`
- same operators as in XPath 1.0
- includes same operands as in XPath 1.0
- includes length values as operands that are not allowed in XPath 1.0
- expressions influenced by the font size evaluate the font size before evaluating any other components of the expression

Core function library defined for property expressions

- functions with access to property values of the current node
    - e.g. `inherited-property-value(property-name)`
        - this obtains a value that may be specified on the current node or may be inherited from the closest ancestral node that specifies the value
- functions with access to property values of other nodes
    - e.g. `from-parent(property-name)`
        - this looks for the property value directly from the parent object specification
    - e.g. `merge-property-values(property-name)`
        - this looks for the property value directly from the particular sibling object specification corresponding to a given state of the user interface
- summarized in Functions summarized by name (page 307)

Numerous property data types

- summarized in Property data types (page 327)
- simple-valued and compound-valued values

Numerous functions available for expressions

- summarized in Functions summarized by name (page 307)
- numeric functions can be applied to length values by reducing the "unit power" and adding it back again after
    - a length has a unit power of 1, while a number has a unit value of zero
    - e.g. `round()` takes a number argument and not a length argument
    - can use: `round( length-value div 1.0cm ) * 1.0cm`

Third Edition - 2002-09-05 - ISBN 1-894049-10-1
Copyright © Crane Softwrights Ltd. Information subject to restrictive legend on first page. Page 41 of 405

Contrasting browsed vs. paginated presentations (cont.)

Of note:

- the hyperlinks are rendered with the page number of the target of each reference
- the page number is shown at the bottom right of the page for navigation
- the module and lesson information is shown at the top left for context

The Extensible Stylesheet Language Formatting Objects (XSL-FO) XML vocabulary is used to express pagination semantics

- architecturally the same as when we use HTML to express browser semantics
- we learn the XSL-FO vocabulary representing the XSL-FO semantics
    - e.g. how to express a hyperlink as a page number citation
    - same as we needed to know what HTML can do in a browser
        - e.g. how to express a hyperlink as an anchor
- instances of our XML vocabularies are transformed into instances of the XSL-FO vocabulary
    - e.g. transforming our `<ref/>` elements into `<page-number-citation/>` elements
    - same process as is needed to go from XML to HTML
        - e.g. transforming our `<ref/>` elements into `<a>` elements
- an XSL-FO engine interprets the XSL-FO semantics to produce the printable results
    - e.g. an XSL-FO tool reads our XSL-FO and produces a PDF file
    - same process as is needed by a web browser for HTML
        - e.g. a browser user agent reads our HTML and renders a screen window
- only difference is the selection of presentation semantics appropriate for the desired target medium

Chapter 2 - The context of XSL-FO

- Introduction - Overview
- Section 1 - The XML family of Recommendations
- Section 2 - Examples

Outcomes:

- introduction to objectives and purpose
- awareness of available documentation
- an awareness of available documentation and a small subset of publicly available resources
- exposure to example scripts

Overview

Chapter 2 - The context of XSL-FO

In this chapter we look at four technologies. In later chapters we will both set the stage for and delve into the semantics of paginated formatting available through detailed examples, definitions of terminology, and an examination of each of the formatting objects.

Extensible Markup Language (XML)

- hierarchically describes an instance of information
  - using embedded markup according to rules specified in the Recommendation
  - according to a vocabulary (a set of element types each with a name, a structure and optionally some attributes) described by the user
- optionally specifies a mechanism for the formal definition of a vocabulary
  - controls the instantiation of new information
  - validates existing information

Document Style Semantics and Specification Language (DSSSL)

- an internationally standardized collection of style semantics
- a specification language for the transformation of structured information and the application of the standardized internationalized style semantics for paginating structured information

Cascading Stylesheets (CSS)

- a set of formatting properties to attach to structured documents
  - defines a cascade of property sources from the markup to the embedded stylesheet to the external stylesheet to the default presentation semantics of the processor
- browsers supporting CSS can render the document structure of HTML documents or XML documents according to the properties

Extensible Stylesheet Language Family (XSLT/XSL/XSL-FO)

- XSL Transformations (XSLT)
  - specifies the transformation of XML-encoded information into a hierarchy using the same or a different document model *primarily for the kinds of transformations for use with XSL*
- XSL (Formatting Semantics, a.k.a. XSL-FO)
  - specifies the vocabulary and semantics of the formatting of information for paginated presentation
  - colloquially referred to at times as XSL Formatting Objects

- `http://www.w3.org/TR/REC-xml`

A Recommendation fulfilling two objectives for information representation

- capturing information in a hierarchical form in markup according to basic XML-defined constraints
    - creating well-formed documents of elements, attributes and other constructs
- restricting and/or validating hierarchical information in XML to arbitrary user-specified constraints
    - defining a model or grammar for the structure and content of a document
        - collection of available element types and their respective attributes
            - inherent relationships between information in the hierarchy
        - vocabulary can be expressed formally in XML 1.0 as a Document Type Definition (DTD)

Nothing in XML is related to presentation or rendition

- no inferred semantics for the display or formatting of information when using XML
- `xml:space` is used only for the significance of the white space in the document

The vocabulary of elements and attributes used in an instance can be validated

- at a grammar level by a declarative document model
    - structural validation
        - the nesting and order of elements and their use of attributes
    - lexical validation and integrity
        - certain aspects of content and the allowable string values of attributes
    - a distinct process separate from the applications acting on the information
        - analysis against the DTD
        - analysis using other validation mechanisms (e.g. XML Schema, RELAX-NG Schema, Schematron, etc.)
- at a semantic level by the application processing the information
    - the semantics of information is not defined by the grammar or structure of the information
    - information "means" exactly what any application processing the information wants it to mean
    - the application analyzes the structure and content of the information for appropriateness to its purpose
        - can test conditions or constraints that cannot be expressed in a formal document model syntax
        - can algorithmically determine validity to support requirements not easily expressed declaratively

Extensible Markup Language (XML) (cont.)

XML vocabularies can be translated to an application's specific vocabulary

- our XML vocabularies should be designed according to the business processes acting on the information, not around the appearance
    - flexibility to have many different appearances for the same information
- a presentation application vocabulary could just be attribute values added to our own vocabularies without changing element names (e.g. Cascading Stylesheets (CSS))
    - we can add properties that are recognized by a browser
    - our information is then rendered visually or aurally according to the properties
- a presentation application vocabulary could be elements and attributes expressing the semantics of browsing information (e.g. Hypertext Markup Language (HTML) or Scalable Vector Graphics (SVG))
    - browsers have adopted common presentation semantics for HTML constructs
    - if the presentation semantics are sufficient, we need only use HTML
    - if the presentation semantics are insufficient, we can use both HTML and CSS
        - CSS-aware browsers would present the information as desired
        - non-CSS-aware browsers would at least present the information using the presentation semantics of HTML

Namespaces distinguish constructs in information from different vocabularies

- a single instance can contain information from different document models
- the recognition of element types by an application is through combination of namespace URI string and un-prefixed element type name
    - the prefix used in the instance is irrelevant to the application

An application can only act on the vocabulary it recognizes and must have a behavior for vocabulary it doesn't recognize

- a web browser understands the HTML and CSS vocabularies
    - displays understood constructs accordingly
    - ignores unrecognized constructs while passing content through to the canvas
- an e-commerce application understands the vocabulary designed to trigger behavior
    - performs the functionality accordingly
    - could exit with an error or warning for unrecognized constructs
- a namespace-aware application can recognize constructs from different vocabularies
    - using namespaces gives us better labels for the elements and attributes in our XML information than simple names
    - the semantics of our information is assumed by the application we use to process our information through the labels we use
        - using better labels results in more successful application processing

XML information links

Links to useful information

- `http://www.xml.com/axml/axml.html` - annotated version
- `http://xml.coverpages.org/xml.html` - Robin Cover's famous resource collection
- `http://xml.coverpages.org/xll.html` - Extensible Linking Language
- `http://www.ucc.ie/xml/` - Peter Flynn FAQ
- `http://www.xmlbooks.com/` - a summary of available printed books
- `http://www.CraneSoftwrights.com/links/trn-20040212.htm` - training material
- `http://www.CraneSoftwrights.com/resources` - free resources
- `http://XMLGuild.info` - consulting and training expertise
- `http://xml.coverpages.org/elementsAndAttrs.html` - a summary of opinions

Related initiatives

- `http://www.w3.org/TR/xmlschema-0/` - W3C XML Schema
- `http://www.oasis-open.org/committees/relax-ng/` - RELAX NG (based on RELAX and TREX)
- `http://www.ascc.net/xml/schematron/` - Schematron
- `http://www.w3.org/TR/DOM-Level-2/` - Document Object Model Level 2
- `http://www.saxproject.org` - Simple API for XML

Examples of processors

- `http://www.jclark.com/xml/xp/` - XP (Java)
- `http://xml.apache.org/` - Xerces (Java, C++ and Perl Interfaces)
- `http://www.xmlsoft.org` - XML for Gnome
- `http://www.textuality.com/Lark/` - Lark (Java)
- `http://www.ltg.ed.ac.uk/software/xml/` - LT XML
- `http://msdn.microsoft.com/downloads/webtechnology/xml/msxml.asp` - Microsoft
- `http://www.a-dos.com` - XML processor and associated tools
- `http://www.xmlsoftware.com/` - list of tools

The above list is just some of the early or interesting processors of the very many that are available commercially and publicly.

- ISO/IEC-10179:1996
  - `http://www.y12.doe.gov/sgml/wg8/dsssl/readme.htm`

Transforming and formatting structured information

- distinguishes the separate behaviors required to style structured information
- a transformation language to rearrange structured information
- pagination semantics for presenting information in fixed-sized folios
- includes an extension mechanism for arbitrary formatting semantics

A programming language for transforming structured information

- specifies relationships between multiple input documents in the transformation to zero or more output documents
- uses side-effect-free dialect of Scheme (itself a derivative of LISP) for the expression language

A standardized set of formatting semantics for paginated output

- specifies the intent of the result of a formatting process
- does not specify the rendering process
- no bias in constructs to any particular writing direction

A framework for implementation-defined sets of formatting semantics

- there exists a set of semantics for formatting into SGML/XML markup, thus implementing a transformation function through the use of formatting facilities

Custody of ISO/IEC JTC 1/SC 34/WG 2

- formerly ISO/IEC JTC 1/WG 4
- formerly ISO/IEC JTC 1/SC 18/WG 8

- `http://www.w3.org/TR/REC-CSS1`
- `http://www.w3.org/TR/REC-CSS2`

Formatting property assignment for web documents (HTML and XML)

- no document manipulation capabilities
- width and length of presentation are not fixed
    - can be changed dynamically by the reader of the information
- developed to address incompatible vendor extensions for formatting that were being added to browsers

Ornamentation of the document tree

- attaching stylistic information to nodes
- simple prefixing and suffixing of nodes with text
- control of white space around information
- overlapping and transparent rectangular regions
- significant use of inheritance of formatting properties from ancestral tree locations
    - defines the "cascade" of application of inheritable formatting properties

Multiple media type support

- character display presentation properties
- tabular presentation properties
- aural presentation properties for visually impaired browsing
    - disabled users
    - mobile users

Doesn't *(shouldn't)* interfere with legacy browsers not supporting CSS

- values expressed in attributes and document metadata
- expression can be external to the document itself
    - required to be external for XML files not using namespaces
- can be introduced to namespace-aware XML documents through the HTML vocabulary

Working group is producing a common formatting model for web documents

- all W3C Recommendations needing presentation properties should use the CSS semantics and associated property names where applicable

Styling structured information

Styling is *transforming* and *formatting* information

- the application of two processes to information to create a rendered result
- the ordering of information for creation isn't necessarily (or shouldn't be constrained to) the ordering of information for presentation or other downstream processes
    - it is a common (though misdirected) first step for people working with these technologies to focus on presentation
    - the ordering should be based on business rules and inherent information properties, not on artificial presentation requirements
    - downstream orderings can be derived from constraints imposed upstream in the process
    - information created richly upstream can be manipulated into less-richly distinguished information downstream, but not easily the other way around
    - exception when the business rules are presentation or appearance oriented (e.g. book publishing)
- the need to present information in more than one ordering requires transformation
- the need to present information in more than one appearance requires formatting

W3C XSL Working Group

- chartered to define a style specification language that covers at least the formatting functionality of both CSS and DSSSL
- not intended to replace CSS, but to provide functionality beyond that defined by CSS
    - e.g. add element reordering and pagination semantics

Two W3C Recommendations

- designed to work together to fulfill these two objectives
- XSL Transformations (XSLT)
    - transforming information obtained from a source into a particular reorganization of that information to be used as a result
- Extensible Stylesheet Language (XSL/XSL-FO)
    - specifying and interpreting formatting semantics for the rendering of paginated information
    - the acronym XSL-FO is unofficial but in wide use, including at the W3C, for just the formatting objects, properties and property values
    - XSL normatively includes XSLT by reference in chapter 2
        - XSLT has specific features designed to be used for XSL-FO

XSLT and XSL-FO are endorsed by members of WSSSL

- an association of researchers and developers passionate about markup technologies

- `http://www.w3.org/TR/xslt`

Transformation by example

- a vocabulary for specifying templates of the result that are filled-in with information from the source
  - the stylesheet includes examples of each of the components of the result
  - the stylesheet writer declares how the XSLT processor builds the result from the supplied examples
- the primary memory management and manipulation (node traversal and node creation) is handled by the XSLT processor using declarative constructs, in contrast to a transformation programming language or interface (e.g. the DOM - Document Object Model) where the programmer is responsible for handling low-level manipulation using imperative constructs
- includes constructs to iterate over structures and information found in the source
- the information being transformed can be traversed in different ways any number of times required for the desired result
- straightforward problems are solved in straightforward ways without needing to know programming
  - useful, commonly-required facilities are implemented by the processor and can be triggered by the stylesheet
  - the language is Turing complete, thus arbitrarily complex algorithms can be implemented (though not necessarily in a pretty fashion)
- includes constructs to manage stylesheets by sharing components

Not intended for general purpose XML transformations

- designed for downstream-processing transformations suited for use with XSL formatting vocabulary
  - includes facilities for working with the XSL vocabulary easily
- still powerful enough for *most* downstream-processing transformation needs
  - an XSLT stylesheet can be (and is) called a transformation script
  - absolutely general purpose when the output from XSLT is going to be input to an XML processor
- does not include certain features appropriate for syntax-level general purpose transformations
  - unsuitable for original markup syntax preservation requirements

- `http://www.w3.org/TR/xsl`

Paginated flow and formatting semantics vocabulary

- capturing agreed-upon formatting semantics for rendering information in a paginated form on different types of media
- XSLT is normatively referenced as an integral component of XSL as a language to transform an instance of an arbitrary vocabulary into the XSL-FO XML vocabulary
- XSL-FO can be regarded simply as a "pagination markup language"
- flow semantics from the DSSSL heritage
    - e.g. headers, footers, page numbers, page number citations, columns, etc.
- formatting semantics from the CSS heritage
    - e.g. visual properties (font, color, etc.) and aural properties (speak, volume, etc.)

Target of transformation

- the stylesheet writer transforms a source document into a hierarchy that uses only the formatting vocabulary in the result tree
- stylesheet is responsible for constructing the result tree that expresses the desired rendering of the information found in the source tree
    - the XML document gets transformed into its appearance
- stylesheet cannot use any user constructs as they would not be recognized by an XSL rendering processor
    - for example, the rendering engine doesn't know what an invoice number or customer number is that may be represented in the source XML
    - the rendering engine does know what a block of text is and what properties of the block can be manipulated for appearance's sake
    - the stylesheet transforms the invoice number and customer number into two blocks of text with specified spacing , font metrics, and area geometry

Device-independent formatting constructs

- the XSL-FO vocabulary describes two media interpretations for objects and properties:
    - visual media
    - aural media
    - a further distinction is also made at times for interactive media
- the results of applying a single stylesheet can be rendered on different types of rendering devices, e.g.: print, display, audio, etc.
- may still be appropriate to have separate stylesheets for dissimilar media
    - device independence allows the information to be rendered on different media, but a given rendering may not be conducive to consumption

Styling semantics and vocabularies

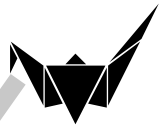
XSLT and XSL-FO processors implement styling semantics

- recognize standardized constructs by their labels in the two respective namespaces
    - elements and their attributes represent semantic concepts
        - XSLT instructions and their controls
        - XSL-FO formatting objects and their properties
- recognize extension constructs by their labels in namespaces recognized by the processor
- accommodate constructs by their labels in unrecognized namespaces

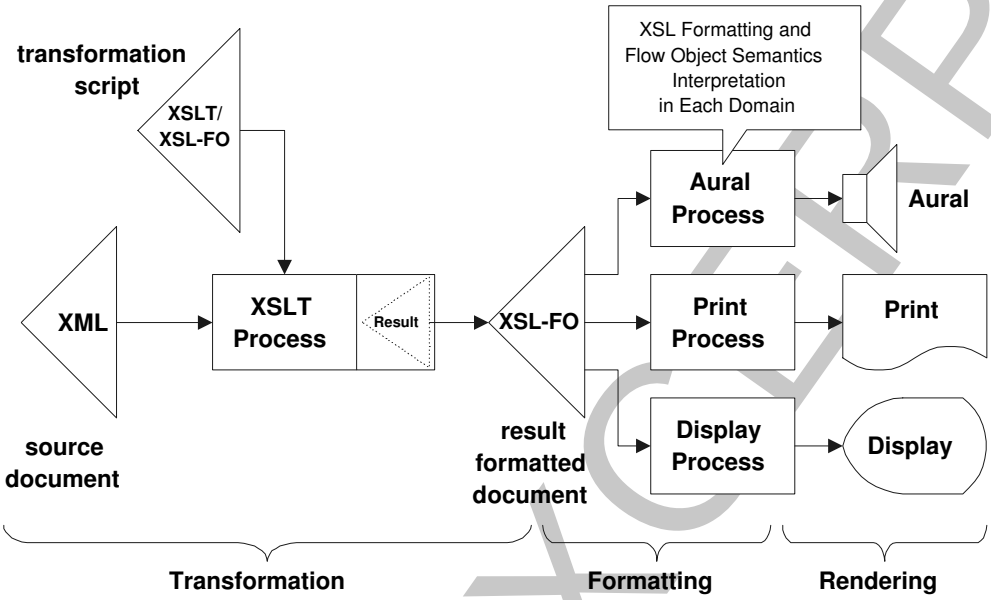
XSLT and XSL-FO document type definitions are described using prose

- there are no standardized XML 1.0 DTD representations of the grammar of the vocabularies
    - DTD semantics and syntax unable to fully express all of the grammatical constraints
- XSLT 1.0 and XSL 1.0 Recommendations describe the document type definitions
- processors do all aspects of validation and interpretation according to the respective document type
- snippets of DTD content model syntax with Kleene operators used in documentation because of the familiarity with the reader
    - "?" for zero or one
    - "*" for zero or more
    - "+" for one or more
- additional constraints not expressible in DTD content model syntax are described in prose

Outboard XSLT and XSL-FO processes



The XSL-FO and foreign object vocabularies can be used in a standalone XML instance, perhaps as the result of an XSLT transformation using an outboard XSLT processor:



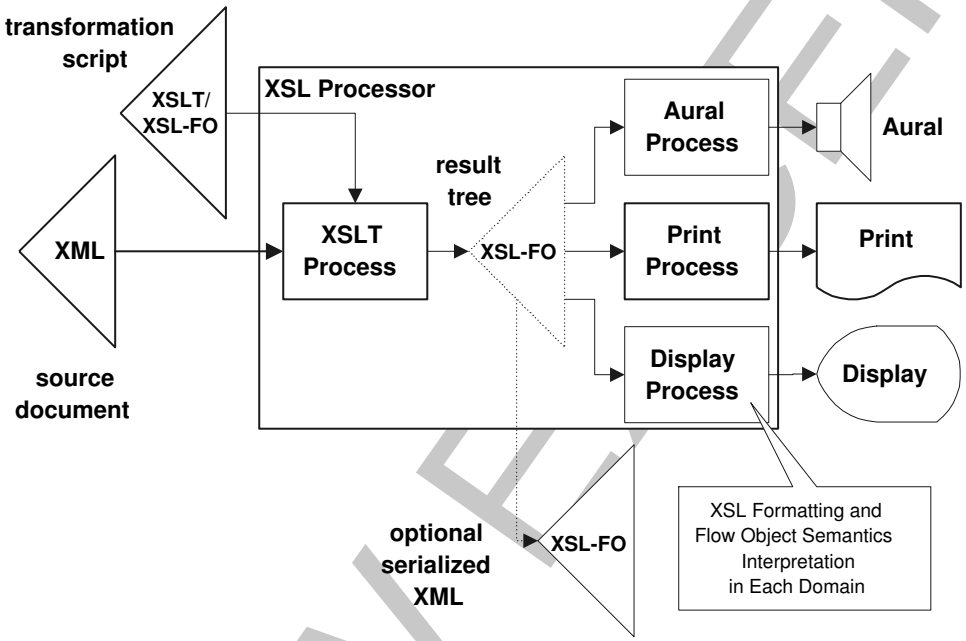
Note the same three distinct phases as when XSLT and XSL-FO processors are combined in a single application:

- transformation creates XSL-FO expressing our intent for formatting the source XML
- XSL-FO process interprets our intent into the information that is to be rendered on the target device
- XSL-FO process effects the rendering to reify the result

Transforming and rendering XML information using XSLT and XSL-FO

When the XSLT result tree is specified to utilize the XSL-FO formatting vocabulary:

- the normative behavior is to interpret the result tree according to the formatting semantics defined in XSL for the XSL-FO formatting vocabulary
- an inboard XSLT processor can effect the transformation to an XSL-FO result tree
- the XSL-FO result tree need not be serialized in XML markup to be conforming to the recommendation
    - useful for diagnostics to evaluate results of transformation

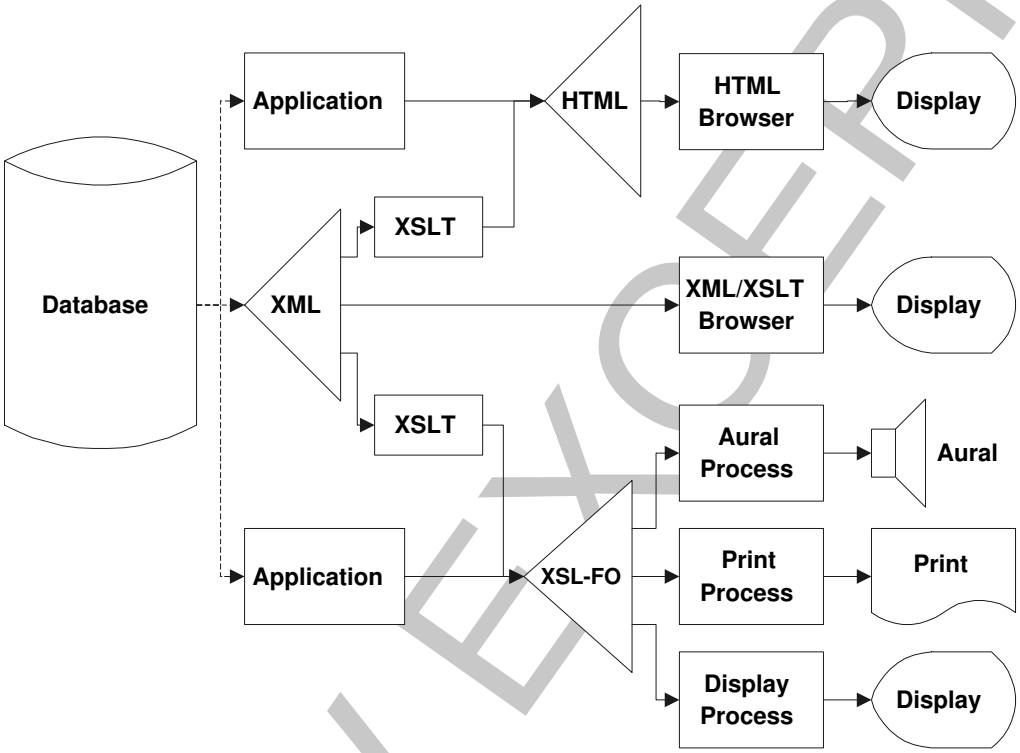


Of note:

- the stylesheet contains only the XSLT transformation vocabulary, the XSL formatting vocabulary, and extension transformation or foreign object vocabularies
- the source XML contains the user's vocabularies
- the result of transformation contains exclusively the XSL formatting vocabulary and any extension formatting vocabularies
    - does not contain any constructs of the source XML vocabulary
    - may or may not be serialized as XML markup (useful for diagnostics)
- the rendering processes implement for each medium the common formatting semantics described by the XSL recommendation
    - for example, space specified before blocks of text can be rendered visually as a vertical gap between left-to-right line-oriented paragraphs or aurally as timed silence before vocalized content

The XSL vocabulary need not be created using XSLT

- volume HTML is often generated directly from applications
- XSLT can be used to transform XML into any vocabulary
- nothing in XSL-FO prevents it from being generated directly from applications



Sole requirement of instance is the use of the XSL vocabulary namespace:

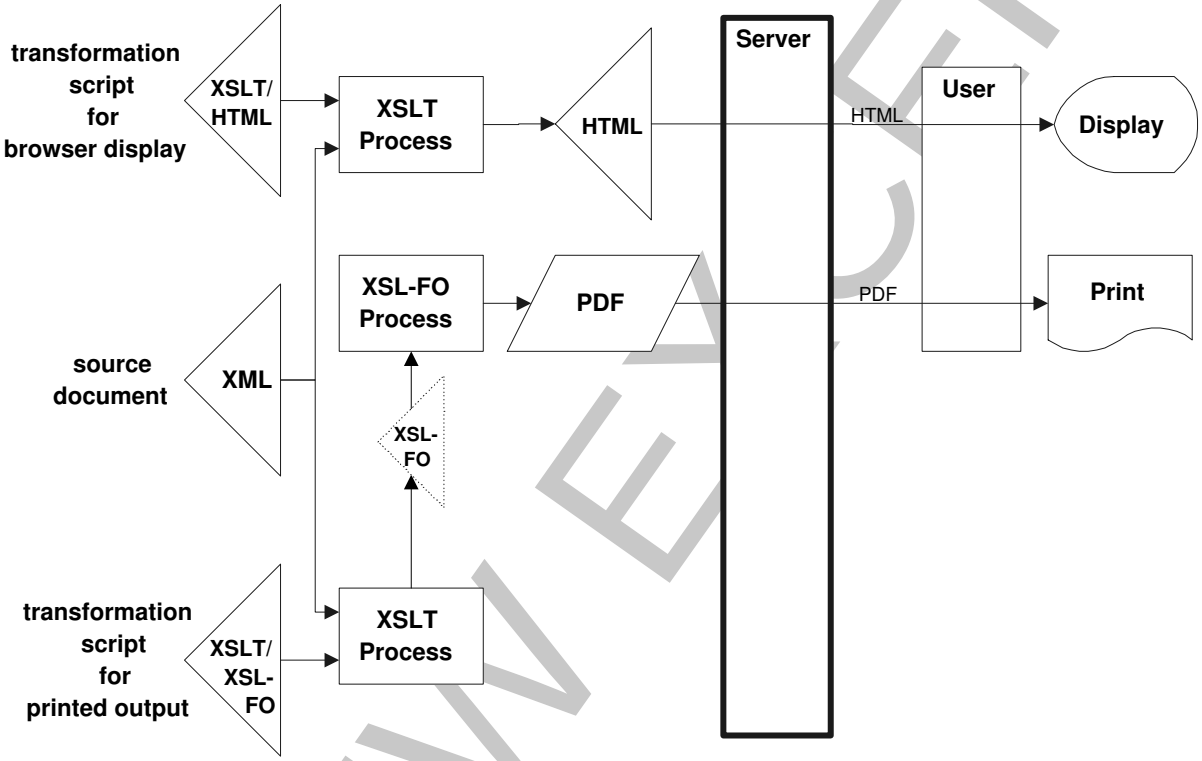
- `http://www.w3.org/1999/XSL/Format`
- can be the default namespace
  - no XSL-FO attribute problems as exhibited with XSLT attributes when using the default namespace for XSLT

Using XSL-FO on a server

Chapter 2 - The context of XSL-FO
Section 1 - The XML family of Recommendations

A typical web-based use of XSL-FO is the server delivery of "printable versions" of information found on a web page

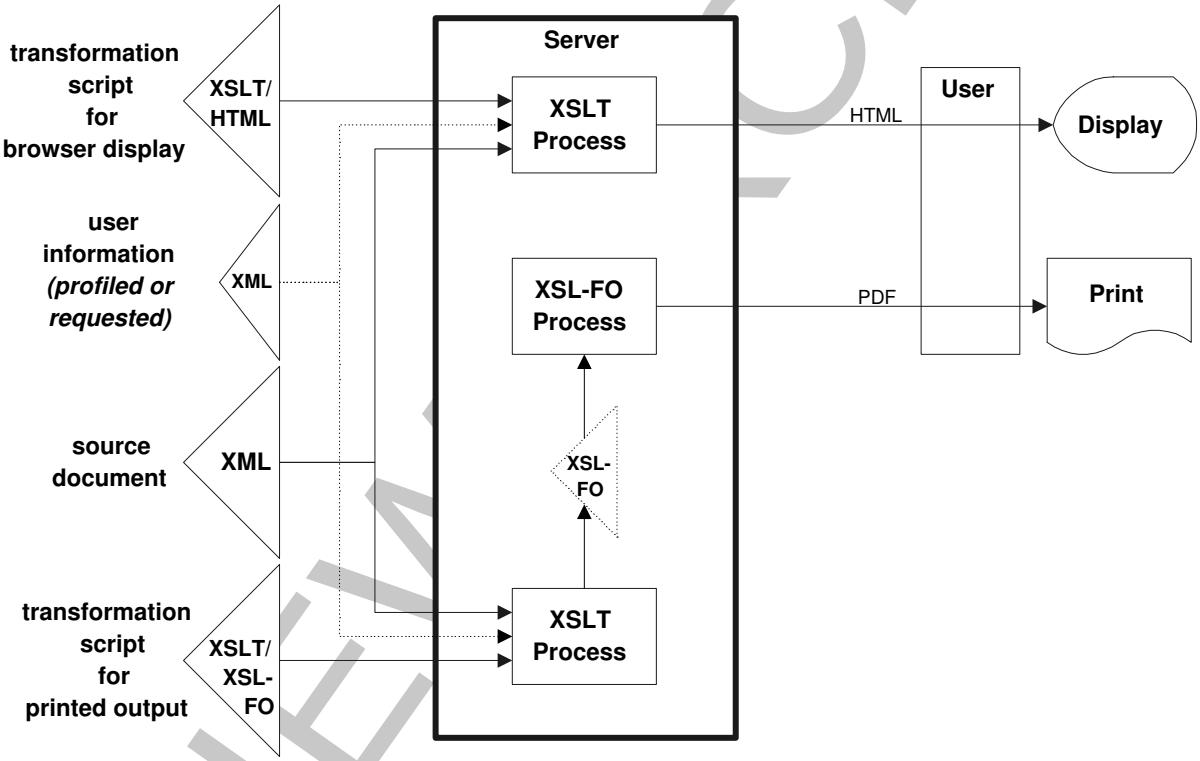
- a transformation from XML to HTML creates the static information for a browser
- a transformation from XML to XSL-FO to PDF creates the static information in printable form
    - user would use the PDF reader on their system to produce the paper through the system printer



Using XSL-FO on a server (cont.)

A dynamic web-based use of XSL-FO is the on-the-fly synthesis and delivery of "printable versions" of information found on a web page

- static information is combined with user information based on the session engaged with the user
  - user information based on a predefined user profile
  - application information based on a dynamic request from the user
- a transformation from XML to HTML is used to deliver the information to a browser
- a transformation from XML to XSL-FO to PDF is used to deliver a paginated form of the information to the user
  - user would use the PDF reader on their system to produce the paper through the system printer



Historical development of the XSL and XSLT Recommendations

Recommendation release history:

- first concept description floated in August 1997 with no official status within the World Wide Web Consortium (W3C)
  - `http://www.w3.org/TR/NOTE-XSL.html`
- the XSL Working Group officially chartered in early 1998
  - `http://www.w3.org/Style/XSL/`
- agreed upon requirements for XSL by the Working Group:
  - `http://www.w3.org/TR/WD-XSLReq`
- the XSL 1.0 Recommendation (XSL-FO) published October 15, 2001
  - `http://www.w3.org/TR/2001/REC-xsl-20011015/`
  - some software only supports Candidate Recommendation:
    - `http://www.w3.org/TR/2000/CR-xsl-20001121/`
- the XSLT/XPath 1.0 Recommendations published November 16, 1999
  - `http://www.w3.org/TR/1999/REC-xslt-19991116`
    - `http://www.w3.org/1999/11/REC-xslt-19991116-errata` - errata
  - `http://www.w3.org/TR/1999/REC-xpath-19991116`
    - `http://www.w3.org/1999/11/REC-xpath-19991116-errata` - errata
- XSLT 1.1 (work abandoned)
  - `http://www.w3.org/TR/2000/WD-xslt11req-20000825` - requirements
  - `http://www.w3.org/TR/2000/WD-xslt11-20001212`
  - no incompatible changes to XSLT 1.0 in XSLT 1.1, only additional functionality
  - too many interactions with plans for XSLT 2.0, so functionality to be folded into XSLT 2.0 release
- XSLT/XPath 2.0 (work in progress)
  - `http://www.w3.org/TR/2001/WD-xslt20req-20010214` - requirements
  - `http://www.w3.org/TR/xpath20/` - current XPath work
    - `http://www.w3.org/TR/query-datamodel/` - shared data model with XQuery
    - `http://www.w3.org/TR/xquery-operators/` - shared operators with XQuery
    - `http://www.w3.org/TR/xquery/` - standalone query work
  - `http://www.w3.org/TR/xslt20/` - current XSLT work
- XSL-FO 1.1 (work in progress)
  - `http://www.w3.org/TR/xsl11/`

XSL information links

Links to useful information

- `http://xml.coverpages.org/xsl.html` - Robin Cover
- `http://www.mulberrytech.com/xsl/xsl-list/` - mail list
- `http://www.dpawson.co.uk` - an XSL/XSLT FAQ
- `http://www.zvon.org/HTMLonly/XSLTutorial/Books/Book1/index.html` - numerous example XSLT scripts and fragments
- `http://www.nag.co.uk/projects/OpenMath/corecd/` - OpenMath project work by David Carlisle
- `http://www.CraneSoftwrights.com/links/trn-20040212.htm` - comprehensive XSLT/XPath and XSL-FO training material
- `http://www.CraneSoftwrights.com/resources`- free XSLT and XSL-FO resources
- `http://incrementaldevelopment.com/xsltrick/` - "Stupid XSLT Tricks"
- `http://www.xmlsoftware.com/` - list of tools
- `http://xml.coverpages.org/xslSoftware.html` - list of tools
- `http://www.exslt.org/` - community effort for XSLT extensions
- `http://exslfo.sf.net` - community effort for XSL-FO extensions
- `http://foa.sourceforge.net/` - open source FO GUI authoring tool
- `http://www.xslfast.com/` - commercial FO GUI authoring tool
- `http://www.inventivedesigners.com/` - commercial FO GUI authoring tool
- `http://www.abisource.com/` - word processing with "Save As..." for XSL-FO
- `http://www.AntennaHouse.com/XSLsample/XSLsample.htm` - paginating XHTML
- `ISBN 1-56609-159-4` - "The Non-Designer's Design Book", Robin Williams, Peachpit Press, Inc., 1994
- `ISBN 0-8230-2121-1/0-8230-2122-X` - "Graphic design for the electronic age; The manual for traditional and desktop publishing", Jan V. White, Xerox Press, 1988 (out of print but worthwhile to search for as a used book)

XSL information links (cont.)

Examples of XSLT processors

- `http://www.jclark.com/xml/xt.html` - James Clark
- `http://saxon.sourceforge.net` - Mike Kay
- `http://msdn.microsoft.com/downloads/webtechnology/xml/msxml.asp` - updated web release of XML/XSLT processor for Internet Explorer 5 (IE6 follows the W3C specifications)
  - `http://www.netcrucible.com/xslt/msxml-faq.htm` - useful FAQ
- `http://technet.oracle.com/tech/xml/` - Oracle
- `http://xml.apache.org/xalan/index.html` - Apache Project JAVA-based implementation (originally from IBM/Lotus AlphaWorks)
- `http://alphaworks.ibm.com/tech/LotusXSL`- IBM/Lotus AlphaWorks wrapper for Xalan
- `http://www.xmlsoft.org` - XSLT for Gnome
- `http://www.DataPower.com` - XSLT-dedicated hardware
- `http://www.sarvega.com` - XSLT-dedicated hardware
- `http://www.ambrosoft.com/gregor.html` - XSLT compiler
- `http://www.infoteria.com` - iXSLT - commercial implementation
- `http://www.unicorn-enterprises.com/` - Unicorn XSLT Processor
- `http://www.a-dos.com` - XSLT processor and associated tools

The above list is just some of the early or interesting processors of the very many that are available commercially and publicly.

XSL information links (cont.)

Examples of XSL formatting object rendering processors

- `http://www.AntennaHouse.com/` - AntennaHouse Windows-based and multi-platform versions
- `http://www.RenderX.com/` RenderX - direct to PDF
- `http://www.3b2.com` - Advent 3B2 - direct to PDF
- `http://www.ArborText.com` - Epic composition tool
- `http://www.Adobe.com` - Adobe Document Server
- `http://www.xmlpdf.com/ibex` - Ibex C# .NET - direct to PDF
- `http://www.Lunasil.com` - Java/COM - direct to PDF
- `http://xml.apache.org/fop/` - FOP - direct to PDF, PCL, others
- `http://xmlroff.sourceforge.net/` - open source to PDF
- `http://www.hcu.ox.ac.uk/TEI/Software/passivetex` - Passive TeX - TeX to PDF
- `http://www.unicorn-enterprises.com/` - Unicorn UFO - TeX to PDF
- `http://www.alphaworks.ibm.com/tech/xfc` - IBM XFC - direct to PDF
- `http://www.xmlmind.com/foconverter` - Pixware XFC - XSL-FO to RTF
- `http://www.jfor.org/` - XSL-FO to RTF
- `http://www.xsmiles.org/` - XML browser using FOP

The above list is just some of the early processors of what is anticipated to be very many that will be available commercially and publicly.

Hello world example

Consider a simple, but complete, XSL-FO instance `hellofo.fo` for an A4 page report:

```
<?xml version="1.0" encoding="UTF-8"?>
<root xmlns="http://www.w3.org/1999/XSL/Format"
      font-size="16pt">
  <layout-master-set>
    <simple-page-master
            margin-right="15mm" margin-left="15mm"
            margin-bottom="15mm" margin-top="15mm"
            page-width="210mm" page-height="297mm"
            master-name="bookpage">
      <region-body region-name="bookpage-body"
            margin-bottom="5mm" margin-top="5mm" />
    </simple-page-master>
  </layout-master-set>
  <page-sequence master-reference="bookpage">
    <title>Hello world example</title>
    <flow flow-name="bookpage-body">
      <block>Hello XSL-FO!</block>
    </flow>
  </page-sequence>
</root>
```

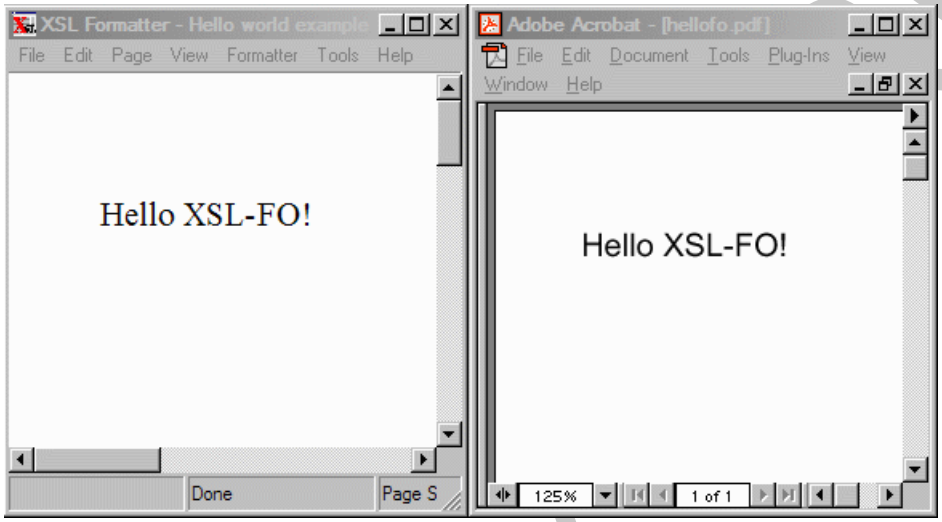
All examples illustrate instances of the XSL-FO vocabulary

- how the instance is created is not material to the semantics of the vocabulary
  - could be hand-authored in a simple text or XML editor
  - could be the result of an XSLT transformation from another XML vocabulary
  - could be output from any application
- the default namespace is used in the examples for brevity and clarity

Hello world example (cont.)

Rendered on the screen in two adjacent windows using conforming XSL-FO processors:

- Antenna House XSL Formatter (an interactive XSL-FO rendering tool)
- Adobe Acrobat (a Portable Document Format (PDF) display tool)
    - PDF created by RenderX XEP (a batch XSL-FO rendering tool)



Note the two renderings are not identical

- XSL-FO only specifies the formatting process, not the rendering process
- the XSL-FO instance may be insufficient in describing the entire formatting intent
- page fidelity is not guaranteed if the instance does not express the entire intent
    - XSLT semantics are extensive, but not necessarily comprehensive for certain nuances of formatting
- the rendering may engage certain property values of its own choosing

No different than two web browsers with different user settings for fonts

- a simple web page not using CSS properties relies on the browser settings
- the HTML constructs represent the intent of what is to be formatted
- absent formatting properties are satisfied using the tool option defaults

A detailed example of flowed content

Chapter 2 - The context of XSL-FO
Section 2 - Examples

Consider a page of content from some instructor-led training material that contains a mixture of a table, a list, a proportionally-spaced paragraph, and mono-spaced paragraphs:

Introduction to XSL-FO

Training material example

Module 2 - The context of XSL-FO
Lesson 2 - Examples

This page's material as an instructor-led handout:

- excerpt of formatting objects created using an XSLT stylesheet and an XSLT stylesheet processor

```
<flow flow-name="pages-body"><table>
 <table-column column-width="( 210mm - 2 * 15mm ) - 2in"/>
 <table-column column-width="1in"/>
 <table-column column-width="1in"/>
 <table-body><table-row><table-cell><block text-align="start">
     <block font-size="19pt">Training material example</block>
     <block font-size="10pt" space-before="10pt">Module
2 - The context of XSL-FO</block>
     <block font-size="10pt">Lesson 2 - Examples</block></block>
  </table-cell>
  <table-cell><block text-align="end"><external-graphic
    src="url(&quot;..\whitesml.bmp&quot;)"/></block></table-cell>
  <table-cell><block text-align="start"><external-graphic
    src="url(&quot;..\cranesml.bmp&quot;)"/></block></table-cell>
 </table-row></table-body></table>
 <block line-height="3px"><leader leader-pattern="rule"
      leader-length="100%" rule-thickness="1pt"/></block>
 <block space-before="6pt" font-size="14pt">
This page's material as an instructor-led handout:</block>
 <list-block provisional-distance-between-starts=".43in"
  provisional-label-separation=".1in" space-before="6pt">
  <list-item relative-align="baseline">
   <list-item-label text-align="end" end-indent="label-end()">
    <block>-</block></list-item-label>
   <list-item-body start-indent="body-start()">
    <block font-size="14pt">excerpt of formatting objects created
using an XSLT stylesheet and an XSLT stylesheet processor</block>
   </list-item-body></list-item></list-block>
 <block space-before="12pt div 2" font-family="Courier"
      linefeed-treatment="preserve" white-space-collapse="false"
      white-space-treatment="preserve" font-size="12pt"><inline
font-size="inherited-property-value(font-size) div 2">01 </inline
>&lt;flow flow-name="pages-body"&gt;&lt;table&gt;
<inline font-size="inherited-property-value(font-size) div 2"
>02 </inline> &lt;table-column column-width...
```

Training material example

Chapter 2 - The context of XSL-FO
Section 2 - Examples

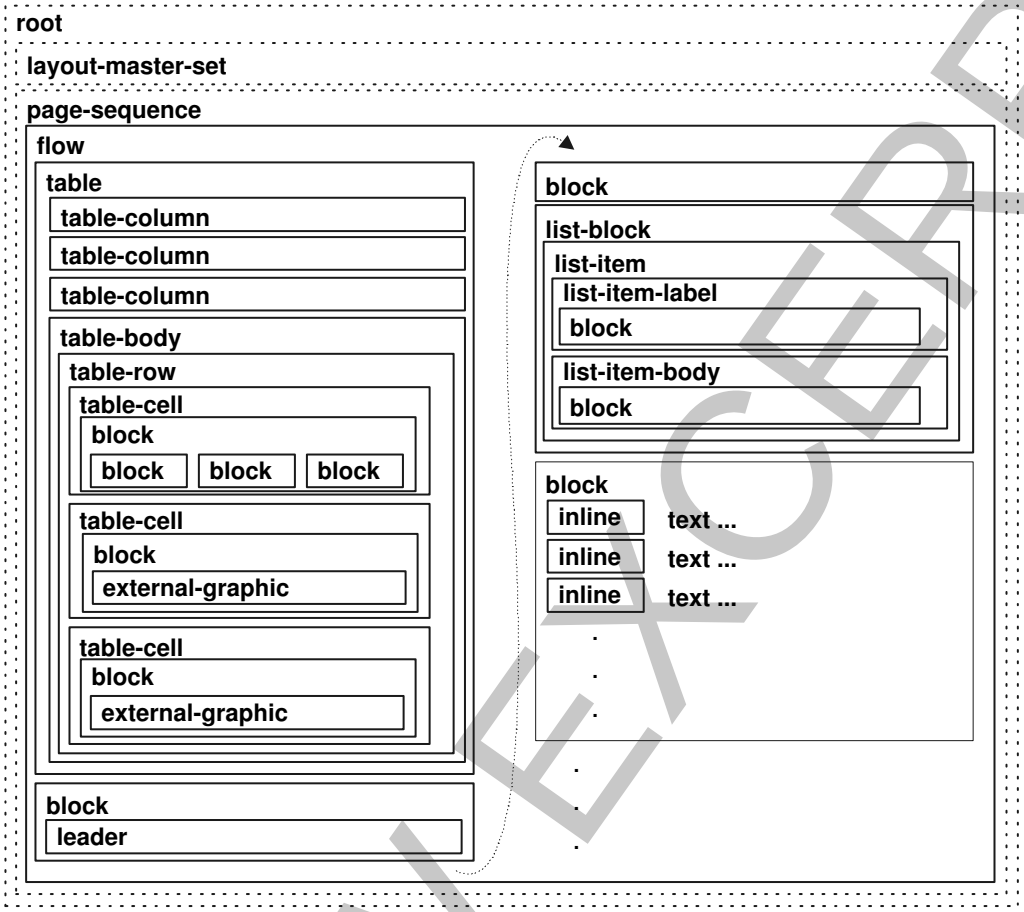
This page's material as an instructor-led handout:

- excerpt of formatting objects created using an XSLT stylesheet and an XSLT stylesheet processor

```
<flow flow-name="pages-body"><table>
 <table-column column-width="( 210mm - 2 * 15mm ) - 2in"/>
 <table-column column-width="1in"/>
 <table-column column-width="1in"/>
 <table-body><table-row><table-cell><block text-align="start">
     <block font-size="19pt">Training material example</block>
     <block font-size="10pt" space-before="10pt">Module
2 - The context of XSL-FO</block>
     <block font-size="10pt">Lesson 2 - Examples</block></block>
  </table-cell>
  <table-cell><block text-align="end"><external-graphic
    src="url(&quot;..\whitesml.bmp&quot;)"/></block></table-cell>
  <table-cell><block text-align="start"><external-graphic
    src="url(&quot;..\cranesml.bmp&quot;)"/></block></table-cell>
 </table-row></table-body></table>
 <block line-height="3px"><leader leader-pattern="rule"
      leader-length="100%" rule-thickness="1pt"/></block>
 <block space-before="6pt" font-size="14pt">
This page's material as an instructor-led handout:</block>
 <list-block provisional-distance-between-starts=".43in"
  provisional-label-separation=".1in" space-before="6pt">
  <list-item relative-align="baseline">
   <list-item-label text-align="end" end-indent="label-end()">
    <block>-</block></list-item-label>
   <list-item-body start-indent="body-start()">
    <block font-size="14pt">excerpt of formatting objects created
using an XSLT stylesheet and an XSLT stylesheet processor</block>
   </list-item-body></list-item></list-block>
 <block space-before="12pt div 2" font-family="Courier"
      linefeed-treatment="preserve" white-space-collapse="false"
      white-space-treatment="preserve" font-size="12pt"><inline
font-size="inherited-property-value(font-size) div 2">01 </inline
>&lt;flow flow-name="pages-body"&gt;&lt;table&gt;
<inline font-size="inherited-property-value(font-size) div 2"
>02 </inline> &lt;table-column column-width...
```

Training material example (cont.)

The nesting of the hierarchy of the formatting objects in the example page:



Chapter 3 - Basic concepts of XSL-FO

- Introduction - Essential concepts and terminology
- Section 1 - Basic concepts
- Section 2 - Processing model
- Section 3 - Formatting object XML vocabulary

Essential concepts and terminology

Chapter 3 - Basic concepts of XSL-FO

Layout-based vs. content-based formatting

- layout-based formatting respects the constraints of the target medium
  - limitations or capacities of target may constrain content or appearance
- content-based formatting respects the quantity and identity of the information
  - generates as much of the target medium as required to accommodate the information

Formatting is different than rendering

- expressing what you want done vs. expressing how it is accomplished
- similar to difference between declarative and imperative programming
- similar to difference between XSLT "transformation by example" and other approaches of "transformation by algorithm"

Differing processing model concepts are expressed using unambiguous terminology

- an XSL-FO instance with elements and attributes
- a formatting object tree with objects and properties
- a refined formatting object tree with objects and area traits
- an area tree with areas and traits

Formatting model and vocabulary properties extend what is currently available for web presentation

- adds arbitrary boundaries as pages instead of infinite-length canvas in a browser
- inspired by DSSSL but diverges more closely to CSS2
- classes of XSL-FO properties:
  - CSS properties by copy (unchanged CSS2 semantics)
  - CSS properties with extended values
  - CSS properties "broken apart" to a finer granularity
  - XSL-FO-specific properties
- supports multiple writing directions and reference orientations

Essential concepts and terminology (cont.)

The XSL-FO objects covered in this chapter are:

- `<root>` (6.4.2)
  - the document element of the XSL-FO instance
- `<layout-master-set>` (6.4.6)
  - the collection of definitions of page geometries and page sequencing and selection patterns
- `<page-sequence>` (6.4.5)
  - the specification of that information to be paginated over a sequence of pages with common static information
- `<flow>` (6.4.18)
  - the content that is flowed to as many pages as required and formatted according to the appearance properties

Layout-based vs. content-based formatting

Layout-based formatting accommodates the medium

- often uses absolute positioning, sizing and page counting
    - a magazine may need a particular columnist's article to appear on the right-hand edge of page 7 and the three lead stories within the first four pages
- rules of layout dictate the quantity and appearance of content
- typically unstructured authoring and formatting
    - desktop publishing, journalism, etc.

Content-based formatting accommodates the information

- rules of content dictate the layout of information (rule-based formatting)
- 40,000 to 60,000 pages in a single aircraft maintenance manual cannot be individually formatted
- typically highly-structured authoring and formatting
    - technical publications: pharmaceutical, aerospace, automotive, etc.

XSL-FO oriented more to content-based formatting than layout-based formatting

- can express the repetition of pages
    - mechanical accommodation of content
- limited support of the order of specific page sequences
    - high-caliber copy-fitting often cannot be mechanical or unattended

XSL-FO is not oriented to loose-leaf publishing

- no inherent maintenance facilities for past versions of individual pages
- no inherent support of change pages (a.k.a. "A pages")
- no inherent support of lists of effective pages

Formatting vs. Rendering

An XSL-FO instance describes the intent of how a stream of information is to be formatted

- XSL-FO instance typically generated by a stylesheet acting on an instance of XML information
  - the original vocabulary of the XML source is transformed into the XSL-FO vocabulary
    - the `#PCDATA` and attribute content is rewrapped in formatting vocabulary
  - resulting transformed file is useful for diagnostics when reified as syntax
  - opportunity for store and forward applications of formatting intent
- no feedback loop from the formatter to the stylesheet (unlike interactive agents)
  - XSL-FO information must be complete with respect to all desired behaviors of the formatter
  - special formatting cases or conditions can be accommodated through contingencies expressed in the XSL-FO semantics

Information to be formatted is repackaged from the source vocabulary into the formatting vocabulary

- information arranged in a collection of formatting objects, each specifying a part of:
  - layout
    - the location of the information positioned on the target medium
    - areas defined by the user and located within other areas
      - hierarchical tree of rectangles on each page
  - appearance/impartation
    - the conveyance of the information to the user
    - visual formatting (font, size, color, weight, etc.)
    - aural synthesis (voice, volume, azimuth, pitch, etc.)
  - pagination/flow
    - the parceling of a stream of content within the layout areas
    - generating areas as required to accommodate the quantity of information
- objects are expressed in an XSL-FO XML instance as elements

A complete alphabetical list of formatting objects is in Annex C XSL-FO object summary (page 130).

- it is not necessary to know all objects to get effective results

Formatting vs. Rendering (cont.)

Formatter responsible for interpreting intent to be rendered

- following the Recommendation the formatter determines what is to be rendered
  - how the formatter interprets the formatting objects is defined in excruciating detail
- some properties are specifically targeted to certain media

The semantics of rendering are not described in detail

- device-specific rendition based on semantics of formatting objects
- how the rendering agent accomplishes the task of effecting the result of formatting is entirely up to the agent
  - must produce the same result as the intent described by the Recommendation

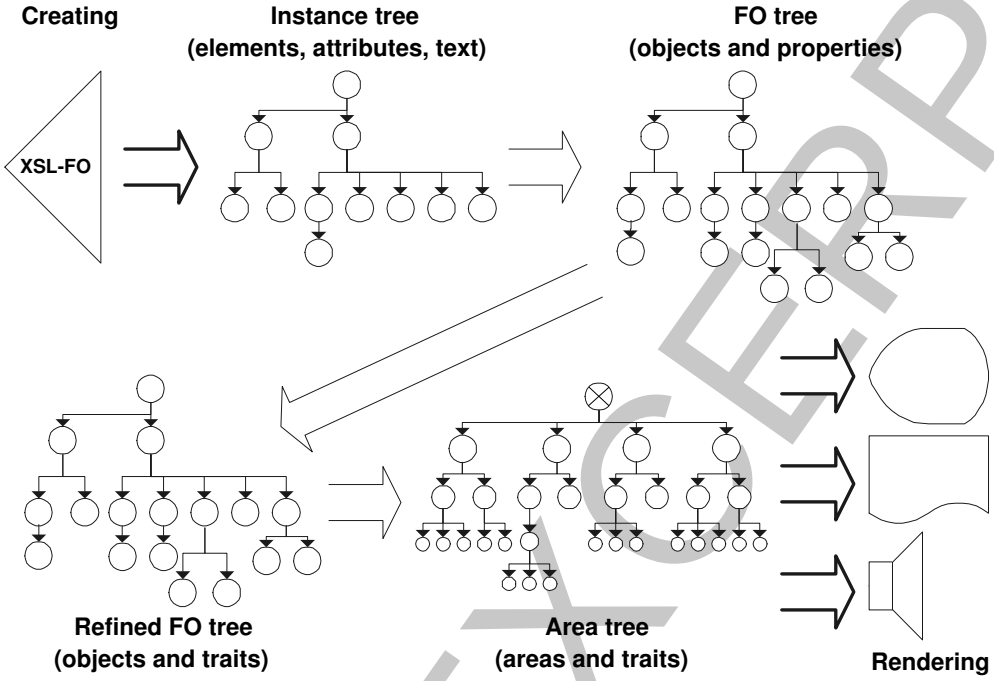
Rendering, itself, may be a multiple-step process

- production of another intermediate formatting language
  - e.g. TeX
- production of an electronic representation of the final form page description
  - e.g. Portable Document Format (PDF) by Adobe
  - e.g. Standard Page Description Language (SPDL) - ISO/IEC 10180
- interpretation and rendering to the physical final form
  - e.g. paper from the page description language
- could be many steps to final result
  - e.g. XML to XSL-FO to TeX to PDF to paper

Processing model of formatting

Chapter 3 - Basic concepts of XSL-FO
Section 2 - Processing model

From formatting objects to area tree in a well-defined process:



Processing model of formatting (cont.)

Formatting vocabulary represents typographical abstractions available to the designer

- documented as semantics in the XSL-FO Recommendation
- expressed as the intent of the designer for the formatting of the page
    - in elements, attributes and text using the XSL-FO vocabulary
- may be created outside of the formatter in an XML instance of XSL-FO
- information represented within the formatter as an Instance Tree

Formatting objects represent the specifications of layout, appearance and pagination

- information interpreted as the Formatting Object Tree
- most elements and text become formatting objects
    - in-stream foreign object elements do not become objects
    - certain white-space-only text nodes do not become objects
    - constructs not from the XSL-FO namespace that are used in `<declarations>` do not become objects
- attributes become properties
    - any property can be specified on any element

Properties are refined to traits where applicable for each formatting object

- information interpreted as the Refined Formatting Object Tree
- a single property may become one or more traits of any kind
    - formatting traits (e.g. size and position)
    - rendering traits (e.g. style and appearance)
- shorthand expansion into individual granules of properties
- expression evaluation for computed properties
    - example: property of "2em" becomes a trait of "40pt" if the font size is "20pt"
- some traits are inherited from ancestral property specifications
    - some properties are not inherited but can be explicitly inherited on request
    - inapplicable properties are removed from the refined tree
- once all traits that are applicable to formatting objects are determined, all traits not applicable to a given object are removed

Processing model of formatting (cont.)

Geometric areas on pages are returned from interpretation of formatting objects and their traits

- information interpreted as areas in the Area Tree
    - the formal theoretical expression of the semantics represented by the vocabulary
- the interpretation of some objects does not return any areas
- the interpretation of some objects returns exactly one area
- the interpretation of some objects returns as many areas as needed by descendent objects
- all areas in XSL-FO 1.0 are rectangular or square
- each area has geometric page position, z-layer position, content, background, padding, borders
- a stylesheet writer has the desired layout of areas in mind and determines which objects will return the desired areas in the desired order
- page areas are ordered but not geometrically related

Rendering agent effects the impartation of areas according to the medium

- Recommendation gives guidelines for each area and trait for visual and aural media
    - some traits are only applicable to interactive media
- some missing trait values can be arbitrarily inferred by the rendering agent (e.g. font, volume)
- the process of rendering is not specified in the Recommendation

Recommendation is written to lead a formatter implementer in carrying out requirements

- some traits are boolean values targeted to the implementer and reflecting an area's role or relative order to other areas
- rigor of Recommendation needed for finely-tuned typographical nuances
    - not written to lead the stylesheet writer in writing a stylesheet
    - many very complex interactions are not important to many stylesheets
- simple things can be done simply

Vocabulary structure

The semantics of a formatting object are defined in terms of areas and their traits

- which areas the object generates (if any)
- where the areas are placed in the area tree hierarchy
- interactions with areas from and specifications for other objects
- see Objects summarized by name (page 132) for a list of all objects

Formatting objects are specified using elements, attributes, and text

- may be a stand-alone XML instance being interpreted by a formatter
- may be a node tree passed as the result of transformation

XSL-FO elements must use the URI "`http://www.w3.org/1999/XSL/Format`" for the namespace

- extension elements ***must not*** use the XSL-FO URI
- extension attributes ***must*** have a non-null prefix
- extension constructs must be ignored if not recognized by an XSL-FO processor
- the default namespace may be used for the XSL-FO elements
- un-prefixed attributes are assumed to be in the vocabulary of the element to which they are attached

Element-type and attribute names:

- all letters are lowercase
- a hyphen separates multiple words in a single name
- a dot separates names from their compound components
- some abbreviations are used if the name is already used in XML or HTML

Examples:

- `space-before="12pt"`
  - all compound components are set to their initial values
- `space-before.conditionality="retain"`
  - sets the individual compound component to retain the space if at the start or end of a reference area

Vocabulary structure (cont.)

Every property is allowed to be specified on every object

- provides for inheritance in the formatting object tree during refinement
- only those traits meaningful to a formatting object are in the refined formatting object tree
    - utilized by some or all of the areas created by the object
- see Property summary (page 161) for a list of all properties

Every object may have a unique identifier specified using `id=` property

- used as the target of a reference from other objects
- value is not inherited from other objects
- value is assigned to the first child area generated for the object
    - if the object doesn't generate any areas, the identifier is lost

Extension objects and/or properties are allowed in any other namespace

- e.g. `xmlns:rx="http://www.renderx.com/XSL/Extensions"`
    - identifies extension functionality described by RenderX
- e.g. `xmlns:axf="http://www.antennahouse.com/names/XSL/Extensions"`
    - identifies extension functionality described by Antenna House, Inc.
- no restrictions on who or which tools can support extended functionality
- important portability issue
    - no obligation for any tool to support any extension functionality

Non-XSL-FO namespaces are allowed for embedded graphic images

- e.g. `xmlns:svg="http://www.w3.org/2000/svg"`
    - the W3C SVG 2-dimensional graphics vocabulary

Direct vs. constraint property specification

Properties influence the behavior of an object in the creation of areas for the formatted result

- properties are specified in an XSL-FO XML instance or node tree as attributes
  - properties become traits in the area tree
- strategic use can promote maintainability or consistency
  - placement in the hierarchy for inheritance
  - specified algorithmically for contextually sensitive behavior
- some properties directly specify a formatting result
  - e.g. `color=` property
  - the foreground color is specified regardless of the context in the area tree
- some properties constrain the formatting result based on an interpretation of context in the area tree
  - e.g. `space-before=` property
  - the amount of space before the area is constrained to the value specified but may be discarded if the area's conditionality allows it to be discarded and the area is at the beginning of a reference area
- compound properties are comprised of components for fine-grained specification
  - often originate from the CSS2 specification but specify too many aspects of formatting
  - e.g. `space-before.optimum=`, `space-before.minimum=`, `space-before.maximum=`, `space-before.conditionality=` and `space-before.precedence=` components
- properties can influence either layout or appearance or both

A complete alphabetical list of formatting properties is in Annex D XSL-FO property summaries (page 140).

- it is not necessary to know all properties to get effective results

Property value expressions

A property's value can be the evaluation of an expression

- may include fixed values
  - e.g. `space-before="20pt div 2"`
- may include contextually-sensitive values
  - e.g. `space-before="from-parent(font-size) div 2"`
- same operators as in XPath 1.0
- includes same operands as in XPath 1.0
- includes length values as operands that are not allowed in XPath 1.0
- expressions influenced by the font size evaluate the font size before evaluating any other components of the expression

Core function library defined for property expressions

- functions with access to property values of the current node
  - e.g. `inherited-property-value(`*`property-name`*`)`
    - this obtains a value that may be specified on the current node or may be inherited from the closest ancestral node that specifies the value
- functions with access to property values of other nodes
  - e.g. `from-parent(`*`property-name`*`)`
    - this looks for the property value directly from the parent object specification
  - e.g. `merge-property-values(`*`property-name`*`)`
    - this looks for the property value directly from the particular sibling object specification corresponding to a given state of the user interface
- summarized in Functions summarized by name (page 127)

Numerous property data types

- summarized in Property data types (page 152)
- simple-valued and compound-valued values

Numerous functions available for expressions

- summarized in Functions summarized by name (page 127)
- numeric functions can be applied to length values by reducing the "unit power" and adding it back again after
  - a length has a unit power of 1, while a number has a unit value of zero
  - e.g. `round()` takes a number argument and not a length argument
  - can use: `round( `*`length-value`*` div 1.0cm ) * 1.0cm`

Inherited properties

Inheritable properties need not be specified for a formatting object

- when not specified, the ancestry of the formatting object tree is examined for the closest specification
- all inheritance completed during refinement before the area tree is created
- see Inherited properties (page 159) for a summary

Inheritance goes "through" block-level or out-of-line constructs

- often the source of unexpected results in a stylesheet's result
- e.g. the same `start-indent=` property used to indent a table is inherited by the table's cell's content
- e.g. a footnote's body inherits properties from the block in which the footnote is specified

Some functions can obtain property values from outside the ancestry

- the attribute can specify the result of a function call instead of a fixed value
- e.g. `merge-property-values()`
- e.g. `from-table-column()`

Some properties look like they are inherited, but do not go "through" out-of-line constructs

- e.g. `text-decoration=`
  - when specified on a block-level construct, it affects all inline descendants of the block, but not any descendants of any out of line areas found in the block such as float or footnote bodies (though footnote inline areas are affected)

Shorthand properties

Shorthand properties specify a number of standalone properties using a single specification

- e.g. `font="italic small-caps bold 40pt/50pt Courier"`
  - one specification sets each of the following properties in order
    - `font-style=`
    - `font-variant=`
    - `font-weight=`
    - `font-size=`
    - `line-height=`
    - `font-family=`
- different than a compound property
  - a shorthand property represents a set of standalone properties
  - a compound property is comprised of a set of components
- individual properties of a shorthand property are supported severally
  - shorthand properties can be used in conjunction with individual properties
  - those that are specified are more precise than those that are inferred
- shorthand properties are processed in increasing precision
  - precedence order independent of attribute specification order
  - more-specific attribute name has higher precedence than less-specific attribute name
    - `border-color=` is more precise than `border=`
    - e.g. `border-bottom-color=` is more precise than `border-color=`
  - border named-edge specifications have higher precedence than generic border specifications
    - e.g. `border-bottom=` is more precise than `border-style=`

A processor is not obliged to support shorthand properties

- see Shorthand properties (page 160) for a summary
- portability problem when developing a stylesheet using one processor and delivering the stylesheet to someone else using a different processor

Conformance

Three levels of conformance that can be claimed by a processor:

- complete
    - entire Recommendation is supported
- extended
    - no support for shorthand properties
    - very few esoteric positioning- and font-related properties not supported
    - remainder of Recommendation fully supported
- basic
    - minimum level of pagination or aural rendering support
    - fallback behaviors defined for unsupported extended facilities
- implementing fallback behaviors cannot be construed as being complete conformance

In a given medium, a processor must support all formatting objects and properties specified for the conformance level claimed

- conformance levels are described for two rendering class distinctions
    - visual media
    - aural media
- can implement fallback processing in other media
    - conformance qualification does not include fallback processing

Top-level formatting objects

All XSL-FO structures must have the following formatting objects:

- `<root>`
  - the document element of an XSL-FO structure
  - handy location for document-wide inherited attributes
- `<layout-master-set>`
  - the collection of page geometries available to use for pagination
  - also collects definitions of patterns of geometry sequencing for nuances in changes of page layout

All XSL-FO structures must have at least one page sequence with flowed content to be paginated

- `<page-sequence>`
  - begin a new page with some content to be paginated
  - defines properties and content common to all pages in the sequence
    - e.g. the formatting of page numbers in the page sequence
    - e.g. the definitions of static content (i.e. headers, footers, etc.) for the page geometries selected for the page sequence
  - as many pages will be created as is needed to accommodate the flow given for the sequence
    - an exception is when the number of pages of the sequence or the starting page number of the following sequence is constrained to be an even or odd number
      - the formatter will generate a blank page to accommodate the criterion
      - the blank page can be defined with static content to inform the reader the page is intentionally left blank
    - the use of `break-before=` and `break-after=` in a sequence can cause information to start at the top of a page in the middle of the sequence
  - the formatter will use either the page geometry or the pattern of geometry sequencing to select the pages, in order, to accommodate the content and any blank pages
- `<flow>`
  - the content to be paginated in a sequence of pages of a given geometry or geometry sequence
  - can only contain block-level child formatting objects as its top-level objects

<root> Object

Purpose:

- the document element of an XSL-FO instance
- returns areas generated by children as a sequence of page-viewport-areas
    - such areas are ordered but there is no geometric relationship between areas

Content:

- (6.4.2) (layout-master-set,declarations?,page-sequence+)
- Child objects (alphabetical):
    - <declarations>(6.4.3; 112)
    - <layout-master-set>(6.4.6; 58)
    - <page-sequence>(6.4.5; 60)

Optional property:
media-usage=(7.25.11; 194)

Property of interest:

- media-usage= used when establishing either pagination with bounded page length or no pagination with infinite page length

<root> Object (cont.)

The XSL-FO instance document element:

```
<?xml version="1.0" encoding="utf-8"?>
<root xmlns="http://www.w3.org/1999/XSL/Format" font-size="16pt">
   <layout-master-set>
      <simple-page-master master-name="bookpage"
                          page-height="297mm" page-width="210mm"
                          margin-top="15mm" margin-bottom="15mm"
                          margin-left="15mm" margin-right="15mm">
         <region-body region-name="bookpage-body"
                      margin-top="5mm" margin-bottom="5mm"/>
      </simple-page-master>
   </layout-master-set>
   <page-sequence master-reference="bookpage">
      <title>Hello world example</title>
      <flow flow-name="bookpage-body">
         <block>Hello XSL-FO!</block>
      </flow>
   </page-sequence>
</root>
```

<layout-master-set> Object

Purpose:

- the collection of definitions of page geometries, available regions, and page selection patterns

Content:

- (6.4.6) (simple-page-master|page-sequence-master)+
- Child objects (alphabetical):
  - <page-sequence-master>(6.4.7; 93)
  - <simple-page-master>(6.4.12; 75)
- Referring object:
  - <root> (6.4.2; 56)

No properties are defined for this formatting object.

`<layout-master-set>` Object (cont.)

The set of layout masters:

```
01 <?xml version="1.0" encoding="utf-8"?>
02 <root xmlns="http://www.w3.org/1999/XSL/Format" font-size="16pt">
03    <layout-master-set>
04       <simple-page-master master-name="bookpage"
05                           page-height="297mm" page-width="210mm"
06                           margin-top="15mm" margin-bottom="15mm"
07                           margin-left="15mm" margin-right="15mm">
08          <region-body region-name="bookpage-body"
09                       margin-top="5mm" margin-bottom="5mm"/>
10       </simple-page-master>
11    </layout-master-set>
12    <page-sequence master-reference="bookpage">
13       <title>Hello world example</title>
14       <flow flow-name="bookpage-body">
15          <block>Hello XSL-FO!</block>
16       </flow>
17    </page-sequence>
18 </root>
```

<page-sequence> Object

Chapter 3 - Basic concepts of XSL-FO
Section 3 - Formatting object XML vocabulary

Purpose:

- the definition of information for a sequence of pages with common static information
- generates as many pages as will fit the flowed content that is supplied
- each new page sequence begins the flow at the top of a new page
    - e.g. starts of chapters, front matter, back matter, etc.

Content:

- (6.4.5) (title?,static-content*,flow)
- Child objects (alphabetical):
    - <flow>(6.4.18; 62)
    - <static-content>(6.4.19; 92)
    - <title>(6.4.20; 75)
- Referring object:
    - <root> (6.4.2; 56)

Required property:
master-reference=(7.25.9; 193)

Optional properties:
country=(7.9.1; 179)
force-page-count=(7.25.6; 185)
format=(7.24.1; 185)
grouping-separator=(7.24.2; 186)
grouping-size=(7.24.3; 186)
id=(7.28.2; 187)
initial-page-number=(7.25.7; 187)
language=(7.9.2; 189)
letter-value=(7.24.4; 191)

Shorthand influencing the above properties:
xml:lang=(7.29.24; 215)

Properties of interest:

- master-reference= selects the page geometry or sequencing of page geometry and the available regions within those page definitions
    - cannot change the page geometry based on content found in the flow
- format= governs how the page number is represented as a sequence of glyphs
    - as used on the page itself
    - as used for citations to the page
    - numbers, roman numerals, etc.
        - utilizes grouping-size=, grouping-separator=, country=, and language= if appropriate
    - formally defined by reference to XSLT (see page 158)

<page-sequence> Object (cont.)

A sequence of pages:

```
01 <?xml version="1.0" encoding="utf-8"?>
02 <root xmlns="http://www.w3.org/1999/XSL/Format" font-size="16pt">
03    <layout-master-set>
04       <simple-page-master master-name="bookpage"
05                           page-height="297mm" page-width="210mm"
06                           margin-top="15mm" margin-bottom="15mm"
07                           margin-left="15mm" margin-right="15mm">
08          <region-body region-name="bookpage-body"
09                       margin-top="5mm" margin-bottom="5mm"/>
10       </simple-page-master>
11    </layout-master-set>
12    <page-sequence master-reference="bookpage">
13       <title>Hello world example</title>
14       <flow flow-name="bookpage-body">
15          <block>Hello XSL-FO!</block>
16       </flow>
17    </page-sequence>
18 </root>
```

<flow> Object

Purpose:

- the content that is flowed to as many pages as required to fit
    - flowed content is not repeated by the formatter
- the length of the flow governs the length of the page sequence

Content:

- (6.4.18) (%block;)+
- Child object:
    - %block; (6.2; 65)
- Referring object:
    - <page-sequence> (6.4.5; 60)
- may begin with any number of <marker> children

Required property:
flow-name=(7.25.5; 182)

Property use:

- flow-name= indicates into which area on the page the child information is to be flowed
    - area does not have to exist on the particular page geometry being rendered
        - the method to ignore a flow on particular pages

<flow> Object (cont.)

A flow of areas for a region:

```
01 <?xml version="1.0" encoding="utf-8"?>
02 <root xmlns="http://www.w3.org/1999/XSL/Format" font-size="16pt">
03    <layout-master-set>
04       <simple-page-master master-name="bookpage"
05                           page-height="297mm" page-width="210mm"
06                           margin-top="15mm" margin-bottom="15mm"
07                           margin-left="15mm" margin-right="15mm">
08          <region-body region-name="bookpage-body"
09                       margin-top="5mm" margin-bottom="5mm"/>
10       </simple-page-master>
11    </layout-master-set>
12    <page-sequence master-reference="bookpage">
13       <title>Hello world example</title>
14       <flow flow-name="bookpage-body">
15          <block>Hello XSL-FO!</block>
16       </flow>
17    </page-sequence>
18 </root>
```

Groupings of formatting objects for flow

Stacking

- areas added to branches of the area tree interact with adjacent areas found in the tree
- areas stack relative to adjacent areas on the rendered result in different directions:
    - block-progression direction
        - e.g. top-to-bottom for the Western European writing direction
    - inline-progression direction
        - e.g. left-to-right for the Western European writing direction

Block-level object

- stacks next to siblings in the block-progression direction
    - new blocks restart in the block-progression direction
- breaks the flow of information in the inline-progression direction
    - siblings cannot be "beside" each other in the inline-progression direction
    - e.g.: blocks, tables, lists, etc.
- the `<block>` object is a frequently used construct
    - e.g. for paragraphs, headings, captions, table cell contents, etc.

Inline-level object

- stacks next to siblings in the inline-progression direction
- does not restart the flow of information in the block-progression direction
- specifies portions of content to be flowed into lines of the parent block where the portions are distinct from their sibling portions

Neutral object

- an object that is allowed anywhere without impacting the stacking of siblings or progression direction of the parent

Out-of-line object

- areas stack out of line to the areas of siblings of the object, without impacting the siblings of the object, by adding areas to different branches of the area tree than the siblings
- does not break the flow of information in any progression direction

Out-of-line inline-level object

- a portion stacks next to the siblings of the object in the line-progression direction
- a portion stacks out-of-line to the siblings of the object, without impacting the siblings of the object

Block-level objects

Those objects represented by the `%block;` parameter entity used in content models:

- `<block>` (6.5.2; 75)
  - the formatting specification for a block of lines, distinct from any preceding block of lines
- `<block-container>` (6.5.3; 75)
  - the specification of a block-level reference area for contained descendant blocks
- `<list-block>` (6.8.2; 81)
  - the parent object of a related set of child list members
- `<table-and-caption>` (6.7.2; 87)
  - the parent object of a captioned collection of tabular content
- `<table>` (6.7.3; 87)
  - the parent object of an uncaptioned collection of tabular content

Allowed as children of the `<flow>` object

- may also be nested inside of other constructs

Stacked in the block-progression-direction of flow

- these interrupt the block-progression flow of information
- two objects of these types cannot be positioned next to each other in the inline-progression-direction within the same containing object
- note that `<block-container>` can selectively be absolutely positioned outside of the flow and its areas will then not stack with its sibling areas

Inline-level objects

Those objects represented by the `%inline;` parameter entity used in content models:

- `<basic-link>` (6.9.2; 81)
  - the inline content of the start of a unidirectional link to a single end point
- `<bidi-override>` (6.6.2; 112)
  - overriding the inherent Unicode text direction for a sequence of characters
- `<character/>` (6.6.3; 112)
  - both the abstract formatting object implied by a simple character in an XSL-FO instance and the concrete formatting object available to be used in place of a simple character
- `<external-graphic/>` (6.6.5; 81)
  - the inline display of graphical or other externally-supplied information
- `<inline>` (6.6.7; 75)
  - the formatting specification for inline content that is distinct from its preceding content within a line generated in a block
- `<inline-container>` (6.6.8; 75)
  - an inline reference area for contained blocks
- `<instream-foreign-object>` (6.6.6; 81)
  - the inline display of graphical or other instance-supplied information
- `<leader>` (6.6.9; 81)
  - the inline display of a rule or a sequence of glyphs
- `<multi-toggle>` (6.9.5; 109)
  - the definition of those interaction-sensitive objects within a candidate rendered sequence of formatting objects
- `<page-number/>` (6.6.10; 92)
  - an inline-level place holder replaced with the page number of the current page
- `<page-number-citation/>` (6.6.11; 75)
  - an inline-level place holder replaced with the page number of the first normal area of the cited formatting object

Not allowed as children of the `<flow>` object

- may be nested inside of constructs (including other inline constructs)

Stacked in the inline-progression-direction of flow

- these do not interrupt the block-progression flow of information
- they reside in the line areas generated by the formatter

Neutral objects

Objects typically allowed anywhere where `#PCDATA`, `%inline;`, or `%block;` constructs are allowed:

- `<multi-properties>` (6.9.6; 109)
  - the collection of candidate property sets from which exactly one set influences the properties of a formatting object based on its status or the status of user interaction
- `<multi-switch>` (6.9.3; 109)
  - the collection of candidate formatting object sequences from which exactly one is rendered at any given time based on an interactive condition that is influenced by the operator while being tracked by the formatter
- `<retrieve-marker/>` (6.11.4; 92)
  - a place holder replaced with the formatting objects of the indicated marker and allowed only within static content
- `<wrapper>` (6.11.2; 75)
  - a generic container construct for specifying inherited properties for descendent constructs

Note that individual flow objects above may have constraints preventing their use in particular objects.

Areas returned by the interpretation of these objects are stacked in the progression-direction of the siblings of these objects

- except for `<retrieve-marker>` the others may be children of the `<flow>` object

Out-of-line objects

Out-of-line objects typically allowed anywhere where `#PCDATA`, `%inline;` , or `%block;` constructs are allowed:

- `<float>` (6.10.2; 103)
    - content that is to be rendered towards either the before, start, or end edges of a region regardless of where in the region the content is defined

Not stacked in the progression-direction of the sibling objects

- areas returned are contained within and governed by ancestral `<page-sequence>` object

Allowed as a child of the `<flow>` object

- may also be nested inside of other constructs

Out-of-line inline-level objects

Out-of-line objects allowed only inline, i.e.: anywhere where `#PCDATA` or `%inline;` constructs are allowed:

- `<footnote>` (6.10.3; 103)
  - content that is to be rendered towards the after-edge of a region regardless of where in the region the content is defined

One of two generated areas stacked in the progression-direction of the sibling inline objects

- other area returned is contained within and governed by ancestral `<page-sequence>` object

Not allowed as a child of the `<flow>` object

- must be placed in a block to behave at the block level

Chapter 4 - Area and page basics

- Introduction - Area and page overview

Area and page overview

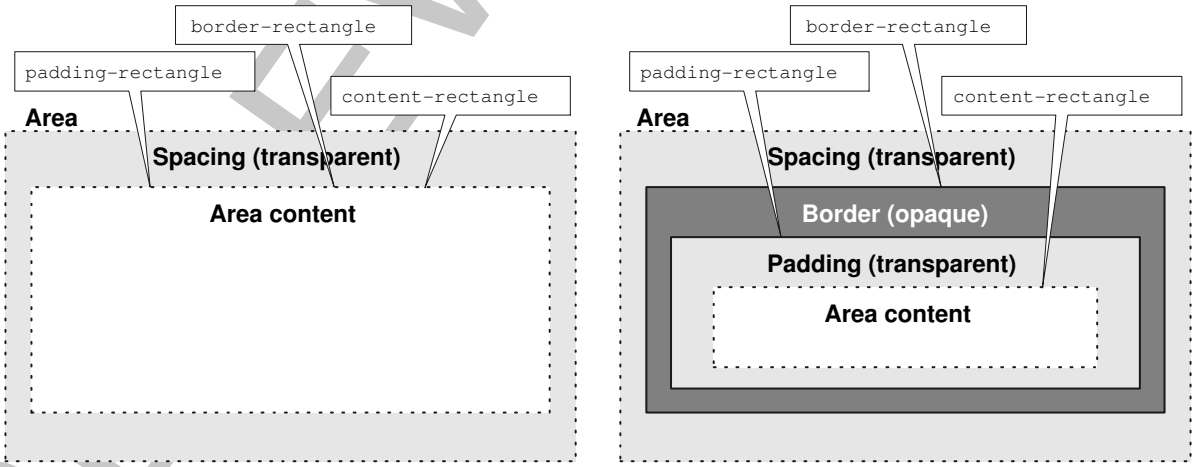
Chapter 4 - Area and page basics

Understanding areas and pages is critical to writing successful XSL-FO instances

- the rendering process renders the area tree created by formatting objects
- to get the desired rendered result, one must know how to position and nest areas of content and set their traits
    - formatting objects are chosen to give the desired layout result
    - the names of formatting objects may be totally irrelevant to the reason they are used in an XSL-FO instance to get the particular areas desired

Area model describes the nature of the areas of content that can be created

- XSL-FO 1.0 only defines rectangular areas (some of which may be square)
- areas of canvas arranged in hierarchical order in area tree
    - child areas arranged inside the parent area's content-rectangle
    - formatting objects may add areas to multiple branches of the hierarchy
    - objects in a given branch of the area tree stack next to each other
- many different rectangles define the formatter's behavior for the area's content
    - area itself is spaced between its siblings and within parent using a transparent spacing specification
    - rectangles of an opaque border around content may be specified
        - with a border thickness described by the differences between respective edges of two rectangles and a pattern with transparent background
    - rectangles of transparent perimeter spacing around rendered child content within the inside edge of the border (the padding rectangle)
- the number of rectangles in play and their nuances can be overwhelming
    - it is not necessary to know all the rectangles to get simple good-quality results
    - it is important to be aware of the different rectangles to better understand the interplay of areas and the controls available in XSL-FO properties



Area and page overview (cont.)

Writing direction and reference orientation govern visual placement of areas on a page

- these values define the block-progression and inline-progression directions for an area
- supports natural directions of common writing systems of the world
- orientation can be overridden to produce special effects in the rendered result
- values also define the before- and after-sides in the block-progression direction and the start- and end-sides in the inline-progression direction

Most child areas inherit unspecified behaviors from the parent areas

- block and inline content stack in the layout areas as specified by the stacking properties
- it is often not necessary to specify most behaviors as these are already implied from the ancestral areas

Child "container" objects are used to create areas that override orientation behavior

- areas created by container objects can alter their behavior to meet specific requirements that differ from the parent area
    - can specify an absolute position outside of the parent for a block container
    - can specify an overflow behavior
    - can specify a different writing direction or reference orientation
- both block-level and inline-level container objects are available to be used within their respective types of parent objects

Areas on the page are not mutually exclusive

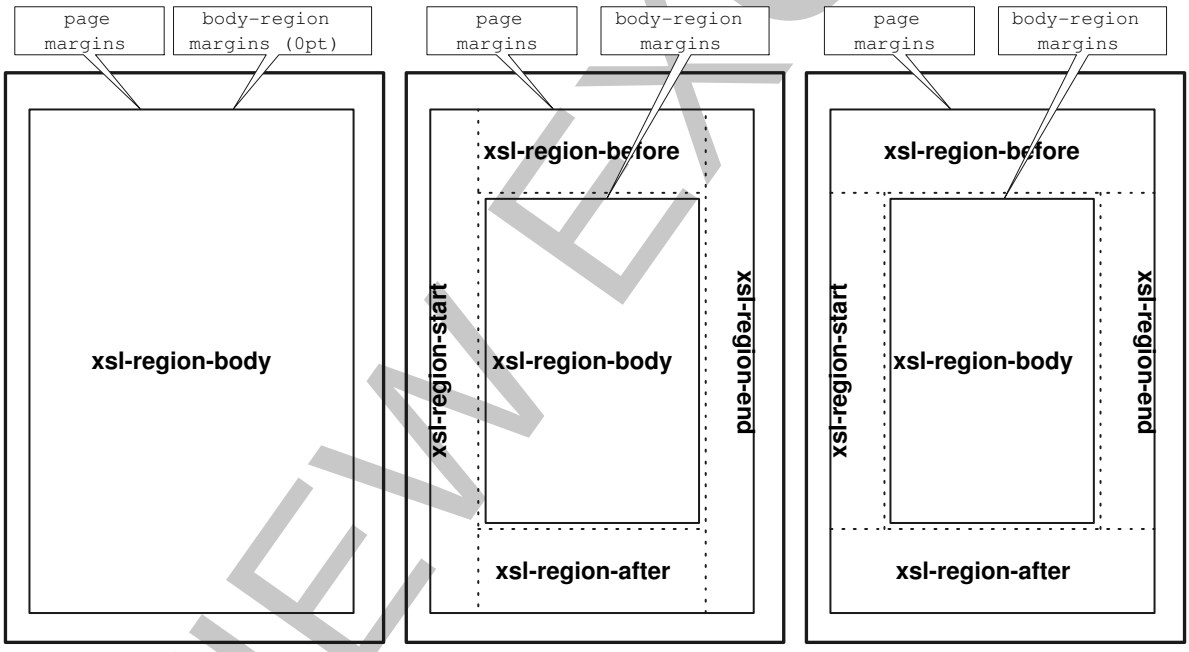
- areas can be formatted to overlap other areas in whole or in part
    - transparent backgrounds show other areas behind
- sibling areas in a given branch of the area tree typically do not overlap
- common formatting problems occur when areas from different branches of the tree occupy the same real estate on the page
    - must plan ahead so the stacking of areas in one branch doesn't interfere with the stacking of areas on other branches

Page geometry

Chapter 4 - Area and page basics

A page is described by the geometry of its size and various regions

- every page has a `<region-body>` with the initial name `xsl-region-body`
  - content flowed in this region goes is in the main reference area
- possible incursion into the body by four perimeter regions on the four edges
  - `<region-before>` with the initial name "`xsl-region-before`"
  - `<region-after>` with the initial name "`xsl-region-after`"
  - `<region-start>` with the initial name "`xsl-region-start`"
  - `<region-end>` with the initial name "`xsl-region-end`"
  - precedence controls which regions occupy the corners of the perimeter
    - default precedence given to `<region-start>` and `<region-end>`
    - overridden only by `precedence=` property on `<region-before>` and `<region-after>`
- regions are referenced in XSL-FO by using their name
  - regions can be custom-named



Of note:

- the middle page shows the start and end regions have precedence for the corners of the page over the before and after regions (the default)
- the `precedence=` property is used to give precedence individually to the before and after regions when needed

Page geometry (cont.)

Page regions are targets for either paginated flow or static content

- paginated flow triggers as many pages as needed by the amount of flowed content
  - when a page's region accepting flow overflows, a new page in the page sequence is triggered
  - different regions on separate pages can accept paginated flow (but not on the same page)
  - the flow indicates the name of the target region for the content
- static content for reproduction on every page triggered by pagination
  - the static definition indications the name of the target region for the content
    - only if the new page includes the named region does a given region's static content get rendered
  - components of static content may be dynamically populated with the page number and the content of user-defined markers appearing on the page being formatted

The formatting objects in each region create descendant areas in that region's branch in the area tree

- the sibling areas in each branch stack separately from the sibling areas in other branches
- without proper body region margins, the perimeter region areas will overlap on top of the body region areas

Area and page constructs

Chapter 4 - Area and page basics

The XSL-FO objects covered in this chapter are summarized as follows.

Content-oriented formatting objects:

- `<wrapper>` (6.11.2)
  - a neutral construct for specifying inherited properties for descendent constructs
- `<block>` (6.5.2)
  - the description of canvas content that is distinct from its preceding area content
- `<initial-property-set>` (6.6.4)
  - an auxiliary construct for specifying properties applied to the first line of the parent
- `<inline>` (6.6.7)
  - a description of canvas content that is distinct from its preceding content within a line generated in a block
- `<page-number-citation>` (6.6.11)
  - an inline-level place holder replaced with the page number of the first normal area of the cited formatting object

Container formatting objects:

- `<block-container>` (6.5.3)
  - the specification of a block-level reference area for contained descendant blocks
- `<inline-container>` (6.6.8)
  - the specification of an inline-level reference area for contained descendant blocks

Page-oriented formatting objects:

- `<simple-page-master>` (6.4.12)
  - the specification of a given page's physical geometry
- `<region-body>` (6.4.13)
  - the definition of the middle area inside any perimeter defined for the page
- `<title>` (6.4.20)
  - a page sequence's ancillary description not rendered on the page canvas

Chapter 5 - Generic body constructs

- Introduction - Often-used formatting constructs

Often-used formatting constructs

Chapter 5 - Generic body constructs

Many publishing requirements involve the following commonly-used constructs:

- pairs of aligned block-level layout areas in the inline-progression direction
  - normally flowing a block-level area breaks the block progression direction, stacking the block area after the previous area
    - holds two block-level areas stacked beside each other
    - block areas are aligned on their respective before edges
  - coupled members (e.g. side-by-side translated polyglot (multi-language) text)
  - unordered members of a collection (e.g. bulleted lists)
  - ordered members of a collection (e.g. sequenced lists with labels of alpha, roman, number, etc.)
- non-textual information
  - static images or dynamic windows into live applications
  - information external to the XSL-FO instance
    - mandatory for non-XML expressions of the information
    - optional for XML expressions of the information (e.g. SVG)
  - information embedded in the XSL-FO instance
    - must be an XML expression of the information (e.g. SVG)
- unidirectional associations of information
  - from areas of the area tree to a target external resource
  - from areas of the area tree to a target area in the same area tree
  - interactivity engaged by the operator viewing the rendered result can navigate the operator to the target location triggering a traversal of the association
  - unidirectional links have no "back link" information to retrace steps
    - such functionality in browsers and page turner applications is maintained by the application itself, not by the inherent properties of the link
- elastic and inelastic inline areas and decorations
  - forcing inline items to the opposite boundaries of a line
  - assistance when the eye travels from one side of a page to the opposite side
    - used in entries in tables of content
  - patterned sequences of characters joining information on a single line
    - e.g. a dot leader
  - drawn rules in the inline-progression direction
    - a non-textual straight-ruled mark
  - when the length is specified the construct is inelastic
    - useful standalone to break up the flow of information with visual barriers
  - when the length cannot be predetermined, the construct can be elastic
    - grows to the length needed

Often-used formatting constructs (cont.)

Chapter 5 - Generic body constructs

An example of a side-by-side bilingual text presentation of an excerpt from the Canadian statute "Employment Equity Act, S.C. 1995, c. 44":

13. Every employer shall, at least once during the period in respect of which the short term numerical goals referred to in paragraph 10(1)(d) are established, review its employment equity plan and revise it by

(a) updating the numerical goals, taking into account the factors referred to in subsection 10 (2); and

(b) making any other changes that are necessary as a result of an assessment made pursuant to paragraph 12(b) or as a result of changing circumstances.

13. Au moins une fois au cours de la période pour laquelle les objectifs quantitatifs à court terme sont fixés, l'employeur procède à la révision de son plan en lui apportant les aménagements rendus nécessaires du fait du suivi ou du changement de sa situation et en adaptant les objectifs quantitatifs, compte tenu des facteurs visés au paragraphe 10(2).

14. Every employer shall provide information to its employees explaining the purpose of employment equity and shall keep its employees informed about measures the employer has undertaken or is planning to undertake to implement employment equity and the progress the employer has made in implementing employment equity.

14. L'employeur informe ses salariés sur l'objet de l'équité en matière d'emploi et leur fait part des mesures qu'il a prises ou qu'il entend prendre pour réaliser l'équité en matière d'emploi, ainsi que des progrès qu'il a accomplis dans ce domaine.

15. (1) Every employer shall consult with its employees' representatives by inviting the representatives to provide their views concerning

15. (1) L'employeur consulte les représentants des salariés et les invite à donner leur avis sur les questions suivantes :

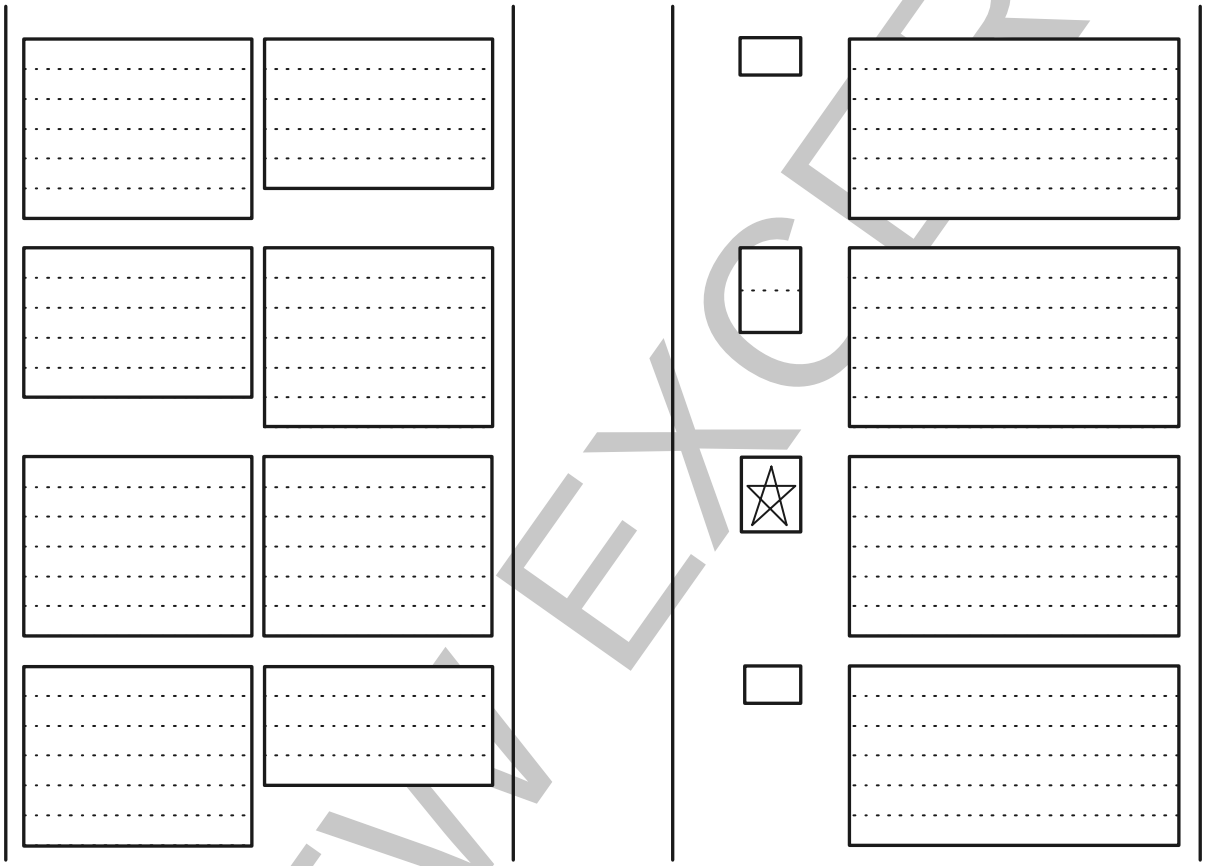
Note:

- the paragraphs are aligned at the before edges
- the sizes of the respective paragraphs are different
- no need to calculate the distance needed to align the starts of paragraphs

Often-used formatting constructs (cont.)

Aligned pairs of block-level layout areas can be used for different purposes

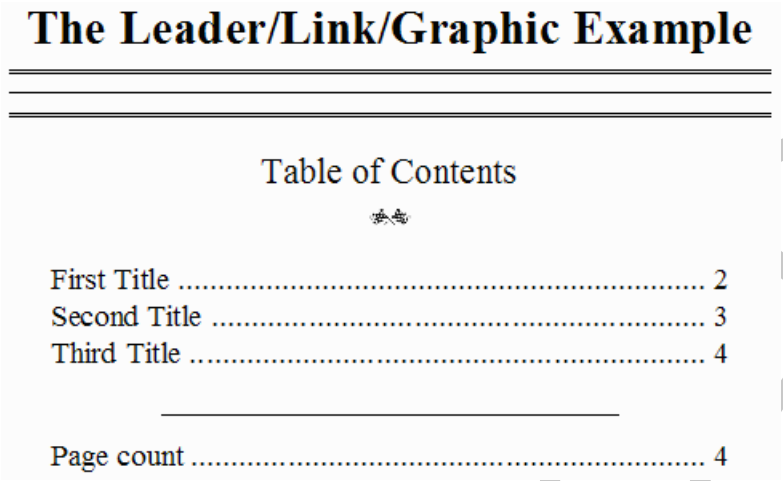
- side-by-side presentation
  - e.g. simultaneous translation with aligned paragraphs
- traditional list structures
  - e.g. numbered lists, bulleted lists, terminology definitions, etc.



The same XSL-FO semantic is used for both kinds of layout above.

Often-used formatting constructs (cont.)

The `samp/leadlink.fo` example illustrates leaders, links and graphics:



The Leader/Link/Graphic Example

Table of Contents

First Title 2
Second Title 3
Third Title 4

Page count 4

The text of each line of the table of contents is "hot"

- the operator interacts with a hot area in order to traverse a link
  - e.g. a mouse click
- the operator is moved to the focus to the target of the association
  - e.g. another page in the same XSL-FO formatted result
  - e.g. a page in the another XSL-FO formatted result
  - e.g. a web browser with a web page address

A graphic image shown below the title

- in this example, this is an external bit image

Various leaders are used on the page

- near the top there are inelastic rule leaders 100% of the width of the page
- above the page count there is an inelastic rule leader 60% the width of the page
- in each entry there is an elastic dot leader that stretches between the start-aligned titles and the end-aligned page numbers

Often-used formatting constructs (cont.)

The XSL-FO objects covered in this chapter are summarized as follows.

List objects:

- `<list-block>` (6.8.2)
  - the collection object of a related set of child member pairs of aligned block-level areas
- `<list-item>` (6.8.3)
  - a member pair of aligned block-level areas in a collection
- `<list-item-label>` (6.8.5)
  - the start-side member of a pair of aligned block-level areas
- `<list-item-body>` (6.8.4)
  - the end-side member of a pair of aligned block-level areas

Graphic objects:

- `<external-graphic>` (6.6.5)
  - the inline display of graphical or other externally-supplied information
- `<instream-foreign-object>` (6.6.6)
  - the inline display of graphical or other instance-supplied information

Link object:

- `<basic-link>` (6.9.2)
  - the inline display of the start resource of a unidirectional link to a single end point

Leader/rule object:

- `<leader>` (6.6.9)
  - the inline elastic or rigid display of a rule or a repeated sequence of glyphs

Chapter 6 - Tables

- Introduction - Tabular presentation of information

Outcomes:

- understand basic concepts of tabular presentation

Tabular presentation of information

A tabular presentation arranges information in a rigid partitioning of groups

- tuples of aligned block-level layout areas
  - see Often-used formatting constructs (page 78) for an example of bilingual text formatting
  - a collection (rows) of collections (columns) of information
  - columns and rows are not being used as indices of a Cartesian plane
    - the grid is only a layout strategy
  - e.g. columns of aligned formatted paragraphs of polyglot (multiple language) text
    - each piece of information is a stand-alone paragraph where each language's paragraph has a different length thus requiring the first line of all translations of the next paragraph to be aligned
- block-level layout areas in a two-dimensional relationship between members of collections
  - traditional Cartesian arrangement of information at the intersection of indexed row and column axes
  - e.g. the corresponding percentages of male and female population at the intersection of age groups in the columns and countries in the rows
    - each piece of information could be a two-line presentation of the percentages for each gender
    - the reader correlates the age group column with the country row to find the information in the corresponding cell

Supports numerous block-level areas arranged in the inline-progression direction

- table contains rows (in block-progression direction) of cells (in inline-progression direction) where each cell contains block-level areas (in block-progression direction)
  - block-level areas would otherwise be arranged in the block-progression direction
- column widths can differ but are fixed for the length of the table
  - widths can be specified values in the supplied objects and properties
  - widths can be based on an automatic weighing of the contents of all of the cells of the table
    - e.g. HTML browser web agents balance column widths based on content

Tabular presentation of information (cont.)

An example of a Cartesian grid presentation of hockey standings:

Division standings - 2003-04-04

Eastern Conference

Northeast Division

City	Games					Points
	Total	Wins	Losses	Ties	OTL	
Ottawa	81	51	21	8	1	**111**
Toronto	81	44	27	7	3	**98**
Boston	81	35	31	11	4	**85**
Montreal	81	29	35	8	9	**75**
Buffalo	80	27	36	9	8	**71**

Atlantic Division

City	Games					Points
	Total	Wins	Losses	Ties	OTL	
New Jersey	80	45	20	9	6	**105**
Philadelphia	80	43	20	13	4	**103**
NY Islanders	79	34	32	11	2	**81**
NY Rangers	80	32	34	10	4	**78**
Pittsburgh	81	27	43	6	5	**65**

Southeast Division

City	Games					Points
	Total	Wins	Losses	Ties	OTL	
Tampa Bay	80	36	23	16	5	**93**
Washington	81	38	29	8	6	**90**
Atlanta	80	29	39	7	5	**70**
Florida	80	23	35	13	9	**68**
Carolina	80	22	41	11	6	**61**

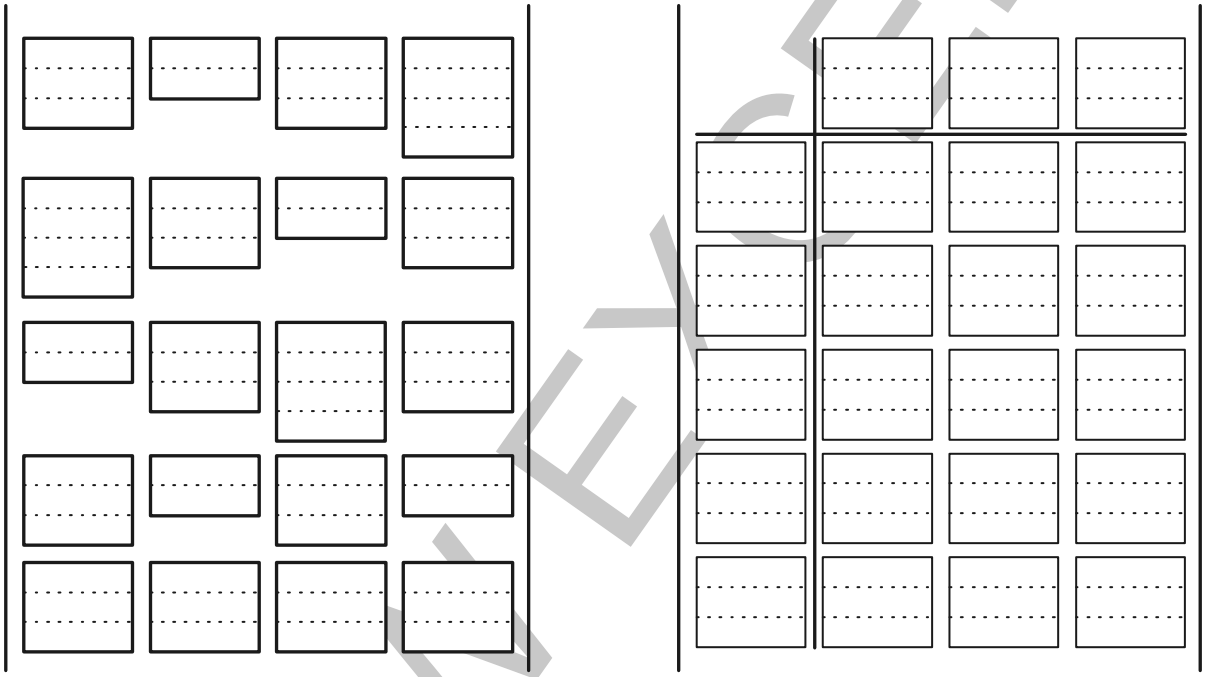
Note:

- rows are spanned for the "Columns" and "Points" headings
- columns are spanned for the "Games" heading

Tabular presentation of information (cont.)

Two layout objectives using the same layout constructs:

- a collection of tuples of information
  - each member of a tuple in a column
  - each tuple in a row
- a coordinate-based layout of cells of information
  - one ordinate along the columns
  - the other ordinate along the rows
  - the corresponding cells in the main area



Of note:

- the column widths are fixed to the same values for all rows in the entire table
  - as in HTML the widths may be based on a balancing of the content of the columns in all of the rows (default)

Tabular presentation of information (cont.)

Independent of the organization of source information

- the original data needn't be modeled by a table construct
  - historical use of structured information tools often forced information designers to model explicit table constructs
  - such models insufficiently capture the semantics of the information being related
- information from any model can be presented in a tabular fashion
  - transformation can rearrange the source information into the tabular relationship

An XSL-FO table has a compound structure with many well-defined behaviors

- a block-level construct that breaks the flow in the block-progression direction
- the caption, header rows and footer rows are constructs repeated on all pages where the table's body rows are rendered
- column properties can be specified once for all cells of rows that are in a given column
  - the linear serialization of a two-dimensional construct necessarily favors one dimension over the other
  - cells are contained within rows, not columns, so a column-oriented construct is necessary to address the second dimension of property assignment
- column-spanning and row-spanning features provide flexible table cell definition

Tabular presentation of information (cont.)

The XSL-FO objects covered in this chapter are:

- `<table-and-caption>` (6.7.2)
  - the parent object of a captioned collection of tabular content
- `<table-caption>` (6.7.5)
  - the caption of a captioned collection of tabular content
- `<table>` (6.7.3)
  - the parent object of an uncaptioned collection of tabular content
- `<table-column>` (6.7.4)
  - the specification of common columnar properties
- `<table-header>` (6.7.6)
  - the rows of tabular content repeated at the before-edge of every break in body content
- `<table-footer>` (6.7.7)
  - the rows of tabular content repeated at the after-edge of every break in body content
- `<table-body>` (6.7.8)
  - the rows of tabular content flowed as the body content
- `<table-row>` (6.7.9)
  - a row of tabular content
- `<table-cell>` (6.7.10)
  - a column of a row of tabular content

Fifth Edition - 2004-02-12 - ISBN 1-894049-13-6

Chapter 7 - Static content and page geometry sequencing

- Introduction - Extended page model for pagination
- Section 1 - Page Sequence Master Interleave (PSMI)

Outcomes:

- understand basic concepts of pagination constructs

Bounded areas (the pages) repeat to accommodate a non-predetermined amount of information (the flow)

- pagination differentiates XSL-FO semantics from web browser display semantics

The pages in the collection need navigational aids repeated on each page

- used to understand a page's role when not in the context of neighboring pages
- page numbers and page number citations
  - a focus on the specific bounded area of a piece of information
  - used on the given page for navigation
  - used on other pages for citations to the given page
    - e.g. cross references
    - e.g. index information
- headers and footers
  - contextual information about a collection of pages in which the page is found
  - e.g. chapter title repeated in the header
- cited information found from the page being formatted can be contextual information
  - information that would not be known about a page by the stylesheet at the time of transformation, possibly changing for each page generated
    - the act of pagination dictates the page boundaries, not the transformation of the source information
  - e.g. dictionary headers (finding one of many items on a page)
  - e.g. subsection citations (finding breaks not triggering new pages)

Every page sequence starts a new page and has its own definitions for static content

- all candidate uses of static content must be defined for each page sequence
- necessary for the stylesheet writer to repeat the definitions in each page sequence if the same behavior is desired
- every page sequence begins on a new page
  - changing a header or footer in the middle of a page sequence requires manipulating the dynamic content that can be placed within static content

Static content is associated with the name of a page region, not a region's position

- must organize in each page geometry the desired region names
- must bind in each page sequence the static content for named regions
- static content is used for only the named regions found on each page geometry used
  - it is not an error to supply static content for a region that isn't used
- static content can be supplied for named sub-regions triggered by the formatter

Extended page model for pagination (cont.)

Differences in page geometry are allowed when more than one page being rendered

- supports tests for differences to be performed within each page sequence construct's flow when the flow triggers the need for a new page
- different named geometries can be identical to others or have differences such as:
    - the change of physical dimensions
    - the choice of region dimensions and names
    - the change of margins
    - the presence of headers and/or footers
    - the count of columns
    - the orientation of regions
    - the backgrounds of regions
    - the absence of flowed content (to utilize an entire page of static content)

Can describe a sequence with odd and even page number parity differences

- e.g. alternating headers and footers
- two different static contents are defined with the page number to be rendered on the outside edge of each side of a bound publication
    - differences in headers and footers of the geometries alternate the names of the flows for the static content between odd and even numbered pages

Can describe a sequence with first, last, and middle page differences

- to utilize differences based on where a given formatted page is within its containing page sequence
- e.g. no heading on the first page of a chapter sequence

Can describe a sequence to replace absent content for forced un-flowed blank pages

- to accommodate a requirement for a specified parity of pages in a given sequence, or the need for the subsequent page sequence to start on a page following the page after the end of a page sequence's flow
- i.e. "this page intentionally left blank"
- by definition, a forced page is made up entirely of static content

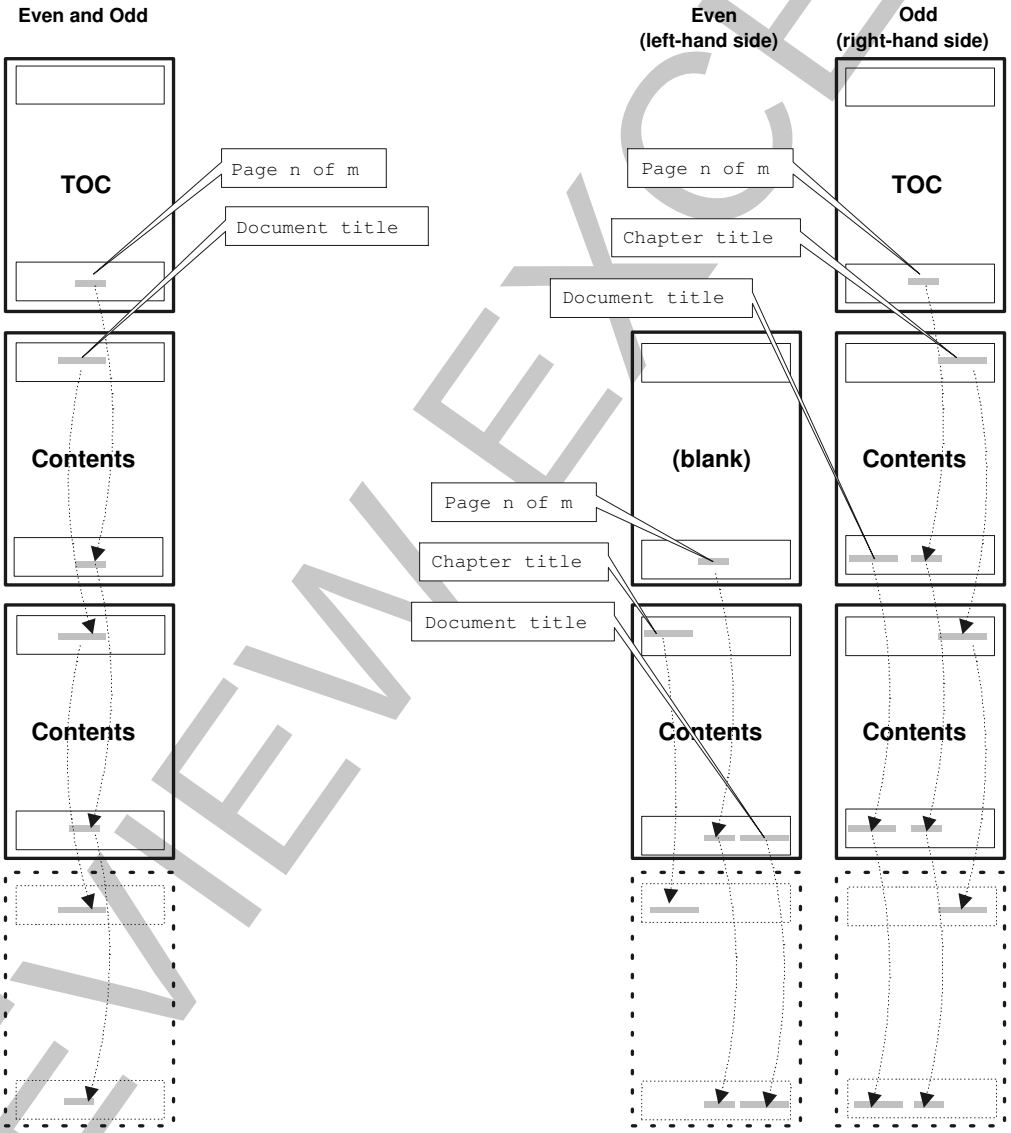
Each page sequence points to a defined sequence of page geometries

- the stylesheet writer choreographs the page geometries by the anticipated needs for the quantity of flow expected to be formatted
- all static content definitions needed for a page sequence must be present in that sequence
    - often requires stylesheet writing techniques that copy information repeatedly

Extended page model for pagination (cont.)

Consider the choreography in each of two possible page plans for a TOC and chapters:

- a single-sided presentation on the left with all page sequences consecutive:
  - document title centered at the top of document content pages (but not the TOC)
  - page number and total page count centered at the bottom of all pages
- a double-sided presentation on the right with contents starting on right-hand page:
  - requires the table of contents to be an even number of pages
  - document title at bottom left of odd pages and bottom right of even pages
  - chapter title at top right of odd pages and top left of even pages
  - page number and total page count centered at the bottom of all pages



Extended page model for pagination (cont.)

The XSL-FO objects covered in this chapter are summarized as follows.

Formatting objects related to static content:

- `<region-before>` (6.4.14)
  - the definition of the body region perimeter area whose before-edge is co-incident with the before-edge of the page's content rectangle
    - in `lr-tb` mode this is the header at the top of the page
- `<region-after>` (6.4.15)
  - the definition of the body region perimeter area whose after-edge is co-incident with the after-edge of the page's content rectangle
    - in `lr-tb` mode this is the footer at the bottom of the page
- `<region-start>` (6.4.16)
  - the definition of the body region perimeter area whose start-edge is co-incident with the start-edge of the page's content rectangle
    - in `lr-tb` mode this is the sidebar at the left of the page
- `<region-end>` (6.4.17)
  - the definition of the body region perimeter area whose end-edge is co-incident with the end-edge of the page's content rectangle
    - in `lr-tb` mode this is the sidebar at the right of the page
- `<static-content>` (6.4.19)
  - the definition of content that is primarily unchanged from page to page in a page sequence
    - entire sequence is repeated on each page except for page numbers and user-defined markers
- `<page-number>` (6.6.10)
  - an inline-level place holder replaced with the page number of the current page
- `<retrieve-marker>` (6.11.4)
  - an inline-level place holder replaced with the formatting objects of the indicated marker
- `<marker>` (6.11.3)
  - the replacement formatting object content for a marker retrieved in static content

Extended page model for pagination (cont.)

Formatting objects related to page geometry sequencing:

- `<page-sequence-master>` (6.4.7)
  - the definition and name of a particular sequence of using page-masters
- `<single-page-master-reference>` (6.4.8)
  - the specification of the single use of a page-master within a sequence of page-masters
- `<repeatable-page-master-reference>` (6.4.9)
  - the specification of the repeated use of a page-master within a sequence of page-masters
- `<repeatable-page-master-alternatives>` (6.4.10)
  - the collection of possible page-master references from which one is to be used based on status conditions detected by the formatter
- `<conditional-page-master-reference>` (6.4.11)
  - a page-master choice available to the formatter when selecting from a collection of candidate page-masters

Changing the page geometry based on content

Sometimes it is necessary to change page geometry based on content

- the XML source author may arbitrarily request a change in the page geometry
- consider the need to change the geometry for a lengthy landscaped table
    - cannot use a rotated table because the table length is limited by the container
    - must switch to a landscape body region so the lengthy table flows to subsequent pages
- consider the need to flow certain figures into double-sized pages
    - must selectively choose a geometry with the different dimensions
- the construct may be deep within the structure of the document being processed

To use XSL-FO one must package all content for each page geometry for the flow of a page sequence

- this can involve a difficult and wasteful recursive process
    - finding all information up to the change in geometry
    - package the information up in a page sequence
    - package up the special constructs in a different page sequence
    - recursively find all information up to the next change in geometry
- this breaks the XSLT model of hierarchical processing
    - susceptible to development and maintenance problems

A two-step process can make this very easy to implement

- the first step creates the flow with supplemental indications of the need to change geometries
    - can come from any depth of processing of the source document
    - block level constructs of a page are the immediate children of the flow
- the second step repackages flow children in as many sequences as necessary
    - recursively checks all flow child constructs for any changes in geometry
    - not necessary to check any depth deeper than the children
- the resulting document is then processed by a standard XSL-FO engine

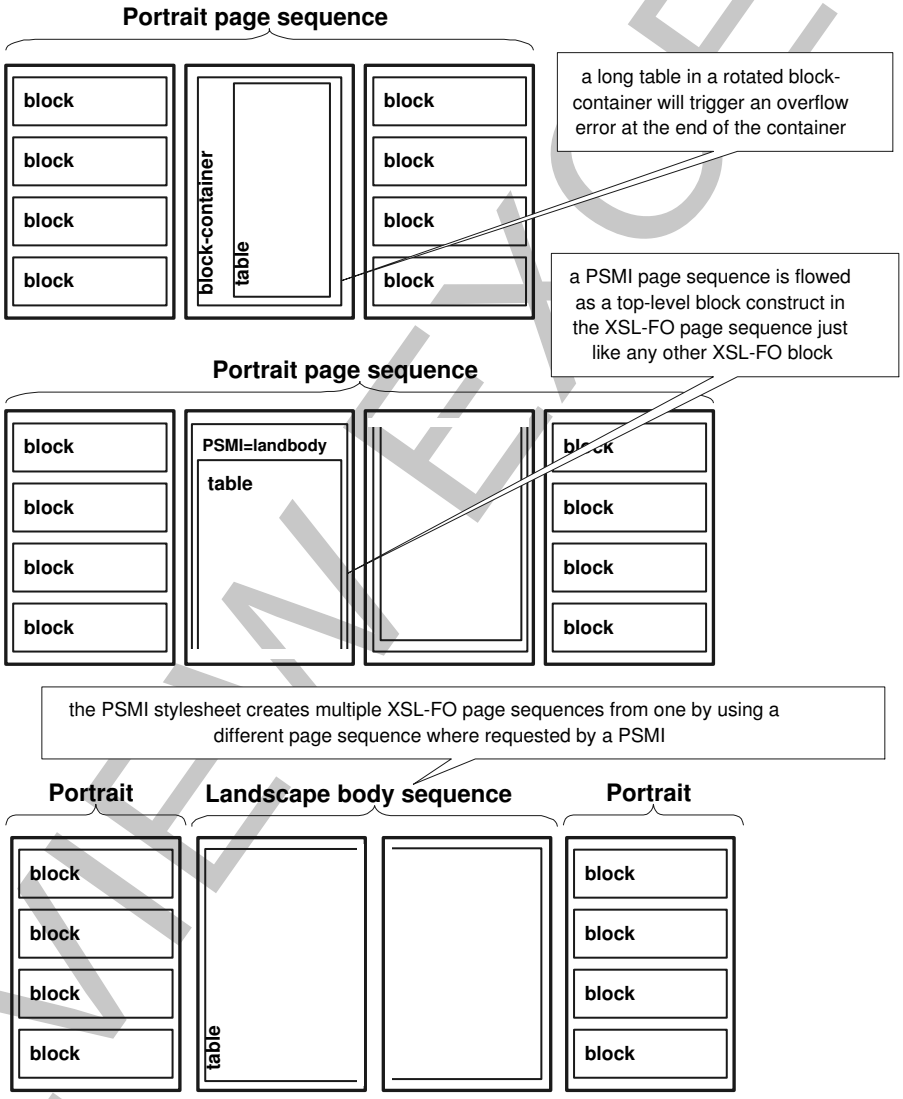
The Page Sequence Master Interleave (PSMI) semantic implements this algorithm

- public resource freely available from Crane Softwrights Ltd.
    - `http://www.CraneSoftwrights.com/links/res-pfux.htm`
- one-element vocabulary designed to express the semantic of this two-step intermediate process
- an XSLT stylesheet to implement the semantic for any XSL-FO+PSMI sequence

Changing the page geometry based on content (cont.)

Consider the need to flow a landscape table in a portrait page geometry

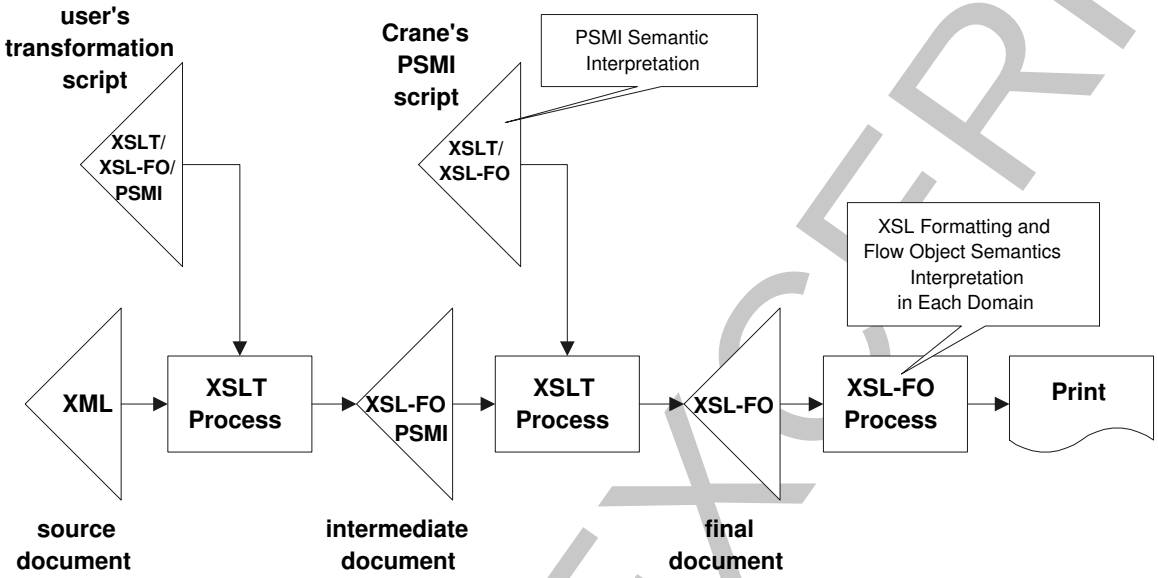
- could rotate a short table in a block container to present in landscape
  - a long table would overflow and not trigger a new page in pagination
- using PSMI one could flow the landscape table into an interleaved page geometry where the body region of that geometry is landscape
- the PSMI stylesheet reads the XSL-FO+PSMI instance and produces a pure XSL-FO instance where three page sequences are created where one was before
- a long table in the landscape body region will correctly trigger new pages in pagination



Changing the page geometry based on content (cont.)

The two-step process involves:

- the user's stylesheet to produce the XSL-FO+PSMI intermediate sequence
- the PSMI stylesheet to produce a pure XSL-FO instance



Changing the page geometry based on content (cont.)

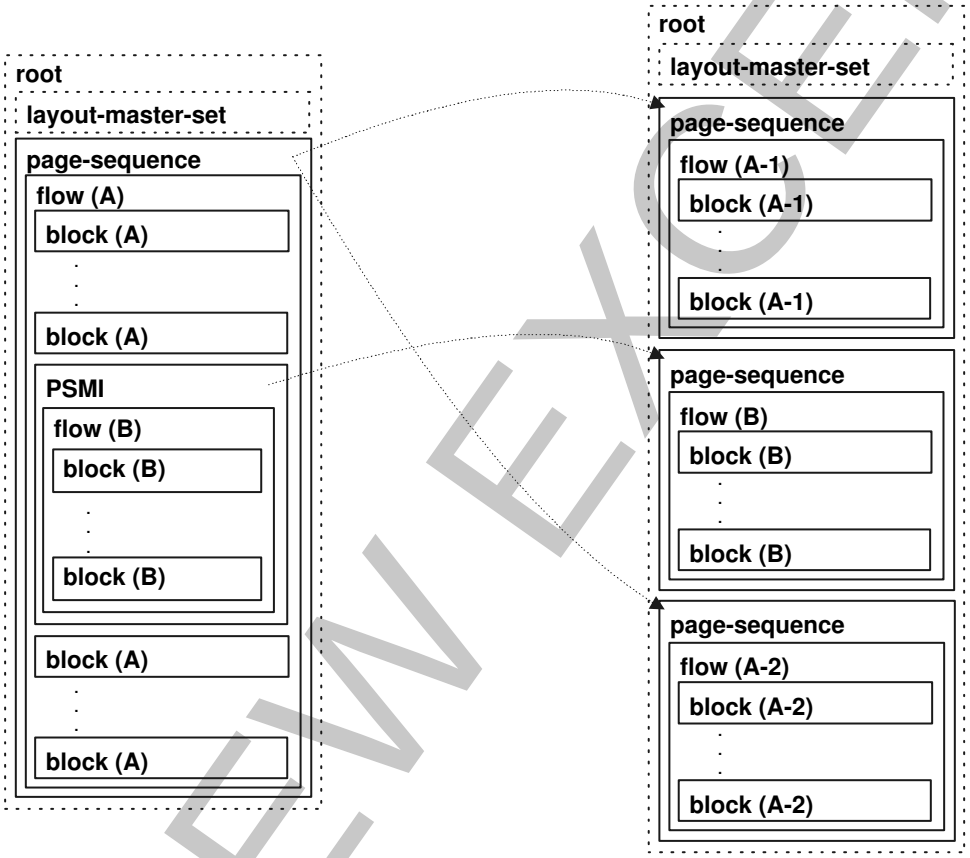
An example of flow implementing the change in page geometry:

```
    <simple-page-master master-name="frame-portraitbody" ...>
       <region-body region-name="frame-body" .../>
       <region-before region-name="frame-before" .../>
       <region-after region-name="frame-after" .../>
    </simple-page-master>
    <simple-page-master master-name="frame-landbody" ...>
       <region-body region-name="frame-body"
                    reference-orientation="90deg" .../>
       <region-before region-name="frame-before" .../>
       <region-after region-name="frame-after" .../>
    </simple-page-master>
...
<page-sequence master-reference="frame-portraitbody">
 <flow flow-name="frame-body">
  <block>Portrait information</block>
  ...
  <block>Next is landscaped</block>
  <psmi:page-sequence master-reference="frame-landbody"
   xmlns:psmi="http://www.CraneSoftwrights.com/resources/psmi">
   <flow flow-name="frame-body">
    <table>
     <table-body>
      <table-row>
       <table-cell border="solid">
        <block>This is a test</block>
        <block>This is a test</block>
        ...
        <block>This is a test</block>
        <block>This is a test</block>
       </table-cell>
      </table-row>
     </table-body>
    </table>
   </flow>
  </psmi:page-sequence>
  <block>Back to portrait</block>
  ...
 </flow>
```

Changing the page geometry based on content (cont.)

The end result of PSMI transformation is a pure XSL-FO instance:

- the parent XSL-FO page-sequence of the PSMI page-sequence is split
- a new XSL-FO page-sequence is created for the PSMI page-sequence and contains its blocks
- the preceding siblings of the PSMI page-sequence are put in the preceding page sequence
- the following siblings of the PSMI page-sequence are put in the following page sequence
- the process accommodates any number of PSMI children of an XSL-FO page-sequence



Chapter 8 - Floats and footnotes

- Introduction - Out-of-line publishing constructs

Outcomes:

- understand basic concepts of out-of-line constructs

Out-of-line publishing constructs

Chapter 8 - Floats and footnotes

Floats and footnotes render information supplemental to the information found in the flow:

- information is easily found by a reader because of a predetermined location on the page
  - defined "in line" of the flow of information being paginated
  - rendered "out-of-line" of the flow of information being paginated
  - considered auxiliary enough not to disturb the flow itself for the reader
    - reader can choose to examine a float or footnote at leisure without interrupting the reading of the flow
- dynamically rendered on the page where detected by the formatter in the flow
  - can only be defined in the flow filling the body region of a page
    - also cannot be defined in an absolutely positioned area
  - it is stacked in a different reference area than the main reference area
- floats are rendered to either the before, start, or end edges within the body region
  - not in the perimeter regions
- footnotes are two-part constructs:
  - footnote citation rendered inline in the flow
  - footnote body rendered at the after edge of the body region
  - an after float is accomplished using a footnote without a footnote citation
    - only when footnotes are not already being used

Footnote labeling is the responsibility of transformation

- the label of a footnote reference is supplied in the instance and not generated by the formatter
- restarting the footnote referencing on a per-page basis is not available in XSL-FO 1.0

No endnote constructs in XSL-FO

- responsibility of transformation to implement endnote functionality
- cite endnotes inline in the flow of the scope
- collect and render endnotes at the end of scope
  - can use empty citations at end to take advantage of XSL-FO footnote construct

Out-of-line publishing constructs (cont.)

Using floats allows the normal flow to stack in the body without interruption

- floats stack at the perimeter edges of the page
- can prevent large gaps in the normal flow when large constructs would otherwise jump to a new page

Many candidate uses of floating constructs

- using `<float>` either within a block or between blocks:
  - floating images to the top of a page
  - side-bar presentations
  - lists with item bodies indented relative to the corresponding item label's formatted length
  - a multi-line drop initial cap in a paragraph
- using `<footnote>`allowed only within a block:
  - traditional footnotes
  - acronym expansions
  - glossary definitions
  - sinking images to the bottom of a page (using an empty inline construct)
  - trailing disclaimer at the bottom of the last page of a rendered document

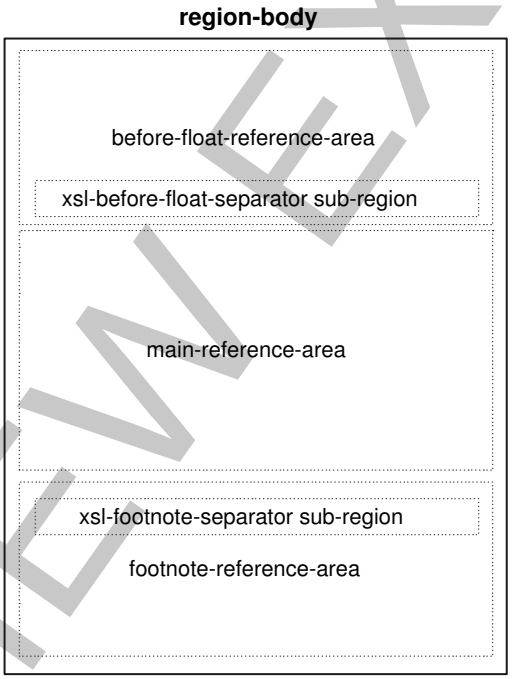
The name of the construct shouldn't prejudice how the construct is used

- consider the need mentioned above to format a disclaimer at the bottom of the last page of a document
- one can flow the disclaimer in the body of a footnote with an empty citation in an empty block at the end of the document

Out-of-line publishing constructs (cont.)

Every page's body region has two sub-regions rendered only if necessary

- before-floats and footnotes are stacked in the body region with other block-level constructs
  - the reader needs some separation rendered to distinguish content belonging in a float or footnote from the content belonging in the body
  - the body region is separated into the before-float-reference-area, main-reference-area, and footnote-reference-area
- the act of defining the separators does not effect their rendering
  - defined using static content
  - only rendered on a page if the floated information is being rendered on the page
  - `xsl-before-float-separator`
    - inside and at the end of the before-float-reference-area
  - `xsl-footnote-separator`
    - inside and at the start of the footnote-reference-area
  - should always be defined as a contingency if floats and footnotes are being used



Remember from Page geometry (page 73) the incursion of perimeter regions into the body if the body margins are not set accordingly

- all reference areas shown above are within the body-region's margins

Out-of-line publishing constructs (cont.)

The XSL-FO objects covered in this chapter are:

- `<float>` (6.10.2)
  - content that is to be rendered towards either the before, start, or end edges of the body region regardless of where in the region the content is defined
- `<footnote>` (6.10.3)
  - content that is to be rendered both in the flow and towards the after-edge of the body region regardless of where in the region the content is defined
- `<footnote-body>` (6.10.4)
  - the portion of footnote content rendered towards the after-edge of the body region

Chapter 9 - Breaks, keeps, spacing, borders and backgrounds

- Introduction - Spacing and arrangement constraint definition

Outcomes:

- understand basic concepts of spacing and stacking

Spacing and arrangement constraint definition

An area's placement is governed by XSL-FO stacking rules and the area's traits refined from object properties

- at the block level in a page
  - before-float-reference-area
  - normal-flow-reference-areas (page columns)
  - footnote-reference-area
- at the line level in a block
  - lines are generated by the formatter, not the stylesheet
- at an inline-level in a line
  - characters, graphics, etc.

Areas that are stacked normally are stacked in the pertinent progression direction

- page-level reference areas stack in the block-progression direction
- lines and table rows stack in the block-progression direction
- page column, table column and inline areas stack in the inline-progression direction

The "natural" stacking of areas may produce typographically unpleasant results

- initial values implement common-sense formatting control for consistent presentation
- many traditional conventions break the behavior of the initial values
  - e.g. needing to keep a heading in the body on the same page as the first paragraph to which the heading applies when the naturally occurring page break would otherwise come between the two items
  - necessary sometimes to override the physical arrangement of information implied by the initial values for area properties

Spacing and arrangement constraint definition (cont.)

Conditionality and precedence can eliminate areas from being rendered

- can prevent the unnecessary use of inter-block spacing when pagination renders a block without an adjacent sibling
  - e.g. a block forced to the top of a new page doesn't always need the space defined for between blocks to be rendered
- formatting special cases can be accommodated with simple specifications of intent
  - the formatter determine the applicability of spaces based on object properties
  - the transformation process can ascribe properties easier than determining space behaviors

Numerous properties are available to specify these nuances of layout

- arbitrarily breaking the flow of content to a new column or page
- maintaining a minimum number of widow and orphan lines of a block on a page
- keeping information together in the same reference area
- drawing borders around information
- painting backgrounds behind information

Chapter 10 - Interactive objects

- Introduction - Dynamic and interactive object rendering

Outcomes:

- understand basic concepts of interactive objects

Dynamic and interactive object rendering

Chapter 10 - Interactive objects

An electronic canvas can offer an active presentation of content

- operator interaction can influence presentation
  - the use of interactive formatting objects can equip the operator to present the content as desired from multiple alternatives
  - the state of an interactive object can be reflected based on previous interaction or lack thereof
- dynamically changing presentation
  - not restricted to a single static rendering of the information

Two areas where interactivity can influence presentation

- reflecting the active state of a linked object using different property values
  - a link can be but hasn't yet been traversed (future potential for visitation)
  - a link would be traversed (under a hovering pointer)
  - a link that is about to be traversed (has the user interface focus)
  - a link that is in the process of being traversed (is active)
  - a link has been traversed (past visitation)
- selecting and switching between alternate available presentations using different sub-trees of formatting objects
  - e.g. a dynamically expandable and collapsible table of contents rendering
  - HTML pages accomplish this in an imperative fashion with scripting in a programming language
  - XSL-FO 1.0 pages accomplish this in a declarative fashion by describing transition paths in a state machine, without having to implement the imperative logic behind the state machine

Dynamic and interactive object rendering (cont.)

Formatting objects related to dynamic properties are:

- `<multi-properties>` (6.9.6)
  - the collection of candidate property sets from which exactly one set influences the properties of a formatting object based on its status or the status of operator interaction
- `<multi-property-set>` (6.9.7)
  - the set of properties associated with a single possible state of a formatting object or operator interaction

Formatting objects related to dynamic presentation are:

- `<multi-switch>` (6.9.3)
  - the collection of candidate formatting object sequences from which exactly one is rendered at any given time based on an interactive condition that is influenced by the operator while being tracked by the formatter
- `<multi-case>` (6.9.4)
  - a single formatting object sequence that is a candidate for rendering based on an interactive condition that is influenced by the operator while being tracked by the formatter
- `<multi-toggle>` (6.9.5)
  - the definition of those interaction-sensitive objects within a candidate rendered sequence of formatting objects

Chapter 11 - Supplemental objects

- Introduction - Specialty constructs

Outcomes:

- understand basic concepts of lesser-used constructs

Specialty constructs

Chapter 11 - Supplemental objects

The objects described here are not typically used very often but can provide useful functionality to those with specific formatting requirements in two areas:

- character-level processing outside of the default behaviors
  - managing the bi-directionality inherent in the Unicode properties of characters
  - instantiating formatting objects explicitly in the formatting object tree
- constructs of a global nature
  - specifying a color system recognized by an XSL-FO processor
  - accommodating a home in the XSL-FO instance for color system specifications

Fifth Edition - 2004-02-12 - ISBN 1-894049-13-6

Specialty constructs (cont.)

The XSL-FO objects covered in this chapter are:

- `<bidi-override>` (6.6.2)
  - specifies how to manage and override the inherent Unicode text direction for a sequence of characters
- `<character>` (6.6.3)
  - both the abstract formatting object implied by a simple character in an XSL-FO instance and the concrete formatting object available to be used in place of a simple character
- `<color-profile>` (6.4.4)
  - the declaration of a profile of candidate color values from which color specifications can be made by formatting objects
- `<declarations>` (6.4.3)
  - a global-scope repository of formatting object constructs for an XSL-FO instance
    - the collection of available color profiles
    - any collections of extended formatting objects supported by an XSL-FO processor
      - must use a processor-recognized namespace URI other than the XSL-FO namespace URI

Chapter 12 - Where XSL-FO 1.0 falls short

- Introduction - What is missing in XSL-FO 1.0?

What is missing in XSL-FO 1.0?

Chapter 12 - Where XSL-FO 1.0 falls short

Important note: this section is conjecture on the part of the author and does not represent any official (or even unofficial) comments from the development committee.

Of course the designers of XSL-FO 1.0 could not put everything into the specification and ever consider their work "complete", but what is available is very useful

- measured amount of functionality to ensure successful implementation and deployment
- need real-world experience with first definition to ensure design principles are sound

Indications are in XSL-FO 1.0 of more complex functionality to follow

- the `<simple-page-master>` construct name implies perhaps a more complete page geometry definition is planned for the future
- property set collections of only a single property
    - if not planning for the future, why not just list the property instead of putting it alone in a set?

Some important functionality needed by compositors is left to a future version

- page-related formatting requirements
- geometry-related formatting requirements
- alignment of areas
- post-formatting requirements

Annex A - Using XSLT with XSL-FO

- Introduction - Using XSLT with XSL-FO
- Section 1 - XSLT language features supporting XSL-FO
- Section 2 - XSL-FO language features similar to XSLT and XPath

Using XSLT with XSL-FO

Annex A - Using XSLT with XSL-FO

XSLT was designed primarily for use with XSL-FO

- result of an XSLT transform is an XML information set that may or may not be serialized as XML syntax
  - result information set can be delivered to XSL-FO formatter without serialization
- a number of features of XSLT assist the writing of XSL-FO stylesheets
- the design of XSL-FO is made simpler by the use of XSLT functionality

XSLT is a normative part of XSL

- section 2.1 of XSL states the following:
  - The provisions in "XSL Transformations" form an integral part of this recommendation and are considered normative.

Certain aspects of XSL-FO are defined by XSLT

- references are made in XSL-FO to sections of XSLT for their definition
- design decisions in XSL-FO pattern themselves after XSLT definitions

No technical reason that XSLT must be used to create XSL-FO

- the formatter can accept an instance of XSL-FO regardless of how that instance was generated

<xsl:attribute-set> instruction

<xsl:attribute-set> is a very useful instruction for the manipulation of many attributes targeted for a given formatting object:

- names a set of attribute instructions that can be called on demand
- uses a namespace-qualified name promoting the easy sharing of stylesheet fragments
- well-defined integration with other methods of specifying attributes for a result element
  - any <xsl:attribute-set> collections named in an xsl:use-attribute-sets= attribute are added first
  - any attribute specifications in the literal result element itself are added next
  - any executed <xsl:attribute> instructions in the literal result element's template are added last
  - earlier defined attribute values are replaced with later defined values without an error
  - last value assigned to an attribute is what remains in the result tree

Simpler list and footnote structures in XSL-FO

The numbering facilities in XSLT allow the list structures in XSL-FO to be more simply defined:

- the collection of list formatting objects are layout oriented, not content oriented
    - a list is a list regardless of how it is labeled or what it contains
- the content of the objects is defined by XSLT, thereby reducing number of objects
    - no distinction between numbered and unnumbered lists
    - the type and structure of the list item labels is entirely out of the scope of the formatting and rendering

The semantics of footnote citation numbering are not part of XSL-FO

- the value of the footnote citation is determined by the transformation process, not by the formatting process
    - a drawback is that this prevents footnote numbering from being page-based

Common errors writing expressions

The expression language is very close, but not identical:

- same operators
- wider set of operands in XSL-FO than XPath

Nuances of differences can make the writing of XSLT confusing

- triggers stylesheet errors

Common errors writing expressions (cont.)

Consider the situation of arithmetic calculations with lengths:

```
<xsl:template name="test">
<xsl:variable name="num" select="100"/>
<xsl:variable name="medium-font" select="'10pt'"/>

<block space-before="10pt div 2">
First test: <xsl:value-of select="$num div 2"/>
</block>

<block space-before="{$medium-font} div 2">
Second test
</block>

<block space-before="{$medium-font div 2}">
Third test
</block>

<block>
  <xsl:attribute name="space-before">
    <xsl:value-of select="$medium-font div 2"/>
  </xsl:attribute>
  Fourth test
</block>

<block>
  <xsl:attribute name="space-before">
    <xsl:value-of select="$medium-font"/> div 2<xsl:text/>
  </xsl:attribute>
  Fifth test
</block>
</xsl:template>
```

There will be no spacing for the third and fourth tests above

- the arithmetic calculation yields the `NaN` result because a length is not a number in XSLT as in XSL-FO
  - XSLT cannot do arithmetic operations with string operands
- a formatter may not report an error and assume a value of 0 for `NaN` since it is a valid number in the numbering system
  - no indication to the stylesheet writer that anything is wrong, yet results are not as expected and no error messages to diagnose

Annex B - XSL-FO expressions

- Introduction - Expressions in XSL-FO
- Section 1 - XSL-FO expressions
- Section 2 - XSL-FO functions

Fifth Edition - 2004-02-12 - ISBN 1-894049-13-6

Expressions in XSL-FO

Annex B - XSL-FO expressions

This annex first summarizes in production order all of the productions in the expression grammar in the Recommendation:

- the left-hand side is the production being defined
- the right-hand side is the definition of the grammatical construct

Secondly, this annex summarizes the built-in functions available in the XSL-FO processor.

Production summary

Annex B - XSL-FO expressions
Section 1 - XSL-FO expressions

```
[1] Expr            ::= AdditiveExpr[11]
[2] PrimaryExpr     ::= '(' Expr[1] ')'
                      | Numeric[5]
                      | Literal[20]
                      | Color[18]
                      | Keyword[24]
                      | EnumerationToken[26]
                      | FunctionCall[3]
[3] FunctionCall    ::= FunctionName[25] '(' ( Argument[4] ( ',' Argument[4])*)?
                      ')'
[4] Argument        ::= Expr[1]
[5] Numeric         ::= AbsoluteNumeric[6]
                      | RelativeNumeric[8]
[6] AbsoluteNumeric ::= AbsoluteLength[7]
[7] AbsoluteLength  ::= Number[15] AbsoluteUnitName[27]?
[8] RelativeNumeric ::= Percent[9]
                      | RelativeLength[10]
[9] Percent         ::= Number[15] '%'
[10] RelativeLength ::= Number[15] RelativeUnitName[27]
[11] AdditiveExpr   ::= MultiplicativeExpr[12]
                      | AdditiveExpr[11] '+' MultiplicativeExpr[12]
                      | AdditiveExpr[11] '-' MultiplicativeExpr[12]
[12] MultiplicativeExpr ::= UnaryExpr[13]
                      | MultiplicativeExpr[12] MultiplyOperator[23]
                      UnaryExpr[13]
                      | MultiplicativeExpr[12] 'div' UnaryExpr[13]
                      | MultiplicativeExpr[12] 'mod' UnaryExpr[13]
[13] UnaryExpr      ::= PrimaryExpr[2]
                      | '-' UnaryExpr[13]
[14] ExprToken      ::= '(' | ')' | '%'
                      | Operator[21]
                      | FunctionName[25]
                      | EnumerationToken[26]
                      | Number[15]
[15] Number         ::= FloatingPointNumber[16]
[16] FloatingPointNumber ::= Digits[17] ('.' Digits[17]?)?
                      | '.' Digits[17]
[17] Digits         ::= [0-9]+
[18] Color          ::= '#' AlphaOrDigits[19]
[19] AlphaOrDigits  ::= [a-fA-F0-9]+
[20] Literal        ::= '"' [^"]* '"'
                      | "'" [^']* "'"
```

```
[21] Operator             ::= OperatorName[22]
                              | MultiplyOperator[23]
                              | '+' | '-'
[22] OperatorName         ::= 'mod' | 'div'
[23] MultiplyOperator     ::= '*'
[24] Keyword              ::= 'inherit'
[25] FunctionName         ::= NCName(XML Namespaces)
[26] EnumerationToken     ::= NCName(XML Namespaces)
[27] AbsoluteUnitName     ::= 'cm' | 'mm' | 'in' | 'pt' | 'pc' | 'px'
[28] RelativeUnitName     ::= 'em'
[29] ExprWhitespace       ::= S(XML)
```


Function summary

Annex B - XSL-FO expressions
Section 2 - XSL-FO functions

Expressions can incorporate numbers and lengths as operands

- a number does not include a unit of measure
  - termed "a unit power of zero"
- a length does include a unit of measure (inches, mm, etc.)
  - termed "a unit power of one"

Performing numeric operations on lengths

- some functions accept either lengths or numbers as operands
  - e.g. `font-size="min(10pt,1em)"`
  - e.g. `<table-cell column-number="min(3,5)">`
  - sometimes (as in `min()`) both operands must have the same unit power
- some functions do not work on lengths, only numbers
  - e.g. `round()` accepts only numbers, not lengths
    - to round a length: convert length to a number, round, then convert result to a length
    - e.g. `round(.75in div 1in)*1in`

Percentages are counted in 1/100 units

- can be utilized in a property as a relation to the property's current value
  - the following expressions all evaluate to the same value:
    - `font-size="150%"`
    - `font-size="1.5 * inherited-property-value(font-size)"`
    - `font-size="1.5em"`

Function groupings

Annex B - XSL-FO expressions
Section 2 - XSL-FO functions

Color Functions (5.10.2)

- `rgb()`
- `rgb-icc()`
- `system-color()`

Font Functions (5.10.3)

- `system-font()`

Number Functions (5.10.1)

- `abs()`
- `ceiling()`
- `floor()`
- `max()`
- `min()`
- `round()`

Property Value Functions (5.10.4)

- `body-start()`
- `from-nearest-specified-value()`
- `from-parent()`
- `from-table-column()`
- `inherited-property-value()`
- `label-end()`
- `merge-property-values()`
- `proportional-column-width()`

Functions summarized by name

Annex B - XSL-FO expressions
Section 2 - XSL-FO functions

numeric `abs(`*numeric*`)` (5.10.1)

- returns absolute value of number or length argument

numeric `body-start()` (5.10.4)

- returns the start indent corresponding to the list-item body when considering the provisional-valued properties of the list

numeric `ceiling(`*numeric*`)` (5.10.1)

- returns number closest to positive infinity

numeric `floor(`*numeric*`)` (5.10.1)

- returns number closest to negative infinity

object `from-nearest-specified-value(`*NCName*`)` (5.10.4)

- returns the value of the named property from the nearest ancestor of the formatting object in which the property value is being explicitly specified
- returns the initial value if there is no parent

object `from-parent(`*NCName*`)` (5.10.4)

- returns the value of the named property from the parent of the formatting object of the property being evaluated
- returns the initial value if there is no parent

object `from-table-column(`*NCName*`)` (5.10.4)

- returns the value of the named property from the `<table-column>` corresponding to the current column in the table

object `inherited-property-value(`*NCName*`)` (5.10.4)

- returns the value of the inherited property named
- it is an error if the property named is not an inherited property

numeric `label-end()` (5.10.4)

- returns the end indent corresponding to the list-item label when considering the provisional-valued properties of the list

numeric `max(`*numeric*`, `*numeric*`)` (5.10.1)

- returns the minimum of two arguments
- arguments may be either lengths or numbers but must be the same type

object `merge-property-values(`*NCName*`)` (5.10.4)

- returns the property calculated from the property set corresponding to the current user-agent state within the parent's child property sets

numeric `min(`*numeric*`, `*numeric*`)` (5.10.1)

- returns the minimum of two arguments
- arguments may be either lengths or numbers but must be the same type

numeric `proportional-column-width(`*numeric*`)` (5.10.4)

- returns the length corresponding to the number of units of proportional measure of the current table column's table as indicated in the supplied argument
- note that proportional measure is that length left over when removing specified column widths from the table width and dividing the result by the number of columns for which widths are not specified

color `rgb(`*numeric*`, `*numeric*`, `*numeric*`)` (5.10.2)

- returns a color from the RGB space
- arguments must be numbers, not lengths

color `rgb-icc(`*numeric*`, `*numeric*`, `*numeric*`, `*NCName*`, `*numeric*`, `*numeric*`)` (5.10.2)

- returns a color from the named ICC color profile
- the first three arguments are the fallback RGB color if the ICC color is not found
- the last arguments are specific to the color profile

numeric `round(`*numeric*`)` (5.10.1)

- returns whole number closest to given number
- returns `ceiling()` on the value .5

color `system-color(`*NCName*`)` (5.10.2)

- returns the system-defined color named in the argument

object `system-font(`*NCName*`, `*NCName*`)` (5.10.3)

- returns the font-size characteristic named in the second argument of the system font named in the first argument
- if the second argument is omitted, the characteristic is that which is being assigned by the expression

Annex C - XSL-FO object summary

- Introduction - Objects
- Section 1 - Formatting objects

Objects

Annex C - XSL-FO object summary

This annex first summarizes in alphabetical order all of the formatting objects in the Recommendation:

- the citation on the first line is to the section of the W3C Recommendation document
- the citation on the second line is to the section of this material

Secondly, this annex summarizes the objects by their type as categorized by the Recommendation:

- the citation on each subordinate line is to the section of this material

Lastly, a prototypical arrangement of all formatting objects is shown to indicate the nesting of child objects inside of expected or typical parent objects.

Fifth Edition - 2004-02-12 - ISBN 1-894049-13-6

Objects summarized by name

Annex C - XSL-FO object summary
Section 1 - Formatting objects

`<basic-link>` (6.9.2; 81) (visual: extended; aural: extended)
`<bidi-override>` (6.6.2; 112) (visual: extended; aural: basic)
`<block>` (6.5.2; 75) (visual: basic; aural: basic)
`<block-container>` (6.5.3; 75) (visual: extended; aural: basic)
`<character>` (6.6.3; 112) (visual: basic; aural: basic)
`<color-profile>` (6.4.4; 112) (visual: extended; aural: N/A)
`<conditional-page-master-reference>` (6.4.11; 93) (visual: extended; aural: extended)
`<declarations>` (6.4.3; 112) (visual: basic; aural: basic)
`<external-graphic>` (6.6.5; 81) (visual: basic; aural: basic)
`<float>` (6.10.2; 103) (visual: extended; aural: extended)
`<flow>` (6.4.18; 62) (visual: basic; aural: basic)
`<footnote>` (6.10.3; 103) (visual: extended; aural: extended)
`<footnote-body>` (6.10.4; 103) (visual: extended; aural: extended)
`<initial-property-set>` (6.6.4; 75) (visual: extended; aural: basic)
`<inline>` (6.6.7; 75) (visual: basic; aural: basic)
`<inline-container>` (6.6.8; 75) (visual: extended; aural: extended)
`<instream-foreign-object>` (6.6.6; 81) (visual: extended; aural: extended)
`<layout-master-set>` (6.4.6; 58) (visual: basic; aural: basic)
`<leader>` (6.6.9; 81) (visual: basic; aural: basic)
`<list-block>` (6.8.2; 81) (visual: basic; aural: basic)
`<list-item>` (6.8.3; 81) (visual: basic; aural: basic)
`<list-item-body>` (6.8.4; 81) (visual: basic; aural: basic)
`<list-item-label>` (6.8.5; 81) (visual: extended; aural: basic)
`<marker>` (6.11.3; 92) (visual: extended; aural: extended)
`<multi-case>` (6.9.4; 109) (visual: basic; aural: basic)
`<multi-properties>` (6.9.6; 109) (visual: extended, need not be implemented for extended conformance for non-interactive media; aural: extended)
`<multi-property-set>` (6.9.7; 109) (visual: extended, need not be implemented for extended conformance for non-interactive media; aural: extended)
`<multi-switch>` (6.9.3; 109) (visual: extended, need not be implemented for extended conformance for non-interactive media; aural: extended)
`<multi-toggle>` (6.9.5; 109) (visual: extended, need not be implemented for extended conformance for non-interactive media; aural: extended)
`<page-number>` (6.6.10; 92) (visual: basic; aural: extended)
`<page-number-citation>` (6.6.11; 75) (visual: extended; aural: extended)
`<page-sequence>` (6.4.5; 60) (visual: basic; aural: basic)
`<page-sequence-master>` (6.4.7; 93) (visual: basic; aural: basic)
`<region-after>` (6.4.15; 92) (visual: extended; aural: extended)
`<region-before>` (6.4.14; 92) (visual: extended; aural: extended)
`<region-body>` (6.4.13; 75) (visual: basic; aural: basic)
`<region-end>` (6.4.17; 92) (visual: extended; aural: extended)
`<region-start>` (6.4.16; 92) (visual: extended; aural: extended)

Prototypical hierarchy

Annex C - XSL-FO object summary
Section 1 - Formatting objects

All formatting objects are listed below in a typical nesting of parents and children. Where sibling children order is not significant, an ellipsis is shown between them.

- <root> (6.4.2; 56)
    - <layout-master-set> (6.4.6; 58)
        - <simple-page-master> (6.4.12; 75)
            - <region-body/> (6.4.13; 75)
            - <region-before/> (6.4.14; 92)
            - <region-after/> (6.4.15; 92)
            - <region-start/> (6.4.16; 92)
            - <region-end/> (6.4.17; 92)
        - ...
        - <page-sequence-master> (6.4.7; 93)
            - <single-page-master-reference/> (6.4.8; 93)
            - ...
            - <repeatable-page-master-reference/> (6.4.9; 93)
            - ...
            - <repeatable-page-master-alternatives> (6.4.10; 93)
                - <conditional-page-master-reference/> (6.4.11; 93)
    - <declarations> (6.4.3; 112)
    - <page-sequence> (6.4.5; 60)
        - <title> (6.4.20; 75)
        - <static-content> (6.4.19; 92)
            - <retrieve-marker/> (6.11.4; 92)
            - ...
            - <page-number/> (6.6.10; 92)
        - <flow> (6.4.18; 62)
            - <block-container> (6.5.3; 75)
            - ...
            - <float> (6.10.2; 103)
            - ...
            - <table-and-caption> (6.7.2; 87)
                - <table-caption> (6.7.5; 87)
                - <table> (6.7.3; 87)
                    - <table-column/> (6.7.4; 87)
                    - <table-header> (6.7.6; 87)
                    - <table-footer> (6.7.7; 87)
                    - <table-body> (6.7.8; 87)
                        - <table-row> (6.7.9; 87)
                        - <table-cell> (6.7.10; 87)
            - ...
            - <list-block> (6.8.2; 81)
                - <list-item> (6.8.3; 81)
                    - <list-item-label> (6.8.5; 81)
                    - <list-item-body> (6.8.4; 81)
            - ...
            - <block> (6.5.2; 75)
                - <marker> (6.11.3; 92)
                - <initial-property-set/> (6.6.4; 75)

- `<inline-container>` (6.6.8; 75)
- ...
- `<basic-link>` (6.9.2; 81)
- ...
- `<bidi-override>` (6.6.2; 112)
- ...
- `<character/>` (6.6.3; 112)
- ...
- `<color-profile/>` (6.4.4; 112)
- ...
- `<external-graphic/>` (6.6.5; 81)
- ...
- `<footnote>` (6.10.3; 103)
    - `<footnote-body>` (6.10.4; 103)
- ...
- `<inline>` (6.6.7; 75)
- ...
- `<instream-foreign-object>` (6.6.6; 81)
- ...
- `<leader>` (6.6.9; 81)
- ...
- `<page-number-citation/>` (6.6.11; 75)
- ...
- `<wrapper>` (6.11.2; 75)
- ...
- `<multi-switch>` (6.9.3; 109)
    - `<multi-case>` (6.9.4; 109)
        - `<multi-toggle>` (6.9.5; 109)
- ...
- `<multi-properties>` (6.9.6; 109)
    - `<multi-property-set/>` (6.9.7; 109)

Objects summarized by type

Annex C - XSL-FO object summary
Section 1 - Formatting objects

Block-level Formatting Objects (6.5)

- `<block>` (6.5.2; 75)
- `<block-container>` (6.5.3; 75)

Declarations and Pagination and Layout Formatting Objects (6.4)

- `<color-profile>` (6.4.4; 112)
- `<conditional-page-master-reference>` (6.4.11; 93)
- `<declarations>` (6.4.3; 112)
- `<flow>` (6.4.18; 62)
- `<layout-master-set>` (6.4.6; 58)
- `<page-sequence>` (6.4.5; 60)
- `<page-sequence-master>` (6.4.7; 93)
- `<region-after>` (6.4.15; 92)
- `<region-before>` (6.4.14; 92)
- `<region-body>` (6.4.13; 75)
- `<region-end>` (6.4.17; 92)
- `<region-start>` (6.4.16; 92)
- `<repeatable-page-master-alternatives>` (6.4.10; 93)
- `<repeatable-page-master-reference>` (6.4.9; 93)
- `<root>` (6.4.2; 56)
- `<simple-page-master>` (6.4.12; 75)
- `<single-page-master-reference>` (6.4.8; 93)
- `<static-content>` (6.4.19; 92)
- `<title>` (6.4.20; 75)

Dynamic Effects: Link and Multi Formatting Objects (6.9)

- `<basic-link>` (6.9.2; 81)
- `<multi-case>` (6.9.4; 109)
- `<multi-properties>` (6.9.6; 109)
- `<multi-property-set>` (6.9.7; 109)
- `<multi-switch>` (6.9.3; 109)
- `<multi-toggle>` (6.9.5; 109)

Formatting Objects for Lists (6.8)

- `<list-block>` (6.8.2; 81)
- `<list-item>` (6.8.3; 81)
- `<list-item-body>` (6.8.4; 81)
- `<list-item-label>` (6.8.5; 81)

Formatting Objects for Tables (6.7)

- `<table-and-caption>` (6.7.2; 87)
- `<table>` (6.7.3; 87)
- `<table-body>` (6.7.8; 87)
- `<table-caption>` (6.7.5; 87)
- `<table-cell>` (6.7.10; 87)
- `<table-column>` (6.7.4; 87)
- `<table-footer>` (6.7.7; 87)
- `<table-header>` (6.7.6; 87)
- `<table-row>` (6.7.9; 87)

Inline-level Formatting Objects (6.6)

- `<bidi-override>` (6.6.2; 112)
- `<character>` (6.6.3; 112)
- `<external-graphic>` (6.6.5; 81)
- `<initial-property-set>` (6.6.4; 75)
- `<inline>` (6.6.7; 75)
- `<inline-container>` (6.6.8; 75)
- `<instream-foreign-object>` (6.6.6; 81)
- `<leader>` (6.6.9; 81)
- `<page-number>` (6.6.10; 92)
- `<page-number-citation>` (6.6.11; 75)

Other Formatting Objects (6.11)

- `<marker>` (6.11.3; 92)
- `<retrieve-marker>` (6.11.4; 92)
- `<wrapper>` (6.11.2; 75)

Out-of-Line Formatting Objects (6.10)

- `<float>` (6.10.2; 103)
- `<footnote>` (6.10.3; 103)
- `<footnote-body>` (6.10.4; 103)

Annex D - XSL-FO property summaries

- Introduction - Properties
- Section 1 - Property groupings
- Section 2 - Data types
- Section 3 - Property summaries

Properties

Annex D - XSL-FO property summaries

This annex first summarizes the collections of common properties.

This annex then summarizes all the data types of the properties.

This annex then summarizes all shorthand and individual properties each in alphabetical order:

- the citation is to the W3C Recommendation
- after the citation is either "CSS" or "XSL" indicating the heritage of this property

Note that much of this annex is synthesized from the application of XSLT to the source XML of the W3C Recommendation document itself, demonstrating the power of using these technologies for information description.

Common Absolute Position Properties

Properties (7.5):

`absolute-position=`(7.5.1; 162)
`bottom=`(7.5.4; 176)
`left=`(7.5.5; 190)
`right=`(7.5.3; 204)
`top=`(7.5.2; 212)

Referring object:

`<block-container>` (6.5.3; 75)

Shorthand influencing the above properties:

`position=`(7.29.20; 201)

Common Accessibility Properties

Annex D - XSL-FO property summaries
Section 1 - Property groupings

Properties (7.4):

`role=`(7.4.2; 204)

`source-document=`(7.4.1; 206)

Referring objects:

`<basic-link>` (6.9.2; 81)
`<block>` (6.5.2; 75)
`<external-graphic>` (6.6.5; 81)
`<footnote>` (6.10.3; 103)
`<footnote-body>` (6.10.4; 103)
`<initial-property-set>` (6.6.4; 75)
`<inline>` (6.6.7; 75)
`<instream-foreign-object>` (6.6.6; 81)
`<leader>` (6.6.9; 81)
`<list-block>` (6.8.2; 81)
`<list-item>` (6.8.3; 81)
`<list-item-body>` (6.8.4; 81)
`<list-item-label>` (6.8.5; 81)
`<multi-case>` (6.9.4; 109)
`<multi-properties>` (6.9.6; 109)
`<multi-switch>` (6.9.3; 109)
`<multi-toggle>` (6.9.5; 109)
`<page-number>` (6.6.10; 92)
`<page-number-citation>` (6.6.11; 75)
`<table-and-caption>` (6.7.2; 87)
`<table>` (6.7.3; 87)
`<table-body>` (6.7.8; 87)
`<table-caption>` (6.7.5; 87)
`<table-cell>` (6.7.10; 87)
`<table-footer>` (6.7.7; 87)
`<table-header>` (6.7.6; 87)
`<table-row>` (6.7.9; 87)
`<title>` (6.4.20; 75)

Fifth Edition - 2004-02-12 - ISBN 1-894049-13-6

Common Aural Properties

Annex D - XSL-FO property summaries
Section 1 - Property groupings

Properties (7.6):

`azimuth=`(7.6.1; 163)
`cue-after=`(7.6.2; 180)
`cue-before=`(7.6.3; 180)
`elevation=`(7.6.4; 181)
`pause-after=`(7.6.5; 200)
`pause-before=`(7.6.6; 200)
`pitch=`(7.6.7; 200)
`pitch-range=`(7.6.8; 200)
`play-during=`(7.6.9; 201)
`richness=`(7.6.10; 204)
`speak=`(7.6.11; 207)
`speak-header=`(7.6.12; 207)
`speak-numeral=`(7.6.13; 207)
`speak-punctuation=`(7.6.14; 208)
`speech-rate=`(7.6.15; 208)
`stress=`(7.6.16; 209)
`voice-family=`(7.6.17; 213)
`volume=`(7.6.18; 213)

Referring objects:

`<basic-link>` (6.9.2; 81)
`<bidi-override>` (6.6.2; 112)
`<block>` (6.5.2; 75)
`<character>` (6.6.3; 112)
`<external-graphic>` (6.6.5; 81)
`<initial-property-set>` (6.6.4; 75)
`<inline>` (6.6.7; 75)
`<instream-foreign-object>` (6.6.6; 81)
`<leader>` (6.6.9; 81)
`<list-block>` (6.8.2; 81)
`<list-item>` (6.8.3; 81)
`<page-number>` (6.6.10; 92)
`<page-number-citation>` (6.6.11; 75)
`<table-and-caption>` (6.7.2; 87)
`<table>` (6.7.3; 87)
`<table-body>` (6.7.8; 87)
`<table-caption>` (6.7.5; 87)
`<table-cell>` (6.7.10; 87)
`<table-footer>` (6.7.7; 87)
`<table-header>` (6.7.6; 87)
`<table-row>` (6.7.9; 87)
`<title>` (6.4.20; 75)

Shorthands influencing the above properties:

`cue=`(7.29.12; 179)
`pause=`(7.29.19; 199)

Common Border, Padding, and Background Properties

Properties (7.7):

`background-attachment=`(7.7.1; 164)
`background-color=`(7.7.2; 164)
`background-image=`(7.7.3; 164)
`background-position-horizontal=`(7.7.5; 165)
`background-position-vertical=`(7.7.6; 165)
`background-repeat=`(7.7.4; 166)
`border-after-color=`(7.7.10; 167)
`border-after-style=`(7.7.11; 167)
`border-after-width=`(7.7.12; 168)
`border-before-color=`(7.7.7; 168)
`border-before-style=`(7.7.8; 168)
`border-before-width=`(7.7.9; 169)
`border-bottom-color=`(7.7.22; 169)
`border-bottom-style=`(7.7.23; 169)
`border-bottom-width=`(7.7.24; 170)
`border-end-color=`(7.7.16; 170)
`border-end-style=`(7.7.17; 171)
`border-end-width=`(7.7.18; 171)
`border-left-color=`(7.7.25; 172)
`border-left-style=`(7.7.26; 172)
`border-left-width=`(7.7.27; 172)
`border-right-color=`(7.7.28; 172)
`border-right-style=`(7.7.29; 173)
`border-right-width=`(7.7.30; 173)
`border-start-color=`(7.7.13; 174)
`border-start-style=`(7.7.14; 174)
`border-start-width=`(7.7.15; 174)
`border-top-color=`(7.7.19; 175)
`border-top-style=`(7.7.20; 175)
`border-top-width=`(7.7.21; 176)
`padding-after=`(7.7.32; 196)
`padding-before=`(7.7.31; 196)
`padding-bottom=`(7.7.36; 196)
`padding-end=`(7.7.34; 197)
`padding-left=`(7.7.37; 197)
`padding-right=`(7.7.38; 197)
`padding-start=`(7.7.33; 198)
`padding-top=`(7.7.35; 198)

Referring objects:

`<basic-link>` (6.9.2; 81)
`<block>` (6.5.2; 75)
`<block-container>` (6.5.3; 75)
`<character>` (6.6.3; 112)
`<external-graphic>` (6.6.5; 81)
`<initial-property-set>` (6.6.4; 75)
`<inline>` (6.6.7; 75)
`<inline-container>` (6.6.8; 75)
`<instream-foreign-object>` (6.6.6; 81)
`<leader>` (6.6.9; 81)
`<list-block>` (6.8.2; 81)
`<list-item>` (6.8.3; 81)
`<page-number>` (6.6.10; 92)
`<page-number-citation>` (6.6.11; 75)
`<region-after>` (6.4.15; 92)
`<region-before>` (6.4.14; 92)
`<region-body>` (6.4.13; 75)
`<region-end>` (6.4.17; 92)
`<region-start>` (6.4.16; 92)
`<table-and-caption>` (6.7.2; 87)
`<table>` (6.7.3; 87)
`<table-body>` (6.7.8; 87)
`<table-caption>` (6.7.5; 87)
`<table-cell>` (6.7.10; 87)
`<table-column>` (6.7.4; 87)
`<table-footer>` (6.7.7; 87)
`<table-header>` (6.7.6; 87)
`<table-row>` (6.7.9; 87)
`<title>` (6.4.20; 75)

Shorthands influencing the above properties:

`background=`(7.29.1; 164)
`background-position=`(7.29.2; 165)
`border-right=`(7.29.7; 172)
`border-style=`(7.29.8; 175)

Common Font Properties

Annex D - XSL-FO property summaries
Section 1 - Property groupings

Properties (7.8):

`font-family=`(7.8.2; 183)
`font-selection-strategy=`(7.8.3; 183)
`font-size=`(7.8.4; 183)
`font-size-adjust=`(7.8.6; 184)
`font-stretch=`(7.8.5; 184)
`font-style=`(7.8.7; 184)
`font-variant=`(7.8.8; 184)
`font-weight=`(7.8.9; 185)

Referring objects:

`<bidi-override>` (6.6.2; 112)
`<block>` (6.5.2; 75)
`<character>` (6.6.3; 112)
`<initial-property-set>` (6.6.4; 75)
`<inline>` (6.6.7; 75)
`<leader>` (6.6.9; 81)
`<page-number>` (6.6.10; 92)
`<page-number-citation>` (6.6.11; 75)
`<title>` (6.4.20; 75)

Shorthand influencing the above properties:

`font=`(7.29.13; 183)

Common Hyphenation Properties

Annex D - XSL-FO property summaries
Section 1 - Property groupings

Properties (7.9):

`country=`(7.9.1; 179)
`hyphenate=`(7.9.4; 186)
`hyphenation-character=`(7.9.5; 186)
`hyphenation-push-character-count=`(7.9.6; 187)
`hyphenation-remain-character-count=`(7.9.7; 187)
`language=`(7.9.2; 189)
`script=`(7.9.3; 205)

Referring objects:

`<block>` (6.5.2; 75)
`<character>` (6.6.3; 112)

Shorthand influencing the above properties:

`xml:lang=`(7.29.24; 215)

Common Margin Properties-Block

Annex D - XSL-FO property summaries
Section 1 - Property groupings

Properties (7.10):

`end-indent=`(7.10.8; 182)
`margin-bottom=`(7.10.2; 192)
`margin-left=`(7.10.3; 192)
`margin-right=`(7.10.4; 192)
`margin-top=`(7.10.1; 193)
`space-after=`(7.10.6; 206)
`space-before=`(7.10.5; 206)
`start-indent=`(7.10.7; 208)

Referring objects:

`<block>` (6.5.2; 75)
`<block-container>` (6.5.3; 75)
`<list-block>` (6.8.2; 81)
`<list-item>` (6.8.3; 81)
`<region-body>` (6.4.13; 75)
`<simple-page-master>` (6.4.12; 75)
`<table-and-caption>` (6.7.2; 87)
`<table>` (6.7.3; 87)

Shorthand influencing the above properties:

`margin=`(7.29.14; 192)

Common Margin Properties-Inline

Annex D - XSL-FO property summaries
Section 1 - Property groupings

Properties (7.11):

`space-end=`(7.11.1; 206)
`space-start=`(7.11.2; 207)

Referring objects:

`<basic-link>` (6.9.2; 81)
`<character>` (6.6.3; 112)
`<external-graphic>` (6.6.5; 81)
`<inline>` (6.6.7; 75)
`<inline-container>` (6.6.8; 75)
`<instream-foreign-object>` (6.6.6; 81)
`<leader>` (6.6.9; 81)
`<page-number>` (6.6.10; 92)
`<page-number-citation>` (6.6.11; 75)
`<title>` (6.4.20; 75)

Common Relative Position Properties

Annex D - XSL-FO property summaries
Section 1 - Property groupings

Property (7.12):

`relative-position=`(7.12.1; 203)

Referring objects:

`<basic-link>` (6.9.2; 81)
`<bidi-override>` (6.6.2; 112)
`<block>` (6.5.2; 75)
`<character>` (6.6.3; 112)
`<external-graphic>` (6.6.5; 81)
`<initial-property-set>` (6.6.4; 75)
`<inline>` (6.6.7; 75)
`<inline-container>` (6.6.8; 75)
`<instream-foreign-object>` (6.6.6; 81)
`<leader>` (6.6.9; 81)
`<list-block>` (6.8.2; 81)
`<list-item>` (6.8.3; 81)
`<page-number>` (6.6.10; 92)
`<page-number-citation>` (6.6.11; 75)
`<table-and-caption>` (6.7.2; 87)
`<table>` (6.7.3; 87)
`<table-body>` (6.7.8; 87)
`<table-caption>` (6.7.5; 87)
`<table-cell>` (6.7.10; 87)
`<table-footer>` (6.7.7; 87)
`<table-header>` (6.7.6; 87)
`<table-row>` (6.7.9; 87)

Shorthand influencing the above property:

`position=`(7.29.20; 201)

Property data types

Annex D - XSL-FO property summaries
Section 2 - Data types

The typographical convention in the XSL-FO 1.0 Recommendation for data types is to use angle brackets around the data type name using a proportional font. This is not to be confused with XSL-FO elements that are also documented using angle brackets, but elements use a monospaced font. The Recommendation convention for this is used here for consistency for those readers referring to the W3C documentation.

A listing of data types:

<angle>

- A representation of an angle consisting of an optional '+' or '-' character immediately followed by a <number> immediately followed by an angle unit identifier. Angle unit identifiers are: 'deg' (for degrees), 'grad' (for grads), and 'rad' (for radians). The specified values are normalized to the range 0deg to 360deg. A property may define additional constraints on the value.

<character>

- A single Unicode character.

<color>

- Either a string of characters representing a keyword or a color function defined in Color Functions. The list of keyword color names is: aqua, black, blue, fuchsia, gray, green, lime, maroon, navy, olive, purple, red, silver, teal, white, and yellow.

<country>

- A string of characters conforming to an ISO 3166 country code.

<family-name>

- A string of characters identifying a font.

<frequency>

- A <number> immediately followed by a frequency unit identifier. Frequency unit identifiers are: 'Hz' (for Hertz) and 'kHz' (for kilo Hertz).

<id>

- A string of characters conforming to the definition of an NCName in XML Names and is unique within the stylesheet.

<idref>

- A string of characters conforming to the definition of an NCName in XML Names and that matches an ID property value used within the stylesheet.

<integer>

- A signed integer value which consists of an optional '+' or '-' character followed by a sequence of digits. A property may define additional constraints on the value.

<keep>

- A compound datatype, with components: within-line, within-column, and within-page. The value of each component is either "auto", "always", or an <integer>.

<language>

- A string of characters conforming to the ISO 639 3-letter code.

<length>

- A signed length value where a 'length' is a real number plus a unit qualification. A property may define additional constraints on the value.

<length-bp-ip-direction>

- A compound datatype, with components: block-progression-direction, and inline-progression-direction. Each component is a <length>. A property may define additional constraints on the values.

<length-conditional>

- A compound datatype, with components: length, conditionality. The length component is a <length>. The conditionality component is either "discard" or "retain". A property may define additional constraints on the values.

<length-range>

- A compound datatype, with components: minimum, optimum, maximum. Each component is a <length>. If "minimum" is greater than optimum, it will be treated as if it had been set to "optimum". If "maximum" is less than optimum, it will be treated as if it had been set to "optimum". A property may define additional constraints on the values.

<name>

- A string of characters representing a name. It must conform to the definition of an NCName in XML Names.

<number>

- A signed real number which consists of an optional '+' or '-' character followed by a sequence of digits followed by an optional '.' character and sequence of digits. A property may define additional constraints on the value.

<percentage>

- A signed real percentage which consists of an optional '+' or '-' character followed by a sequence of digits followed by an optional '.' character and sequence of digits followed by '%'. A property may define additional constraints on the value.

<script>

- A string of characters conforming to an ISO 15924 script code.

<space>

- A compound datatype, with components: minimum, optimum, maximum, precedence, and conditionality. The minimum, optimum, and maximum components are <length>s. The precedence component is either "force" or an <integer>. The conditionality component is either "discard" or "retain". If "minimum" is greater than optimum, it will be treated as if it had been set to "optimum". If "maximum" is less than optimum, it will be treated as if it had been set to "optimum".

<string>

- A sequence of characters.

<time>

- A <number> immediately followed by a time unit identifier. Time unit identifiers are: 'ms' (for milliseconds) and 's' (for seconds).

<uri-specification>

- A sequence of characters that is "url(", followed by optional white space, followed by an optional single quote (') or double quote (") character, followed by a URI reference as defined in RFC2396, followed by an optional single quote (') or double quote (") character, followed by optional white space, followed by ")". The two quote characters must be the same and must both be present or absent. If the URI reference contains a single quote, the two quote characters must be present and be double quotes.

Recommendation bibliography

Annex D - XSL-FO property summaries
Section 2 - Data types

Many of the data types refer to other standards and recommendations:

CSS2

- World Wide Web Consortium. *Cascading Style Sheets, level 2 (CSS2)*, as amended by Errata document 2001/04/04. W3C Recommendation. (`http://www.w3.org/TR/1998/REC-CSS2-19980512/`)

DSSSL

- International Organization for Standardization, International Electrotechnical Commission. *ISO/IEC 10179:1996. Document Style Semantics and Specification Language (DSSSL)*. International Standard.

ICC

- International Color Consortium. *Specification ICC.1:1998-09, File Format for Color Profiles*. (`http://www.color.org/ICC-1_1998-09.PDF`)

IEEE 754

- Institute of Electrical and Electronics Engineers. *IEEE Standard for Binary Floating-Point Arithmetic*. ANSI/IEEE Std 754-1985.

ISO15924

- International Organization for Standardization. *ISO 15924:1998. Code for the representation of names of scripts*. Draft International Standard.

ISO31

- International Organization for Standardization. *ISO 31:1992, Amended 1998. Quantities and units* International Standard.

JLS

- J. Gosling, B. Joy, and G. Steele. *The Java Language Specification*. (`http://java.sun.com/docs/books/jls/index.html`)

OpenType

- Microsoft, Adobe. *OpenType specification v.1.2*. (`http://www.microsoft.com/truetype/tt/tt.htm`)

RDF

- World Wide Web Consortium. *Resource Description Framework (RDF) Model and Syntax Specification*. W3C Recommendation. (`http://www.w3.org/TR/REC-rdf-syntax/`)

RFC2070

- IETF. *RFC 2070. Internationalization of the Hypertext Markup Language*. (`http://www.ietf.org/rfc/rfc2070.txt`)

RFC2119

- IETF. *RFC 2119. Key words for use in RFCs to Indicate Requirement Levels*. (`http://www.ietf.org/rfc/rfc2119.txt`)

RFC2396

- IETF. *RFC 2396. Uniform Resource Identifiers (URI): Generic Syntax*. (`http://www.ietf.org/rfc/rfc2396.txt`)

RFC3066

- IETF. *RFC 3066. Tags for the Identification of Languages*. (`http://www.ietf.org/rfc/rfc3066.txt`)

sRGB

- Anderson, M., Motta, R., Chandrasekar, S., and Stokes, M. *A Standard Default Color Space for the Internet - sRGB.* (`http://www.w3.org/Graphics/Color/sRGB.html`)

UNICODE

- Unicode Consortium. *The Unicode Standard, Version 3.0*. (`http://www.unicode.org/unicode/uni2book/u2.html`)

UNICODE Character Database

- Unicode Consortium. Unicode Character Database. (`http://www.unicode.org/Public/UNIDATA/`)

UNICODE TR20

- Unicode Consortium. Dürst, Martin and Freytag, Asmus. *Unicode Technical Report #20. Unicode in XML and other Markup Languages* Unicode Technical Report. (`http://www.unicode.org/unicode/reports/tr20/`)

UNICODE UAX #9

- Unicode Consortium. *The Unicode Standard, Version 3.1.0. Unicode Standard Annex #9: The Bidirectional Algorithm.* (`http://www.unicode.org/unicode/reports/tr9/`)

XML

- World Wide Web Consortium. *Extensible Markup Language (XML) 1.0.* W3C Recommendation. (`http://www.w3.org/TR/2000/REC-xml-20001006`)

XML Names

- World Wide Web Consortium. *Namespaces in XML.* W3C Recommendation. (`http://www.w3.org/TR/REC-xml-names/`)

XPath

- World Wide Web Consortium. *XML Path Language.* W3C Recommendation. (`http://www.w3.org/TR/xpath`)

XSLT

- World Wide Web Consortium. *XSL Transformations (XSLT).* W3C Recommendation. (`http://www.w3.org/TR/xslt`)

Format tokens from XSLT

A token can be used to represent how numbers are to be rendered as a sequence of characters:

- for `<xsl:number/>` in XSLT
- for page numbering in XSL-FO
- each of the representation of the numbers, the format of the separator sequences between multiple numbers, and the format of the terminator sequence can be specified using attributes allowing attribute value templates
- may be omitted to indicate decimal
    - default presentation is with the period separator (not terminator) sequence

`format="`*`token`*`"`

- specifies the counting scheme to be used when formatting the value
- `format="1"` counts `1, 2, ..., 9, 10, 11, ..., 99, 100, 101, ...`
- `format="01"` counts `01, 02, ..., 09, 10,..., 99, 100, ...`
    - each `"0"` prefix is a zero-fill indication for number values formatted less than the length of the format string
    - `grouping-separator=","` specifies the character between groups of digits
    - `grouping-size="3"` specifies the number of digits in each group (e.g.: 1,000,000)
- `format="a"` counts `a, b, ..., z, aa, ab, ac, ...`
- `format="A"` counts `A, B, ... Z, AA, AB, AC, ...`
    - `lang=` specifies the alphabet to be used
        - when numbering with an alphabetic sequence `a, b, ...` or `A, B, ...`
    - the same range of values as the `xml:lang=` attribute in XML 1.0
- `format="i"` counts `i, ii, iii, iv, v, ..., ix, x, xi, ...`
- `format="I"` counts `I, II, III, IV, V, ..., IX, X, XI, ...`
- `format="`*`a-Unicode-character`*`"` specifies a translation
    - converts the number into a representation based upon a specific language
    - the XSLT recommendation lists a number of examples of Unicode character representing specific conversions such as Katakana (regular and "iroha" orderings), Thai, Hebrew, Greek, Old Slavic, etc.
    - `letter-value="alphabetic"` and `letter-value="traditional"` for ambiguous distinctions
        - distinguishes numbering schemes in those languages where the first character of the sequence is ambiguous
        - unlike English where the differing first characters of `"a"` and `"i"` distinguish alphabetic and roman numeral formats

Inherited properties

Annex D - XSL-FO property summaries
Section 3 - Property summaries

The following traits are inherited from ancestral property specifications:

`auto-restore=` (7.22.2; 163)
`azimuth=` (7.6.1; 163)
`border-collapse=` (7.26.3; 170)
`border-separation=` (7.26.5; 173)
`border-spacing=` (7.29.9; 173)
`caption-side=` (7.26.7; 177)
`color=` (7.17.1; 178)
`country=` (7.9.1; 179)
`direction=` (7.27.1; 180)
`display-align=` (7.13.4; 180)
`elevation=` (7.6.4; 181)
`empty-cells=` (7.26.10; 181)
`end-indent=` (7.10.8; 182)
`font-family=` (7.8.2; 183)
`font-selection-strategy=` (7.8.3; 183)
`font-size=` (7.8.4; 183)
`font-size-adjust=` (7.8.6; 184)
`font-stretch=` (7.8.5; 184)
`font-style=` (7.8.7; 184)
`font-variant=` (7.8.8; 184)
`font-weight=` (7.8.9; 185)
`font=` (7.29.13; 183)
`glyph-orientation-horizontal=` (7.27.2; 185)
`glyph-orientation-vertical=` (7.27.3; 185)
`hyphenate=` (7.9.4; 186)
`hyphenation-character=` (7.9.5; 186)
`hyphenation-keep=` (7.15.1; 186)
`hyphenation-ladder-count=` (7.15.2; 187)
`hyphenation-push-character-count=` (7.9.6; 187)
`hyphenation-remain-character-count=` (7.9.7; 187)
`intrusion-displace=` (7.18.3; 188)
`keep-together=` (7.19.3; 188)
`language=` (7.9.2; 189)
`last-line-end-indent=` (7.15.3; 189)
`leader-alignment=` (7.21.1; 189)
`leader-length=` (7.21.4; 190)
`leader-pattern=` (7.21.2; 190)
`leader-pattern-width=` (7.21.3; 190)
`letter-spacing=` (7.16.2; 190)
`linefeed-treatment=` (7.15.7; 192)
`line-height=` (7.15.4; 191)
`line-height-shift-adjustment=` (7.15.5; 191)
`line-stacking-strategy=` (7.15.6; 191)
`orphans=` (7.19.6; 195)
`page-break-inside=` (7.29.18; 199)
`pitch=` (7.6.7; 200)
`pitch-range=` (7.6.8; 200)
`provisional-distance-between-starts=` (7.28.4; 201)
`provisional-label-separation=` (7.28.3; 202)
`reference-orientation=` (7.20.3; 202)
`relative-align=` (7.13.6; 202)
`richness=` (7.6.10; 204)
`rule-style=` (7.21.5; 204)
`rule-thickness=` (7.21.6; 205)
`score-spaces=` (7.28.6; 205)
`script=` (7.9.3; 205)
`speak=` (7.6.11; 207)
`speak-header=` (7.6.12; 207)
`speak-numeral=` (7.6.13; 207)
`speak-punctuation=` (7.6.14; 208)
`speech-rate=` (7.6.15; 208)
`start-indent=` (7.10.7; 208)
`stress=` (7.6.16; 209)
`text-align=` (7.15.9; 210)
`text-align-last=` (7.15.10; 210)
`text-indent=` (7.15.11; 211)
`text-transform=` (7.16.6; 212)
`visibility=` (7.28.8; 213)
`voice-family=` (7.6.17; 213)
`volume=` (7.6.18; 213)
`white-space-collapse=` (7.15.12; 214)
`white-space-treatment=` (7.15.8; 214)
`white-space=` (7.29.23; 214)
`widows=` (7.19.7; 214)
`word-spacing=` (7.16.8; 215)
`wrap-option=` (7.15.13; 215)
`writing-mode=` (7.27.7; 215)
`xml:lang=` (7.29.24; 215)

Shorthand properties

Annex D - XSL-FO property summaries
Section 3 - Property summaries

Important notes:

- shorthand properties need not be supported by a processor unless it is claiming complete conformance (the highest XSL-FO conformance level)
    - this is an important portability issue in that if one develops a stylesheet using a formatter that does recognize shorthand properties, that stylesheet may not run properly on a formatter that does not recognize shorthand properties
- if the value "`inherit`" is being used for a shorthand, it applies to all subproperty values
    - other subproperty values cannot be specified at the same time

Summary of shorthand properties

`background=` (7.29.1; 164)
`background-position=` (7.29.2; 165)
`border=` (7.29.3; 167)
`border-bottom=` (7.29.4; 169)
`border-color=` (7.29.5; 170)
`border-left=` (7.29.6; 171)
`border-right=` (7.29.7; 172)
`border-spacing=` (7.29.9; 173)
`border-style=` (7.29.8; 175)
`border-top=` (7.29.10; 175)
`border-width=` (7.29.11; 176)
`cue=` (7.29.12; 179)
`font=` (7.29.13; 183)
`margin=` (7.29.14; 192)
`padding=` (7.29.15; 195)
`page-break-after=` (7.29.16; 198)
`page-break-before=` (7.29.17; 199)
`page-break-inside=` (7.29.18; 199)
`pause=` (7.29.19; 199)
`position=` (7.29.20; 201)
`size=` (7.29.21; 205)
`vertical-align=` (7.29.22; 212)
`white-space=` (7.29.23; 214)
`xml:lang=` (7.29.24; 215)

Property summary

Annex D - XSL-FO property summaries
Section 3 - Property summaries

A alphabetical listing of all properties indicating each of:

- section number where found in Recommendation
- whether originally from CSS or from XSL
- conformance level (basic, extended, complete) required for support to be expected

Property definition notes:

- the following summary is extracted from the XSL Recommendation and that references to "prose" refer to the prose found in the Recommendation as noted in the given section number, and not necessarily to the content of this book
- in certain cases the XSL Recommendation lists the applicable formatting objects indirectly through the CSS definition and the list is incorrect due to XSL overrides
    - in this reference that list has been replaced with a synthesized list based on explicit references made in the XSL Recommendation definition of the formatting to the property
        - directly for the object
        - indirectly through common properties
        - indirectly through shorthand properties whose

`absolute-position=` (7.5.1; CSS; complete):

- Value: auto | absolute | fixed | inherit
- Initial: auto
- Inherited: no
- Media: visual

- an object's positioning as normally flowed relative to its sibling ("`auto`"), offset from its containing area ("`absolute`"), offset from the medium ("`fixed`"), or that of an ascendent specified value ("`inherit`")
- object to which this property applies: `<block-container>`
- shorthand impacting on this property: `position=`

`active-state=` (7.22.1; XSL; extended):

- Value: link | visited | active | hover | focus
- Initial: no, a value is required
- Inherited: no
- Media: interactive

- the state tested for a `<basic-link>` descendant of the parent `<multi-properties>` as being either not yet visited ("`link`"), ready to be engaged ("`hover`"), in the act of being engaged ("`active`"), the focus for input ("`focus`"), and already having been visited ("`visited`")
- object to which this property applies: `<multi-property-set>`

`alignment-adjust=` (7.13.1; XSL; basic):

- Value: auto | baseline | before-edge | text-before-edge | middle | central | after-edge | text-after-edge | ideographic | alphabetic | hanging | mathematical | <percentage> | <length> | inherit
- Initial: auto
- Inherited: no
- Percentages: see prose
- Media: visual

- determines the alignment point with which the formatter aligns areas created by formatting objects (such as graphics) that either do not have a baseline-table or don't have a desired baseline in that table to the `alignment-baseline=` of the parent
  - refer to `baseline-shift=` for suitable (not exclusive) behavior for text
- percentages are of either the computed area for an image, the font size for a character, or the line height for other constructs
- positive lengths are opposite to the shift direction
- see Chapter 4 Area and page basics (page 70) for details of values
- objects to which this property applies: `<basic-link>`, `<character>`, `<external-graphic>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<page-number>`, `<page-number-citation>`
- shorthand impacting on this property: `vertical-align=`

`alignment-baseline=` (7.13.2; XSL; basic):

- Value: auto | baseline | before-edge | text-before-edge | middle | central | after-edge | text-after-edge | ideographic | alphabetic | hanging | mathematical | inherit
- Initial: auto
- Inherited: no
- Media: visual

- specifies to which of an object's parent's baselines the object's alignment point (as defined by `alignment-adjust=`) is aligned
- objects to which this property applies: `<basic-link>`, `<character>`, `<external-graphic>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<page-number>`, `<page-number-citation>`
- shorthand impacting on this property: `vertical-align=`

`auto-restore=` (7.22.2; XSL; extended):

- Value: true | false
- Initial: false
- Inherited: yes
- Media: interactive

- when "true" for an object that object's `<multi-case>` will be restored to its initial value when that object is hidden by the hiding of an ancestral `<multi-switch>` construct
- object to which this property applies: `<multi-switch>`

`azimuth=` (7.6.1; CSS; basic):

- Value: <angle> | [[left-side | far-left | left | center-left | center | center-right | right | far-right | right-side] || behind] | leftwards | rightwards | inherit
- Initial: center
- Inherited: yes
- Media: aural

- directs from which lateral direction an aural presentation is heard with respect to the listener
- objects to which this property applies: `<basic-link>`, `<bidi-override>`, `<block>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`

`background=` (7.29.1; CSS; shorthand):

- Value: [`background-color=` || `background-image=` || `background-repeat=` || `background-attachment=` || `background-position=`] | inherit
- Initial: not defined for shorthand properties
- Inherited: no
- Percentages: allowed on 'background-position'
- Media: visual

- provides a shorthand method of specifying the five individual values
- any values omitted are given their initial values

`background-attachment=` (7.7.1; CSS; extended):

- Value: scroll | fixed | inherit
- Initial: scroll
- Inherited: no
- Media: visual

- when the background is "`fixed`" it remains stationary in the viewport while the foreground scrolls in front of it
- objects to which this property applies: `<basic-link>`, `<block>`, `<block-container>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<region-after>`, `<region-before>`, `<region-body>`, `<region-end>`, `<region-start>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-column>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`
- shorthand impacting on this property: `background=`

`background-color=` (7.7.2; CSS; basic):

- Value: <color> | transparent | inherit
- Initial: transparent
- Inherited: no
- Media: visual

- fills the padding rectangle behind an object's content, thus the color is seen within the inside edge of a visible border
  - gaps in a visible border show through the background color of the parent area
- objects to which this property applies: `<basic-link>`, `<block>`, `<block-container>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<region-after>`, `<region-before>`, `<region-body>`, `<region-end>`, `<region-start>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-column>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`
- shorthand impacting on this property: `background=`

`background-image=` (7.7.3; CSS; extended):

- Value: <uri-specification> | none | inherit
- Initial: none
- Inherited: no
- Media: visual

- positioned (and possibly repeated with `background-repeat=` within the padding rectangle behind an object's content
- the `background-color=` will show through transparent parts of the background image
- objects to which this property applies: `<basic-link>`, `<block>`, `<block-container>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<region-after>`, `<region-before>`, `<region-body>`, `<region-end>`, `<region-start>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-column>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`
- shorthand impacting on this property: `background=`

`background-position=` (7.29.2; CSS; shorthand):

- Value: [[<percentage> | <length>]{1,2} | [[top | center | bottom] || [left | center | right]]] | inherit
- Initial: 0% 0%
- Inherited: no
- Percentages: refer to the size of the box itself
- Media: visual

- a shorthand for specifying one or two positions oriented to the upper left corner of the padding-rectangle, setting individual values for `background-position-horizontal=` and `background-position-vertical=`
- the horizontal value is specified first followed by the vertical value
- an absent value is interpreted as "`center`" which is placed at a position of 50%

`background-position-horizontal=` (7.7.5; CSS; extended):

- Value: <percentage> | <length> | left | center | right | inherit
- Initial: 0%
- Inherited: no
- Percentages: refer to the size of the padding-rectangle
- Media: visual

- objects to which this property applies: `<basic-link>`, `<block>`, `<block-container>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<region-after>`, `<region-before>`, `<region-body>`, `<region-end>`, `<region-start>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-column>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`
- shorthand impacting on this property: `background-position=`

`background-position-vertical=` (7.7.6; CSS; extended):

- Value: <percentage> | <length> | top | center | bottom | inherit
- Initial: 0%
- Inherited: no
- Percentages: refer to the size of the padding-rectangle
- Media: visual

- objects to which this property applies: `<basic-link>`, `<block>`, `<block-container>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<region-after>`, `<region-before>`, `<region-body>`, `<region-end>`, `<region-start>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-column>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`
- shorthand impacting on this property: `background-position=`

`background-repeat=` (7.7.4; CSS; extended):

- Value: repeat | repeat-x | repeat-y | no-repeat | inherit
- Initial: repeat
- Inherited: no
- Media: visual

- "x" is the horizontal direction, "y" is the vertical direction, both relative to the reference-orientation but independent of the writing-mode
- objects to which this property applies: `<basic-link>`, `<block>`, `<block-container>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<region-after>`, `<region-before>`, `<region-body>`, `<region-end>`, `<region-start>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-column>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`
- shorthand impacting on this property: `background=`

`baseline-shift=` (7.13.3; XSL; basic):

- Value: baseline | sub | super | <percentage> | <length> | inherit
- Initial: baseline
- Inherited: no
- Percentages: refers to the "line-height" of the parent area
- Media: visual

- repositions the dominant-baseline (and all other entries in the baseline-table) of an object relative to the dominant-baseline of the parent area
- using the parent area as a reference is more consistent for a set of child objects of different sizes
- objects to which this property applies: `<basic-link>`, `<character>`, `<external-graphic>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<page-number>`, `<page-number-citation>`
- shorthand impacting on this property: `vertical-align=`

`blank-or-not-blank=` (7.25.1; XSL; extended):

- Value: blank | not-blank | any | inherit
- Initial: any
- Inherited: no
- Media: visual

- object to which this property applies: `<conditional-page-master-reference>`

`block-progression-dimension=` (7.14.1; CSS; basic):

- Value: auto | <length> | <percentage> | <length-range> | inherit
- Initial: auto
- Inherited: no
- Percentages: see prose
- Media: visual

- specifies the block-progression-dimension of the content-rectangle for each area generated by the formatting object
- objects to which this property applies: `<block-container>`, `<external-graphic>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<table>`, `<table-caption>`, `<table-cell>`, `<table-row>`

`border=` (7.29.3; CSS; shorthand):

- Value: [`border-width=` || `border-style=` || <color>] | inherit
- Initial: see individual properties
- Inherited: no
- Media: visual

- sets any of the border width, style or color properties for all of the four borders of a box

`border-after-color=` (7.7.10; CSS; basic):

- Value: <color> | inherit
- Initial: the value of the 'color' property
- Inherited: no
- Media: visual

- objects to which this property applies: `<basic-link>`, `<block>`, `<block-container>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<region-after>`, `<region-before>`, `<region-body>`, `<region-end>`, `<region-start>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-column>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`
- shorthands impacting on this property: `border=`, `border-color=`

`border-after-precedence=` (7.26.1; XSL; basic):

- Value: force | <integer> | inherit
- Initial: `<table>`: 6, `<table-cell>`: 5, `<table-column>`: 4, `<table-row>`: 3, `<table-body>`: 2, `<table-header>`: 1, `<table-footer>`: 0
- Inherited: no
- Media: visual

- specifies the precedence of the formatting object's border compared to the precedence of the coincident border of:
  - an adjacent formatting object when borders are collapsed using `border-collapse=`
  - an overlapping table construct in the formatting object hierarchy
- the border specification precedence that is highest of all of the coincident borders dictates which formatting object's properties that border will exhibit
- objects to which this property applies: `<table>`, `<table-body>`, `<table-cell>`, `<table-column>`, `<table-footer>`, `<table-header>`, `<table-row>`

`border-after-style=` (7.7.11; CSS; basic):

- Value: `border-style=` | inherit
- Initial: none
- Inherited: no
- Media: visual

- objects to which this property applies: `<basic-link>`, `<block>`, `<block-container>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<region-after>`, `<region-before>`, `<region-body>`, `<region-end>`, `<region-start>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-column>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`
- shorthand impacting on this property: `border-style=`

`border-after-width=` (7.7.12; CSS; basic):

- Value: `border-width=` | <length-conditional> | inherit
- Initial: medium
- Inherited: no
- Media: visual

- objects to which this property applies: `<basic-link>`, `<block>`, `<block-container>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<region-after>`, `<region-before>`, `<region-body>`, `<region-end>`, `<region-start>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-column>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`
- shorthand impacting on this property: `border-width=`

`border-before-color=` (7.7.7; CSS; basic):

- Value: <color> | inherit
- Initial: the value of the 'color' property
- Inherited: no
- Media: visual

- objects to which this property applies: `<basic-link>`, `<block>`, `<block-container>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<region-after>`, `<region-before>`, `<region-body>`, `<region-end>`, `<region-start>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-column>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`
- shorthands impacting on this property: `border=`, `border-color=`

`border-before-precedence=` (7.26.2; XSL; basic):

- Value: force | <integer> | inherit
- Initial: `<table>`: 6, `<table-cell>`: 5, `<table-column>`: 4, `<table-row>`: 3, `<table-body>`: 2, `<table-header>`: 1, `<table-footer>`: 0
- Inherited: no
- Media: visual

- see `border-after-precedence=` for details
- objects to which this property applies: `<table>`, `<table-body>`, `<table-cell>`, `<table-column>`, `<table-footer>`, `<table-header>`, `<table-row>`

`border-before-style=` (7.7.8; CSS; basic):

- Value: `border-style=` | inherit
- Initial: none
- Inherited: no
- Media: visual

- objects to which this property applies: `<basic-link>`, `<block>`, `<block-container>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<region-after>`, `<region-before>`, `<region-body>`, `<region-end>`, `<region-start>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-column>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`
- shorthand impacting on this property: `border-style=`

`border-before-width=` (7.7.9; CSS; basic):

- Value: `border-width=` | <length-conditional> | inherit
- Initial: medium
- Inherited: no
- Media: visual

- objects to which this property applies: `<basic-link>`, `<block>`, `<block-container>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<region-after>`, `<region-before>`, `<region-body>`, `<region-end>`, `<region-start>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-column>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`
- shorthand impacting on this property: `border-width=`

`border-bottom=` (7.29.4; CSS; shorthand):

- Value: [`border-width=` || `border-style=` || <color>] | inherit
- Initial: see individual properties
- Inherited: no
- Media: visual

- a shorthand for specifying a border's width, style and color

`border-bottom-color=` (7.7.22; CSS; basic):

- Value: <color> | inherit
- Initial: the value of the 'color' property
- Inherited: no
- Media: visual

- objects to which this property applies: `<basic-link>`, `<block>`, `<block-container>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<region-after>`, `<region-before>`, `<region-body>`, `<region-end>`, `<region-start>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-column>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`
- shorthands impacting on this property: `border-bottom=`, `border-color=`

`border-bottom-style=` (7.7.23; CSS; basic):

- Value: `border-style=` | inherit
- Initial: none
- Inherited: no
- Media: visual

- objects to which this property applies: `<basic-link>`, `<block>`, `<block-container>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<region-after>`, `<region-before>`, `<region-body>`, `<region-end>`, `<region-start>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-column>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`
- shorthands impacting on this property: `border-bottom=`, `border-style=`

`border-bottom-width=` (7.7.24; CSS; basic):

- Value: `border-width=` | inherit
- Initial: medium
- Inherited: no
- Media: visual

- objects to which this property applies: `<basic-link>`, `<block>`, `<block-container>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<region-after>`, `<region-before>`, `<region-body>`, `<region-end>`, `<region-start>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-column>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`
- shorthands impacting on this property: `border-bottom=`, `border-width=`

`border-collapse=` (7.26.3; CSS; extended):

- Value: collapse | collapse-with-precedence | separate | inherit
- Initial: collapse
- Inherited: yes
- Media: visual

- object to which this property applies: `<table>`

`border-color=` (7.29.5; CSS; shorthand):

- Value: [<color> | transparent]{1,4} | inherit
- Initial: see individual properties
- Inherited: no
- Media: visual

- up to four individual values for the border color can be specified in this shorthand
  - see Chapter 9 Breaks, keeps, spacing, borders and backgrounds (page 104) for a discussion of borders
- four colors specifies each of the borders of a table in the order: before, end, after, start
- three colors specifies the borders in the order: before, end/start, after
- two colors specifies the borders in the order: before/after, end/start
- one color specifies the color of all borders

`border-end-color=` (7.7.16; CSS; basic):

- Value: <color> | inherit
- Initial: the value of the 'color' property
- Inherited: no
- Media: visual

- objects to which this property applies: <basic-link>, <block>, <block-container>, <character>, <external-graphic>, <initial-property-set>, <inline>, <inline-container>, <instream-foreign-object>, <leader>, <list-block>, <list-item>, <page-number>, <page-number-citation>, <region-after>, <region-before>, <region-body>, <region-end>, <region-start>, <table-and-caption>, <table>, <table-body>, <table-caption>, <table-cell>, <table-column>, <table-footer>, <table-header>, <table-row>, <title>
- shorthands impacting on this property: border=, border-color=

border-end-precedence= (7.26.4; XSL; basic):

- Value: force | <integer> | inherit
- Initial: <table>: 6, <table-cell>: 5, <table-column>: 4, <table-row>: 3, <table-body>: 2, <table-header>: 1, <table-footer>: 0
- see border-after-precedence= for details
- objects to which this property applies: <table>, <table-body>, <table-cell>, <table-column>, <table-footer>, <table-header>, <table-row>
- Inherited: no
- Media: visual

border-end-style= (7.7.17; CSS; basic):

- Value: border-style= | inherit
- Initial: none
- objects to which this property applies: <basic-link>, <block>, <block-container>, <character>, <external-graphic>, <initial-property-set>, <inline>, <inline-container>, <instream-foreign-object>, <leader>, <list-block>, <list-item>, <page-number>, <page-number-citation>, <region-after>, <region-before>, <region-body>, <region-end>, <region-start>, <table-and-caption>, <table>, <table-body>, <table-caption>, <table-cell>, <table-column>, <table-footer>, <table-header>, <table-row>, <title>
- shorthand impacting on this property: border-style=
- Inherited: no
- Media: visual

border-end-width= (7.7.18; CSS; basic):

- Value: border-width= | <length-conditional> | inherit
- Initial: medium
- objects to which this property applies: <basic-link>, <block>, <block-container>, <character>, <external-graphic>, <initial-property-set>, <inline>, <inline-container>, <instream-foreign-object>, <leader>, <list-block>, <list-item>, <page-number>, <page-number-citation>, <region-after>, <region-before>, <region-body>, <region-end>, <region-start>, <table-and-caption>, <table>, <table-body>, <table-caption>, <table-cell>, <table-column>, <table-footer>, <table-header>, <table-row>, <title>
- shorthand impacting on this property: border-width=
- Inherited: no
- Media: visual

border-left= (7.29.6; CSS; shorthand):

- Inherited: no
- Media: visual

- Value: [`border-width=` || `border-style=` || <color>] | inherit
- Initial: see individual properties

- a shorthand for specifying a border's width, style and color

`border-left-color=` (7.7.25; CSS; basic):

- Value: <color> | inherit
- Initial: the value of the 'color' property
- Inherited: no
- Media: visual

- objects to which this property applies: `<basic-link>`, `<block>`, `<block-container>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<region-after>`, `<region-before>`, `<region-body>`, `<region-end>`, `<region-start>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-column>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`
- shorthands impacting on this property: `border-color=`, `border-left=`

`border-left-style=` (7.7.26; CSS; basic):

- Value: `border-style=` | inherit
- Initial: none
- Inherited: no
- Media: visual

- objects to which this property applies: `<basic-link>`, `<block>`, `<block-container>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<region-after>`, `<region-before>`, `<region-body>`, `<region-end>`, `<region-start>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-column>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`
- shorthands impacting on this property: `border-left=`, `border-style=`

`border-left-width=` (7.7.27; CSS; basic):

- Value: `border-width=` | inherit
- Initial: medium
- Inherited: no
- Media: visual

- objects to which this property applies: `<basic-link>`, `<block>`, `<block-container>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<region-after>`, `<region-before>`, `<region-body>`, `<region-end>`, `<region-start>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-column>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`
- shorthands impacting on this property: `border-left=`, `border-width=`

`border-right=` (7.29.7; CSS; shorthand):

- Value: [`border-width=` || `border-style=` || <color>] | inherit
- Initial: see individual properties
- Inherited: no
- Media: visual

- a shorthand for specifying a border's width, style and color

`border-right-color=` (7.7.28; CSS; basic):

- Value: <color> | inherit
- Initial: the value of the 'color' property
- Inherited: no
- Media: visual
- objects to which this property applies: `<basic-link>`, `<block>`, `<block-container>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<region-after>`, `<region-before>`, `<region-body>`, `<region-end>`, `<region-start>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-column>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`
- shorthands impacting on this property: `border-color=`, `border-right=`

`border-right-style=` (7.7.29; CSS; basic):

- Value: `border-style=` | inherit
- Initial: none
- Inherited: no
- Media: visual
- objects to which this property applies: `<basic-link>`, `<block>`, `<block-container>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<region-after>`, `<region-before>`, `<region-body>`, `<region-end>`, `<region-start>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-column>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`
- shorthands impacting on this property: `border-right=`, `border-style=`

`border-right-width=` (7.7.30; CSS; basic):

- Value: `border-width=` | inherit
- Initial: medium
- Inherited: no
- Media: visual
- objects to which this property applies: `<basic-link>`, `<block>`, `<block-container>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<region-after>`, `<region-before>`, `<region-body>`, `<region-end>`, `<region-start>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-column>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`
- shorthands impacting on this property: `border-right=`, `border-width=`

`border-separation=` (7.26.5; XSL; extended):

- Value: <length-bp-ip-direction> | inherit
- Initial: .block-progression-direction="0pt" .inline-progression-direction="0pt"
- Inherited: yes
- Media: visual
- specifies the distance between the borders of adjacent cells when using borders are not collapsed using `border-collapse=`
- the space in between borders is filled with the table background color
- object to which this property applies: `<table>`
- shorthand impacting on this property: `border-spacing=`

`border-spacing=` (7.29.9; CSS; shorthand):

- Value: <length> <length>? | inherit
- Initial: 0pt
- Inherited: yes
- Media: visual

- this shorthand specifies the two components of the compound `border-separation=` property as separate values in the order of the `.inline-progression-direction` component from the first value then the `.block-progression-direction` component from the second value
- specifying only one value sets both components simultaneously

`border-start-color=` (7.7.13; CSS; basic):

- Value: <color> | inherit
- Initial: the value of the 'color' property
- Inherited: no
- Media: visual

- objects to which this property applies: `<basic-link>`, `<block>`, `<block-container>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<region-after>`, `<region-before>`, `<region-body>`, `<region-end>`, `<region-start>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-column>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`
- shorthands impacting on this property: `border=`, `border-color=`

`border-start-precedence=` (7.26.6; XSL; basic):

- Value: force | <integer> | inherit
- Initial: `<table>`: 6, `<table-cell>`: 5, `<table-column>`: 4, `<table-row>`: 3, `<table-body>`: 2, `<table-header>`: 1, `<table-footer>`: 0
- Inherited: no
- Media: visual

- see `border-after-precedence=` for details
- objects to which this property applies: `<table>`, `<table-body>`, `<table-cell>`, `<table-column>`, `<table-footer>`, `<table-header>`, `<table-row>`

`border-start-style=` (7.7.14; CSS; basic):

- Value: `border-style=` | inherit
- Initial: none
- Inherited: no
- Media: visual

- objects to which this property applies: `<basic-link>`, `<block>`, `<block-container>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<region-after>`, `<region-before>`, `<region-body>`, `<region-end>`, `<region-start>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-column>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`
- shorthand impacting on this property: `border-style=`

`border-start-width=` (7.7.15; CSS; basic):

- Value: `border-width=` | <length-conditional> | inherit
- Initial: medium
- Inherited: no
- Media: visual

- objects to which this property applies: `<basic-link>`, `<block>`, `<block-container>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<region-after>`, `<region-before>`, `<region-body>`, `<region-end>`, `<region-start>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-column>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`
- shorthand impacting on this property: `border-width=`

`border-style=` (7.29.8; CSS; shorthand):

- Value: *border-style*{1,4} | inherit
- Inherited: no
- Initial: see individual properties
- Media: visual

- up to four individual values for the border style can be specified in this shorthand
  - see Chapter 9 Breaks, keeps, spacing, borders and backgrounds (page 104) for a discussion of borders
- four values specifies each of the borders of a table in the order: before, end, after, start
- three values specifies the borders in the order: before, end/start, after
- two values specifies the borders in the order: before/after, end/start
- one value specifies the style of all borders

`border-top=` (7.29.10; CSS; shorthand):

- Value: [`border-width=` || `border-style=` || <color>] | inherit
- Inherited: no
- Media: visual
- Initial: see individual properties

- a shorthand for specifying a border's width, style and color

`border-top-color=` (7.7.19; CSS; basic):

- Value: <color> | inherit
- Inherited: no
- Initial: the value of the 'color' property
- Media: visual

- objects to which this property applies: `<basic-link>`, `<block>`, `<block-container>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<region-after>`, `<region-before>`, `<region-body>`, `<region-end>`, `<region-start>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-column>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`
- shorthands impacting on this property: `border-color=`, `border-top=`

`border-top-style=` (7.7.20; CSS; basic):

- Value: `border-style=` | inherit
- Inherited: no
- Initial: none
- Media: visual

- objects to which this property applies: `<basic-link>`, `<block>`, `<block-container>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<region-after>`, `<region-before>`, `<region-body>`, `<region-end>`, `<region-start>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-column>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`
- shorthands impacting on this property: `border-style=`, `border-top=`

`border-top-width=` (7.7.21; CSS; basic):

- Value: `border-width=` | inherit
- Initial: medium
- Inherited: no
- Media: visual

- objects to which this property applies: `<basic-link>`, `<block>`, `<block-container>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<region-after>`, `<region-before>`, `<region-body>`, `<region-end>`, `<region-start>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-column>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`
- shorthands impacting on this property: `border-top=`, `border-width=`

`border-width=` (7.29.11; CSS; shorthand):

- Value: *border-width*{1,4} | inherit
- Initial: see individual properties
- Inherited: no
- Media: visual

- up to four individual values for the border width can be specified in this shorthand
  - see Chapter 9 Breaks, keeps, spacing, borders and backgrounds (page 104) for a discussion of borders
- four values specifies each of the borders of a table in the order: before, end, after, start
- three values specifies the borders in the order: before, end/start, after
- two values specifies the borders in the order: before/after, end/start
- one value specifies the width of all borders

`bottom=` (7.5.4; CSS; extended):

- Value: <length> | <percentage> | auto | inherit
- Initial: auto
- Inherited: no
- Percentages: refer to height of containing block
- Media: visual

- specifies the distance between the margin edge and the containing block
- a value other than "`auto`" overrides a `height=` value of "`auto`"
- the percentage is ignored if the containing block height is not specified explicitly
- object to which this property applies: `<block-container>`

`break-after=` (7.19.1; XSL; basic):

- Value: auto | column | page | even-page | odd-page | inherit
- Initial: auto
- Inherited: no
- Media: visual

- this property has no effect on `<table-row>` if that row has a row-spanning cell including the following row
- objects to which this property applies: `<block>`, `<block-container>`, `<list-block>`, `<list-item>`, `<table-and-caption>`, `<table>`, `<table-row>`
- shorthand impacting on this property: `page-break-after=`

`break-before=` (7.19.2; XSL; basic):

- Value: auto | column | page | even-page | odd-page | inherit
- Initial: auto
- Inherited: no
- Media: visual

- this property has no effect on `<table-row>` if the previous row has a row-spanning cell including the given row
- objects to which this property applies: `<block>`, `<block-container>`, `<list-block>`, `<list-item>`, `<table-and-caption>`, `<table>`, `<table-row>`
- shorthand impacting on this property: `page-break-before=`

`caption-side=` (7.26.7; CSS; complete):

- Value: before | after | start | end | top | bottom | left | right | inherit
- Initial: before
- Inherited: yes
- Media: visual

- object to which this property applies: `<table-and-caption>`

`case-name=` (7.22.3; XSL; extended):

- Value: <name>
- Initial: none, a value is required
- Inherited: no, a value is required
- Media: interactive

- this name must be unique among the siblings of the `<multi-case>`
- object to which this property applies: `<multi-case>`

`case-title=` (7.22.4; XSL; extended):

- Value: <string>
- Initial: none, a value is required
- Inherited: no, a value is required
- Media: interactive

- this string can be displayed in a menu corresponding to the associated `<multi-case>` objects of allowed `<multi-toggle>` destinations when more than one candidate destination is specified
- object to which this property applies: `<multi-case>`

`character=` (7.16.1; XSL; basic):

- Value: <character>
- Initial: N/A, value is required
- Inherited: no, a value is required
- Media: visual

- object to which this property applies: `<character>`

`clear=` (7.18.1; CSS; extended):

- Value: start | end | left | right | both | none | inherit
- Initial: none
- Inherited: no
- Media: visual

- specifies which side-floats whose parent reference-area is the nearest ancestor of the reference-area of the generated area must be clear of the given block to which this property is specified
    - the value indicates which side float the given construct is supposed to clear
- this may alter the `space-before=` property to meet the constraints
- also applies to `<block>` even though not listed in the Recommendation in the list of properties for the object
- object to which this property applies: `<float>`

`clip=` (7.20.1; CSS; extended):

- Value: ***shape*** | auto | inherit
- Initial: auto
- Inherited: no
- Media: visual
- the only shape supported is `rect( top, right, bottom, left )`
    - the four values are the offsets of the clipping area from the respective sides of the containing area
- the value "`auto`" represents an offset of zero
- objects to which this property applies: `<block-container>`, `<external-graphic>`, `<inline-container>`, `<instream-foreign-object>`, `<region-after>`, `<region-before>`, `<region-body>`, `<region-end>`, `<region-start>`

`color=` (7.17.1; CSS; basic):

- Value: <color> | inherit
- Initial: depends on user agent
- Inherited: yes
- Media: visual
- see Function groupings (page 126) for the color functions available
- objects to which this property applies: `<bidi-override>`, `<block>`, `<character>`, `<initial-property-set>`, `<inline>`, `<leader>`, `<title>`

`color-profile-name=` (7.17.2; XSL; extended):

- Value: <name> | inherit
- Initial: N/A, value is required
- Inherited: no
- Media: visual
- object to which this property applies: `<color-profile>`

`column-count=` (7.25.2; XSL; extended):

- Value: <number> | inherit
- Initial: 1
- Inherited: no
- Media: visual
- object to which this property applies: `<region-body>`

`column-gap=` (7.25.3; XSL; extended):

- Value: <length> | <percentage> | inherit
- Initial: 12.0pt
- Inherited: no
- Percentages: refer to width of the region being divided into columns.
- Media: visual
- a negative value is translated to 0pt
- object to which this property applies: `<region-body>`

`column-number=` (7.26.8; XSL; basic):

- Value: <number>
- Initial: see prose
- Inherited: no
- Media: visual

- the initial value for `<table-column>` is 1 plus the `column-number=` of the previous `<table-column>`, or "1" for the first
- the initial value for `<table-cell>` is the `column-number=` of the previous cell plus the `number-columns-spanned=` of that previous cell
- objects to which this property applies: `<table-cell>`, `<table-column>`

`column-width=` (7.26.9; XSL; basic):

- Value: <length> | <percentage>
- Initial: see prose
- Inherited: no
- Percentages: refer to width of table
- Media: visual

- see Chapter 6 Tables (page 82) for a discussion of column widths
- object to which this property applies: `<table-column>`

`content-height=` (7.14.2; XSL; extended):

- Value: auto | scale-to-fit | <length> | <percentage> | inherit
- Initial: auto
- Inherited: no
- Percentages: intrinsic height
- Media: visual

- objects to which this property applies: `<external-graphic>`, `<instream-foreign-object>`

`content-type=` (7.28.1; XSL; extended):

- Value: <string> | auto
- Initial: auto
- Inherited: no
- Media: visual

- the string is prefixed by "`namespace-prefix:`" for a namespace specification
  - a null prefix refers to the default namespace
- the string is prefixed by "`content-type:`" for a mime-type specification
- objects to which this property applies: `<external-graphic>`, `<instream-foreign-object>`

`content-width=` (7.14.3; XSL; extended):

- Value: auto | scale-to-fit | <length> | <percentage> | inherit
- Initial: auto
- Inherited: no
- Percentages: intrinsic width
- Media: visual

- objects to which this property applies: `<external-graphic>`, `<instream-foreign-object>`

`country=` (7.9.1; XSL; extended):

- Value: none | <country> | inherit
- Initial: none
- Inherited: yes
- Media: visual

- specifies the country used in language- and locale- coupled services
  - e.g. line-justification, line-breaking, hyphenation, etc.
- objects to which this property applies: `<block>`, `<character>`, `<page-sequence>`
- shorthand impacting on this property: `xml:lang=`

`cue=` (7.29.12; CSS; shorthand):

- Value: `cue-before=` || `cue-after=` | inherit
- Inherited: no
- Initial: not defined for shorthand properties
- Media: aural
- when a single value is specified it is applied to both properties

`cue-after=` (7.6.2; CSS; basic):

- Value: <uri-specification> | none | inherit
- Inherited: no
- Initial: none
- Media: aural
- specifies the URL of rendered information as an "auditory icon" after a construct is rendered
- objects to which this property applies: `<basic-link>`, `<bidi-override>`, `<block>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`
- shorthand impacting on this property: `cue=`

`cue-before=` (7.6.3; CSS; basic):

- Value: <uri-specification> | none | inherit
- Inherited: no
- Initial: none
- Media: aural
- specifies the URL of rendered information as an "auditory icon" before a construct is rendered
- objects to which this property applies: `<basic-link>`, `<bidi-override>`, `<block>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`
- shorthand impacting on this property: `cue=`

`destination-placement-offset=` (7.22.5; XSL; extended):

- Value: <length>
- Inherited: no
- Initial: 0pt
- Media: interactive
- specifies where in the destination view port or page (from the beginning) that the targeted location is to be rendered
- object to which this property applies: `<basic-link>`

`direction=` (7.27.1; CSS; basic):

- Value: ltr | rtl | inherit
- Inherited: yes
- Initial: ltr
- Media: visual
- this property is deprecated for all formatting objects other than `<bidi-override>`
  - if used for such other objects, this overrides the direction implied by any `writing-mode=` property
- object to which this property applies: `<bidi-override>`

`display-align=` (7.13.4; XSL; extended):

- Value: auto | before | center | after | inherit
- Initial: auto
- Inherited: yes
- Media: visual

- this specifies the alignment, in the block-progression-direction, of the areas that are the children of a reference-area
- the value of "`auto`" infers either the `relative-align=` property if applicable, or the value "`before`"
- objects to which this property applies: `<block-container>`, `<external-graphic>`, `<inline-container>`, `<instream-foreign-object>`, `<region-after>`, `<region-before>`, `<region-body>`, `<region-end>`, `<region-start>`, `<table-cell>`

`dominant-baseline=` (7.13.5; XSL; basic):

- Value: auto | use-script | no-change | reset-size | ideographic | alphabetic | hanging | mathematical | central | middle | text-after-edge | text-before-edge | inherit
- Initial: auto
- Inherited: no
- Media: visual

- determines or re-determines a scaled-baseline-table
- objects to which this property applies: `<basic-link>`, `<character>`, `<external-graphic>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<page-number>`, `<page-number-citation>`
- shorthand impacting on this property: `vertical-align=`

`elevation=` (7.6.4; CSS; basic):

- Value: <angle> | below | level | above | higher | lower | inherit
- Initial: level
- Inherited: yes
- Media: aural

- the angles "`90deg`" and "`-90deg`" position the sound respectively directly above or below the listener
- the values "`higher`" and "`lower`" respectively increase and decrease the current elevation by 10 degrees
- objects to which this property applies: `<basic-link>`, `<bidi-override>`, `<block>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`

`empty-cells=` (7.26.10; CSS; extended):

- Value: show | hide | inherit
- Initial: show
- Inherited: yes
- Media: visual

- specifies the rendering of the borders and background of cells without visible content when using the separated borders model specified by `border-collapse=`
- if all cells in a row have a value of "`hide`", the row behaves as if were not displayed at all
- visible content is everything other than XML white-space (carriage-return, linefeed, tab, and space)
- object to which this property applies: `<table-cell>`

`end-indent=` (7.10.8; XSL; basic):

- Value: <length> | <percentage> | inherit
- Initial: 0pt
- Inherited: yes
- Percentages: refer to inline-progression-dimension of containing reference-area
- Media: visual
- objects to which this property applies: `<block>`, `<block-container>`, `<list-block>`, `<list-item>`, `<region-body>`, `<simple-page-master>`, `<table-and-caption>`, `<table>`

`ends-row=` (7.26.11; XSL; extended):

- Value: true | false
- Initial: false
- Inherited: no
- Media: visual
- object to which this property applies: `<table-cell>`

`extent=` (7.25.4; XSL; extended):

- Value: <length> | <percentage> | inherit
- Initial: 0.0pt
- Inherited: no
- Percentages: refer to the corresponding height or width of the page-viewport-area.
- Media: visual
- objects to which this property applies: `<region-after>`, `<region-before>`, `<region-end>`, `<region-start>`

`external-destination=` (7.22.6; XSL; extended):

- Value: <uri-specification>
- Initial: empty string
- Inherited: no
- Media: interactive
- at least one of `external-destination=` and `internal-destination=` properties should be assigned
- if both are assigned, the system may either report the error, or use `internal-destination=`
- object to which this property applies: `<basic-link>`

`float=` (7.18.2; CSS; extended):

- Value: before | start | end | left | right | none | inherit
- Initial: none
- Inherited: no
- Media: visual
- unlike CSS, this property only applies to `<float>` and to no other formatting object
- object to which this property applies: `<float>`

`flow-name=` (7.25.5; XSL; basic):

- Value: <name>
- Initial: an empty name
- Inherited: no, a value is required
- Media: visual

- must be unique within a `<page-sequence>`
- the following names are reserved: "`xsl-region-body`", "`xsl-region-before`", "`xsl-region-after`", "`xsl-region-start`", "`xsl-region-end`", "`xsl-before-float-separator`", "`xsl-footnote-separator`"
- objects to which this property applies: `<flow>`, `<static-content>`

`font=` (7.29.13; CSS; shorthand):

- Value: [[`font-style=` || `font-variant=` || `font-weight=`]? `font-size=` [/ `line-height=`]? `font-family=`] | caption | icon | menu | message-box | small-caption | status-bar | inherit
- Initial: see individual properties
- Inherited: yes
- Media: visual

- individual system font characteristics, such as font-family, font-size, etc. may be obtained by the use of the `system-font()` function without any arguments

`font-family=` (7.8.2; CSS; basic):

- Value: [[<family-name> | *generic-family*],]* [<family-name> | *generic-family*] | inherit
- Initial: depends on user agent
- Inherited: yes
- Media: visual

- font family names with spaces should be quoted, but will otherwise be normalized
- generic font family names are not quoted
  - the values are "`serif`", "`sans-serif`", "`cursive`", "`fantasy`", and "`monospace`"
- this property is a prioritized list of font family names which are tried in sequence to find an available font that matches the selection criteria specified by `font-selection-strategy=`
- objects to which this property applies: `<bidi-override>`, `<block>`, `<character>`, `<initial-property-set>`, `<inline>`, `<leader>`, `<page-number>`, `<page-number-citation>`, `<title>`
- shorthand impacting on this property: `font=`

`font-selection-strategy=` (7.8.3; XSL; complete):

- Value: auto | character-by-character | inherit
- Initial: auto
- Inherited: yes
- Media: visual

- specifies if the font selected is done in an implementation-defined manner or if each individual character is checked for the desired font
- font selection is based on `font-family=`, `font-style=`, `font-variant=`, `font-weight=`, `font-stretch=`, `font-size=` and possibly one or more characters in context
- objects to which this property applies: `<bidi-override>`, `<block>`, `<character>`, `<initial-property-set>`, `<inline>`, `<leader>`, `<page-number>`, `<page-number-citation>`, `<title>`

`font-size=` (7.8.4; CSS; basic):

- Value: *absolute-size* | *relative-size* | <length> | <percentage> | inherit
- Initial: medium
- Inherited: yes, the computed value is inherited
- Percentages: refer to parent element's font size
- Media: visual

- possible values for *absolute-size* are: "`xx-small`", "`x-small`", "`small`", "`medium`", "`large`", "`x-large`" and "`xx-large`"
- possible values for *relative-size* are: "`larger`" and "`smaller`"
- objects to which this property applies: `<bidi-override>`, `<block>`, `<character>`, `<initial-property-set>`, `<inline>`, `<leader>`, `<page-number>`, `<page-number-citation>`, `<title>`
- shorthand impacting on this property: `font=`

`font-size-adjust=` (7.8.6; CSS; extended):

- Value: <number> | none | inherit
- Initial: none
- Inherited: yes
- Media: visual

- a number value specifies the aspect (font size divided by x-height)
  - smaller fonts with larger values are typically easier to read than those with smaller values
- note that child elements will inherit unadjusted values because inheritance is based on computed values
- objects to which this property applies: `<bidi-override>`, `<block>`, `<character>`, `<initial-property-set>`, `<inline>`, `<leader>`, `<page-number>`, `<page-number-citation>`, `<title>`

`font-stretch=` (7.8.5; CSS; extended):

- Value: normal | wider | narrower | ultra-condensed | extra-condensed | condensed | semi-condensed | semi-expanded | expanded | extra-expanded | ultra-expanded | inherit
- Initial: normal
- Inherited: yes
- Media: visual

- selects a normal, condensed or extended face from a font family
- values of "`wider`" and "`narrower`" do not go beyond the respective values of "`ultra-expanded`" and "`ultra-condensed`"
- objects to which this property applies: `<bidi-override>`, `<block>`, `<character>`, `<initial-property-set>`, `<inline>`, `<leader>`, `<page-number>`, `<page-number-citation>`, `<title>`

`font-style=` (7.8.7; CSS; basic):

- Value: normal | italic | oblique | backslant | inherit
- Initial: normal
- Inherited: yes
- Media: visual

- note that "`italic`" will match "`oblique`" if no italic face is available in the font family
- objects to which this property applies: `<bidi-override>`, `<block>`, `<character>`, `<initial-property-set>`, `<inline>`, `<leader>`, `<page-number>`, `<page-number-citation>`, `<title>`
- shorthand impacting on this property: `font=`

`font-variant=` (7.8.8; CSS; basic):
- Value: normal | small-caps | inherit
- Inherited: yes
- Initial: normal
- Media: visual
- objects to which this property applies: `<bidi-override>`, `<block>`, `<character>`, `<initial-property-set>`, `<inline>`, `<leader>`, `<page-number>`, `<page-number-citation>`, `<title>`
- shorthand impacting on this property: `font=`

`font-weight=` (7.8.9; CSS; basic):
- Value: normal | bold | bolder | lighter | 100 | 200 | 300 | 400 | 500 | 600 | 700 | 800 | 900 | inherit
- Inherited: yes
- Media: visual
- Initial: normal
- "`normal`" maps to a value of "`400`"
- "`500`" will map to a medium weight value (if available for the font)
- "`bold`" maps to a value of "`700`"
- "`lighter`" and "`bolder`" will find the closest font that is respectively different from the inherited font weight
- objects to which this property applies: `<bidi-override>`, `<block>`, `<character>`, `<initial-property-set>`, `<inline>`, `<leader>`, `<page-number>`, `<page-number-citation>`, `<title>`
- shorthand impacting on this property: `font=`

`force-page-count=` (7.25.6; XSL; extended):
- Value: auto | even | odd | end-on-even | end-on-odd | no-force | inherit
- Inherited: no
- Media: visual
- Initial: auto
- object to which this property applies: `<page-sequence>`

`format=` (7.24.1; XSL; basic):
- Value: <string>
- Inherited: no
- Initial: 1
- Media: all
- see Format tokens from XSLT (page 158) for details
- object to which this property applies: `<page-sequence>`

`glyph-orientation-horizontal=` (7.27.2; XSL; extended):
- Value: <angle> | inherit
- Inherited: yes
- Initial: 0deg
- Media: visual
- applied only to text written in a writing-mode with a left-to-write or right-to-left inline-progression-direction
- the only allowable angles are 0, 90, 180 and 270 degrees, counted clockwise
- "`0deg`" indicates the top of the glyph is towards the top of the reference-area
- object to which this property applies: `<character>`

`glyph-orientation-vertical=` (7.27.3; XSL; extended):
- Value: auto | <angle> | inherit
- Inherited: yes
- Initial: auto
- Media: visual

- applied only to text written in a writing-mode with a top-to-bottom or bottom-to-top inline-progression-direction
- the only allowable angles are 0, 90, 180 and 270 degrees, counted clockwise
- "`0deg`" indicates the top of the glyph is towards the top of the reference-area
- commonly used to differentiate between the preferred orientation of alphabetic text in vertically written Japanese documents ("`auto`") vs. the orientation of alphabetic text in western signage and advertising ("`0deg`")
- object to which this property applies: `<character>`

`grouping-separator=` (7.24.2; XSL; extended):

- Value: <character>
- Initial: no separator
- Inherited: no
- Media: all

- see Format tokens from XSLT (page 158) for details
- object to which this property applies: `<page-sequence>`

`grouping-size=` (7.24.3; XSL; extended):

- Value: <number>
- Initial: no grouping
- Inherited: no
- Media: all

- see Format tokens from XSLT (page 158) for details
- object to which this property applies: `<page-sequence>`

`height=` (7.14.4; CSS; basic):

- Value: <length> | <percentage> | auto | inherit
- Initial: auto
- Inherited: no
- Percentages: see prose
- Media: visual

- percentages are based on the explicit height of the containing block and are ignored if that height is not explicit
- negative values are illegal
- objects to which this property applies: `<block-container>`, `<external-graphic>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<table>`, `<table-caption>`, `<table-cell>`, `<table-row>`

`hyphenate=` (7.9.4; XSL; extended):

- Value: false | true | inherit
- Initial: false
- Inherited: yes
- Media: visual

- specifies if hyphenation may or may not be used in the line-breaking algorithm for the text in the object
- objects to which this property applies: `<block>`, `<character>`

`hyphenation-character=` (7.9.5; XSL; extended):

- Value: <character> | inherit
- Initial: The Unicode hyphen character U+2010
- Inherited: yes
- Media: visual

- specifies the hyphenation character to be used when a hyphenation break occurs
- objects to which this property applies: `<block>`, `<character>`

`hyphenation-keep=` (7.15.1; XSL; extended):

- Value: auto | column | page | inherit
- Initial: auto
- Inherited: yes
- Media: visual

- specifies whether hyphenation can be performed on the last line that fits in a given reference-area
- object to which this property applies: `<block>`

`hyphenation-ladder-count=` (7.15.2; XSL; extended):

- Value: no-limit | <number> | inherit
- Initial: no-limit
- Inherited: yes
- Media: visual

- specifies a limit on the number of successive hyphenated line-areas the formatter may generate in a block-area
- object to which this property applies: `<block>`

`hyphenation-push-character-count=` (7.9.6; XSL; extended):

- Value: <number> | inherit
- Initial: 2
- Inherited: yes
- Media: visual

- specifies the minimum number of characters in a hyphenated word after the hyphenation character
- objects to which this property applies: `<block>`, `<character>`

`hyphenation-remain-character-count=` (7.9.7; XSL; extended):

- Value: <number> | inherit
- Initial: 2
- Inherited: yes
- Media: visual

- specifies the minimum number of characters in a hyphenated word before the hyphenation character
- objects to which this property applies: `<block>`, `<character>`

`id=` (7.28.2; XSL; basic):

- Value: <id>
- Initial: see prose
- Inherited: no, see prose
- Media: all

- objects to which this property applies: `<basic-link>`, `<bidi-override>`, `<block>`, `<block-container>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<list-item-body>`, `<list-item-label>`, `<multi-case>`, `<multi-properties>`, `<multi-property-set>`, `<multi-switch>`, `<multi-toggle>`, `<page-number>`, `<page-number-citation>`, `<page-sequence>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<wrapper>`

`indicate-destination=` (7.22.7; XSL; extended):

- Value: true | false
- Initial: false
- Inherited: no
- Media: interactive

- specifies that areas that belong to the link target when traversed should, in a system-dependent manner, be indicated
- object to which this property applies: `<basic-link>`

`initial-page-number=` (7.25.7; XSL; basic):

- Value: auto | auto-odd | auto-even | <number> | inherit
- Initial: auto
- Inherited: no
- Media: visual
- object to which this property applies: `<page-sequence>`

`inline-progression-dimension=` (7.14.5; CSS; basic):

- Value: auto | <length> | <percentage> | <length-range> | inherit
- Initial: auto
- Inherited: no
- Percentages: see prose
- Media: visual
- a range allows the size to be adjusted by the formatter
- this does not apply when `line-height=` applies to the same dimension of the areas generated by this formatting object
- objects to which this property applies: `<block-container>`, `<external-graphic>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<table>`, `<table-caption>`, `<table-cell>`

`internal-destination=` (7.22.8; XSL; extended):

- Value: empty string | <idref>
- Initial: empty string
- Inherited: no
- Media: interactive
- at least one of `external-destination=` and `internal-destination=` properties should be assigned
- if both are assigned, the system may either report the error, or use `internal-destination=`
- object to which this property applies: `<basic-link>`

`intrusion-displace=` (7.18.3; XSL; extended):

- Value: auto | none | line | indent | block | inherit
- Initial: auto
- Inherited: yes
- Media: visual
- this specifies the displacement strategy in the presence of intrusions (e.g. side floats)
- "`none`" allows any present intrusion to overlay the given line or block areas
- "`block`" reduces the inline progression dimension of the entire block that is in any way affected by the presence of an intrusion
- "`line`" reduces the inline progression dimension of only those line areas affected by the presence of an intrusion
- "`indent`" will act as "`line`" but will also preserve for each line any indentation relative to all other lines impacted by the given intrusion
- "`auto`"assumes "`block`" for reference-areas and "`line`"for other areas
- objects to which this property applies: `<block>`, `<block-container>`, `<list-block>`, `<list-item>`, `<table-and-caption>`, `<table>`, `<table-caption>`

`keep-together=` (7.19.3; XSL; extended):

- Value: <keep> | inherit
- Initial: .within-line=auto, .within-column=auto, .within-page=auto
- Inherited: yes
- Media: visual

- objects to which this property applies: `<basic-link>`, `<block>`, `<block-container>`, `<inline>`, `<inline-container>`, `<list-block>`, `<list-item>`, `<list-item-body>`, `<list-item-label>`, `<table-and-caption>`, `<table>`, `<table-caption>`, `<table-row>`
- shorthand impacting on this property: `page-break-inside=`

`keep-with-next=` (7.19.4; XSL; basic):

- Value: <keep> | inherit
- Initial: .within-line=auto, .within-column=auto, .within-page=auto
- Inherited: no
- Media: visual
- objects to which this property applies: `<basic-link>`, `<block>`, `<block-container>`, `<character>`, `<external-graphic>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<table-and-caption>`, `<table>`, `<table-row>`
- shorthands impacting on this property: `page-break-after=`, `page-break-before=`

`keep-with-previous=` (7.19.5; XSL; basic):

- Value: <keep> | inherit
- Initial: .within-line=auto, .within-column=auto, .within-page=auto
- Inherited: no
- Media: visual
- objects to which this property applies: `<basic-link>`, `<block>`, `<block-container>`, `<character>`, `<external-graphic>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<table-and-caption>`, `<table>`, `<table-row>`

`language=` (7.9.2; XSL; extended):

- Value: none | <language> | inherit
- Initial: none
- Inherited: yes
- Media: visual
- specifies the language used in language- and locale- coupled services
  - e.g. line-justification, line-breaking, hyphenation, etc.
- objects to which this property applies: `<block>`, `<character>`, `<page-sequence>`
- shorthand impacting on this property: `xml:lang=`

`last-line-end-indent=` (7.15.3; XSL; extended):

- Value: <length> | <percentage> | inherit
- Initial: 0pt
- Inherited: yes
- Percentages: refer to inline-progression-dimension of closest ancestor block-area that is not a line-area
- Media: visual
- positive values indent the end edge, negative values outdent the end edge
- object to which this property applies: `<block>`

`leader-alignment=` (7.21.1; XSL; extended):

- Value: none | reference-area | page | inherit
- Initial: none
- Inherited: yes
- Media: visual

- specifies whether all `<leader>` objects having identical content and property values shall have their patterns aligned with each other, with respect to their common reference-area or page
- "`reference-area`" specifies the pattern is aligned to the content-rectangle start-edge
- "`page`" specifies the pattern is aligned to the page's start-edge
- object to which this property applies: `<leader>`

`leader-length=` (7.21.4; XSL; basic):

- Value: <length-range> | <percentage> | inherit
- Initial: leader-length.minimum=0pt, .optimum=12.0pt, .maximum=100%
- Inherited: yes
- Percentages: refer to the inline-progression-dimension of content-rectangle of parent area
- Media: visual

- object to which this property applies: `<leader>`

`leader-pattern=` (7.21.2; XSL; basic):

- Value: space | rule | dots | use-content | inherit
- Initial: space
- Inherited: yes
- Media: visual

- "`use-content`" specifies the children of the `<leader>` comprise the repeating pattern
- "`space`", "`rule`" and "`dots`" are available in all implementations
- an implementation may interpret "`use-content`" as "`space`"
- object to which this property applies: `<leader>`

`leader-pattern-width=` (7.21.3; XSL; extended):

- Value: use-font-metrics | <length> | <percentage> | inherit
- Initial: use-font-metrics
- Inherited: yes
- Percentages: refer to the inline-progression-dimension of content-rectangle of parent area
- Media: visual

- object to which this property applies: `<leader>`

`left=` (7.5.5; CSS; extended):

- Value: <length> | <percentage> | auto | inherit
- Initial: auto
- Inherited: no
- Percentages: refer to width of containing block
- Media: visual

- specifies the distance between the margin edge and the containing block
- a value other than "`auto`" overrides a `width=` value of "`auto`"
- the `end-indent=` property is adjusted to correspond to any specified geometry of the content rectangle
- the percentage is ignored if the containing block width is not specified explicitly
- object to which this property applies: `<block-container>`

`letter-spacing=` (7.16.2; CSS; extended):

- Value: normal | <length> | <space> | inherit
- Initial: normal
- Inherited: yes
- Media: visual

- specifies an amount to add (may be negative) to the built-in inter-character spacing
- specifying a value other than "`normal`" turns off the use of ligatures
- specifying a length turns off the modifications of inter-character spacing, but not the inter-word spacing, for justifying text
- objects to which this property applies: `<bidi-override>`, `<character>`, `<initial-property-set>`, `<leader>`, `<page-number>`, `<page-number-citation>`

`letter-value=` (7.24.4; XSL; basic):

- Value: auto | alphabetic | traditional
- Initial: auto
- Inherited: no
- Media: all

- see Format tokens from XSLT (page 158) for details
- "`auto`" corresponds to an unspecified attribute in XSLT
- object to which this property applies: `<page-sequence>`

`line-height=` (7.15.4; CSS; basic):

- Value: normal | <length> | <number> | <percentage> | <space> | inherit
- Initial: normal
- Inherited: yes
- Percentages: refer to the font size of the element itself
- Media: visual

- specifies the minimal height for a block-level construct of each generated inline box
- specifies the exact height for an inline-level construct
- the value "`normal`" is implementation specific, typically between 1.0 and 1.2
- objects to which this property applies: `<basic-link>`, `<bidi-override>`, `<block>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<page-number>`, `<page-number-citation>`, `<title>`
- shorthand impacting on this property: `font=`

`line-height-shift-adjustment=` (7.15.5; XSL; extended):

- Value: consider-shifts | disregard-shifts | inherit
- Initial: consider-shifts
- Inherited: yes
- Media: visual

- specifies if the line-height of the line is adjusted for content that has a baseline shift
- a value of "`disregard-shifts`" will prevent superscript and subscript characters from disrupting the line-spacing
- see Chapter 4 Area and page basics (page 70) for more details on half-leading
- see Chapter 4 Area and page basics (page 70) for more details on superscripts and subscripts
- object to which this property applies: `<block>`

`line-stacking-strategy=` (7.15.6; XSL; basic):

- Value: line-height | font-height | max-height | inherit
- Initial: max-height
- Inherited: yes
- Media: visual

- specifies the strategy for positioning adjacent lines relative to each other
- see Chapter 4 Area and page basics (page 70) for details
- "`line-height`" may be interpreted as "`max-height`"
- object to which this property applies: `<block>`

`linefeed-treatment=` (7.15.7; XSL; extended):

- Value: ignore | preserve | treat-as-space | treat-as-zero-width-space | inherit
- Initial: treat-as-space
- Inherited: yes
- Media: visual
- object to which this property applies: `<block>`

`margin=` (7.29.14; CSS; shorthand):

- Value: *margin-width*{1,4} | inherit
- Initial: not defined for shorthand properties
- Inherited: no
- Percentages: refer to width of containing block
- Media: visual
- *margin-width* is one of <length>, <percentage> or "`auto`"
- up to four individual values for margins can be specified in this shorthand
- four values specifies each of the margins in the order: `margin-top=`, `margin-right=`, `margin-bottom=`, `margin-left=`
- three values specifies the margins in the order: `margin-top=`, `margin-right=`/`margin-left=`, `margin-bottom=`
- two values specifies the margins in the order: `margin-top=`/`margin-bottom=`, `margin-right=`/`margin-left=`
- one value specifies all the margins

`margin-bottom=` (7.10.2; CSS; basic):

- Value: *margin-width* | inherit
- Initial: 0pt
- Inherited: no
- Percentages: refer to width of containing block
- Media: visual
- *margin-width* is one of <length>, <percentage> or "`auto`"
- objects to which this property applies: `<block>`, `<block-container>`, `<list-block>`, `<list-item>`, `<region-body>`, `<simple-page-master>`, `<table-and-caption>`, `<table>`
- shorthand impacting on this property: `margin=`

`margin-left=` (7.10.3; CSS; basic):

- Value: *margin-width* | inherit
- Initial: 0pt
- Inherited: no
- Percentages: refer to width of containing block
- Media: visual
- *margin-width* is one of <length>, <percentage> or "`auto`"
- objects to which this property applies: `<block>`, `<block-container>`, `<list-block>`, `<list-item>`, `<region-body>`, `<simple-page-master>`, `<table-and-caption>`, `<table>`
- shorthand impacting on this property: `margin=`

`margin-right=` (7.10.4; CSS; basic):

- Value: *margin-width* | inherit
- Initial: 0pt
- Inherited: no
- Percentages: refer to width of containing block
- Media: visual
- *margin-width* is one of <length>, <percentage> or "`auto`"
- objects to which this property applies: `<block>`, `<block-container>`, `<list-block>`, `<list-item>`, `<region-body>`, `<simple-page-master>`, `<table-and-caption>`, `<table>`
- shorthand impacting on this property: `margin=`

`margin-top=` (7.10.1; CSS; basic):

- Value: *margin-width* | inherit
- Initial: 0pt
- Inherited: no
- Percentages: refer to width of containing block
- Media: visual
- *margin-width* is one of <length>, <percentage> or "`auto`"
- objects to which this property applies: `<block>`, `<block-container>`, `<list-block>`, `<list-item>`, `<region-body>`, `<simple-page-master>`, `<table-and-caption>`, `<table>`
- shorthand impacting on this property: `margin=`

`marker-class-name=` (7.23.1; XSL; extended):

- Value: <name>
- Initial: an empty name
- Inherited: no, a value is required
- Media: paged
- identifies the `<marker>` as being in a group with others that have the same name
- object to which this property applies: `<marker>`

`master-name=` (7.25.8; XSL; basic):

- Value: <name>
- Initial: an empty name
- Inherited: no, a value is required
- Media: visual
- objects to which this property applies: `<page-sequence-master>`, `<simple-page-master>`

`master-reference=` (7.25.9; XSL; basic):

- Value: <name>
- Initial: an empty name
- Inherited: no, a value is required
- Media: visual
- objects to which this property applies: `<conditional-page-master-reference>`, `<page-sequence>`, `<repeatable-page-master-reference>`, `<single-page-master-reference>`

`max-height=` (7.14.6; CSS; complete):

- Value: <length> | <percentage> | none | inherit
- Initial: 0pt
- Inherited: no
- Percentages: refer to height of containing block
- Media: visual

- percentages are based on the explicit height of the containing block and are ignored if that height is not explicit

`max-width=` (7.14.7; CSS; complete):

- Value: <length> | <percentage> | none | inherit
- Initial: none
- Inherited: no
- Percentages: refer to width of containing block
- Media: visual

- this property is mapped to either the `inline-progression-dimension=` or the `block-progression-dimension=`, based on the applicable values of the `writing-mode=` and `reference-orientation=` properties

`maximum-repeats=` (7.25.10; XSL; extended):

- Value: <number> | no-limit | inherit
- Initial: no-limit
- Inherited: no
- Media: visual

- objects to which this property applies: `<repeatable-page-master-alternatives>`, `<repeatable-page-master-reference>`

`media-usage=` (7.25.11; XSL; extended):

- Value: auto | paginate | bounded-in-one-dimension | unbounded
- Initial: auto
- Inherited: no
- Percentages: NA
- Media: visual

- specifies how the selected display medium is used to present the page(s) specified by the stylesheet
- "`unbounded`" generates only one page per `<page-sequence>` and requires that neither `page-height=` nor `page-width=` be specified
- "`bounded-in-one-dimension`" generates only one page per `<page-sequence>` and requires exactly one of either `page-height=` or `page-width=` to be specified on the first page master that is used
- object to which this property applies: `<root>`

`min-height=` (7.14.8; CSS; complete):

- Value: <length> | <percentage> | inherit
- Initial: 0pt
- Inherited: no
- Percentages: refer to height of containing block
- Media: visual

- percentages are based on the explicit height of the containing block and are ignored if that height is not explicit

`min-width=` (7.14.9; CSS; complete):

- Value: <length> | <percentage> | inherit
- Initial: depends on UA
- Inherited: no
- Percentages: refer to width of containing block
- Media: visual

- percentages are based on the explicit height of the containing block and are ignored if that height is not explicit

`number-columns-repeated=` (7.26.12; XSL; basic):

- Value: <number>
- Inherited: no
- Initial: 1
- Media: visual
- object to which this property applies: `<table-column>`

`number-columns-spanned=` (7.26.13; XSL; basic):

- Value: <number>
- Inherited: no
- Initial: 1
- Media: visual
- objects to which this property applies: `<table-cell>`, `<table-column>`

`number-rows-spanned=` (7.26.14; XSL; basic):

- Value: <number>
- Inherited: no
- Initial: 1
- Media: visual
- object to which this property applies: `<table-cell>`

`odd-or-even=` (7.25.12; XSL; extended):

- Value: odd | even | any | inherit
- Inherited: no
- Initial: any
- Media: visual
- object to which this property applies: `<conditional-page-master-reference>`

`orphans=` (7.19.6; CSS; basic):

- Value: <integer> | inherit
- Inherited: yes
- Initial: 2
- Media: visual
- specifies the minimum number of lines of a block that must be left at the bottom of a page
- object to which this property applies: `<block>`

`overflow=` (7.20.2; CSS; basic):

- Value: visible | hidden | scroll | error-if-overflow | auto | inherit
- Inherited: no
- Media: visual
- Initial: auto
- "`visible`" allows overflow content to be rendered outside the box
- "`hidden`" clips overflow content
- "`scroll`" provides a scrolling region for the entire content
- "`error-if-overflow`" is the same as "`hidden`" but also triggers an error
- objects to which this property applies: `<block-container>`, `<external-graphic>`, `<inline-container>`, `<instream-foreign-object>`, `<region-after>`, `<region-before>`, `<region-body>`, `<region-end>`, `<region-start>`

`padding=` (7.29.15; CSS; shorthand):

- Value: *padding-width*{1,4} | inherit
- Inherited: no
- Initial: not defined for shorthand properties
- Percentages: refer to width of containing block
- Media: visual

- *padding-width* is one of <length> or <percentage>
- up to four individual values for padding values can be specified in this shorthand
- four values specifies each of the padding values in the order: `padding-top=`, `padding-right=`, `padding-bottom=`, `padding-left=`
- three values specifies the padding values in the order: `padding-top=`, `padding-right=`/`padding-left=`, `padding-bottom=`
- two values specifies the padding values in the order: `padding-top=`/`padding-bottom=`, `padding-right=`/`padding-left=`
- one value specifies all the padding values

`padding-after=` (7.7.32; CSS; basic):

- Value: *padding-width* | <length-conditional> | inherit
- Initial: 0pt
- Inherited: no
- Percentages: refer to width of containing block
- Media: visual

- *padding-width* is one of <length> or <percentage>
- objects to which this property applies: `<basic-link>`, `<block>`, `<block-container>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<region-after>`, `<region-before>`, `<region-body>`, `<region-end>`, `<region-start>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-column>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`

`padding-before=` (7.7.31; CSS; basic):

- Value: *padding-width* | <length-conditional> | inherit
- Initial: 0pt
- Inherited: no
- Percentages: refer to width of containing block
- Media: visual

- *padding-width* is one of <length> or <percentage>
- objects to which this property applies: `<basic-link>`, `<block>`, `<block-container>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<region-after>`, `<region-before>`, `<region-body>`, `<region-end>`, `<region-start>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-column>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`

`padding-bottom=` (7.7.36; CSS; basic):

- Value: *padding-width* | inherit
- Initial: 0pt
- Inherited: no
- Percentages: refer to width of containing block
- Media: visual

- *padding-width* is one of <length> or <percentage>
- objects to which this property applies: `<basic-link>`, `<block>`, `<block-container>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<region-after>`, `<region-before>`, `<region-body>`, `<region-end>`, `<region-start>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-column>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`
- shorthand impacting on this property: `padding=`

`padding-end=` (7.7.34; CSS; basic):

- Value: *padding-width* | <length-conditional> | inherit
- Initial: 0pt
- Inherited: no
- Percentages: refer to width of containing block
- Media: visual

- *padding-width* is one of <length> or <percentage>
- objects to which this property applies: `<basic-link>`, `<block>`, `<block-container>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<region-after>`, `<region-before>`, `<region-body>`, `<region-end>`, `<region-start>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-column>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`

`padding-left=` (7.7.37; CSS; basic):

- Value: *padding-width* | inherit
- Initial: 0pt
- Inherited: no
- Percentages: refer to width of containing block
- Media: visual

- *padding-width* is one of <length> or <percentage>
- objects to which this property applies: `<basic-link>`, `<block>`, `<block-container>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<region-after>`, `<region-before>`, `<region-body>`, `<region-end>`, `<region-start>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-column>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`
- shorthand impacting on this property: `padding=`

`padding-right=` (7.7.38; CSS; basic):

- Value: *padding-width* | inherit
- Initial: 0pt
- Inherited: no
- Percentages: refer to width of containing block
- Media: visual

- *padding-width* is one of <length> or <percentage>
- objects to which this property applies: `<basic-link>`, `<block>`, `<block-container>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<region-after>`, `<region-before>`, `<region-body>`, `<region-end>`, `<region-start>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-column>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`
- shorthand impacting on this property: `padding=`

`padding-start=` (7.7.33; CSS; basic):

- Value: *padding-width* | <length-conditional> | inherit
- Initial: 0pt
- Inherited: no
- Percentages: refer to width of containing block
- Media: visual

- *padding-width* is one of <length> or <percentage>
- objects to which this property applies: `<basic-link>`, `<block>`, `<block-container>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<region-after>`, `<region-before>`, `<region-body>`, `<region-end>`, `<region-start>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-column>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`

`padding-top=` (7.7.35; CSS; basic):

- Value: *padding-width* | inherit
- Initial: 0pt
- Inherited: no
- Percentages: refer to width of containing block
- Media: visual

- *padding-width* is one of <length> or <percentage>
- objects to which this property applies: `<basic-link>`, `<block>`, `<block-container>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<region-after>`, `<region-before>`, `<region-body>`, `<region-end>`, `<region-start>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-column>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`
- shorthand impacting on this property: `padding=`

`page-break-after=` (7.29.16; CSS; shorthand):

- Value: auto | always | avoid | left | right | inherit
- Initial: auto
- Inherited: no
- Media: visual

Value	break-after=	keep-with-next=
"auto"	"auto"	"auto"
"always"	"page"	"auto"
"avoid"	"auto"	"always"
"left"	"even-page"	"auto"
"right"	"odd-page"	"auto"

`page-break-before=` (7.29.17; CSS; shorthand):

- Value: auto | always | avoid | left | right | inherit
- Initial: auto
- Inherited: no
- Media: visual

Value	break-before=	keep-with-previous=
"auto"	"auto"	"auto"
"always"	"page"	"auto"
"avoid"	"auto"	"always"
"left"	"even-page"	"auto"
"right"	"odd-page"	"auto"

`page-break-inside=` (7.29.18; CSS; shorthand):

- Value: avoid | auto | inherit
- Initial: auto
- Inherited: yes
- Media: visual

- "auto"infers `keep-together=` of "auto"
- "avoid" infers `keep-together=` of "always"

`page-height=` (7.25.13; XSL; basic):

- Value: auto | indefinite | <length> | inherit
- Initial: auto
- Inherited: no
- Media: visual

- "auto" determines the value from the window (for continues media) or the size of the page
- "indefinite" determines the value from the laid-out content
  - overrides a value of "indefinite" for `page-width=` to be "auto"
- object to which this property applies: `<simple-page-master>`
- shorthand impacting on this property: `size=`

`page-position=` (7.25.14; XSL; extended):

- Value: first | last | rest | any | inherit
- Initial: any
- Inherited: no
- Media: visual

- object to which this property applies: `<conditional-page-master-reference>`

`page-width=` (7.25.15; XSL; basic):

- Value: auto | indefinite | <length> | inherit
- Initial: auto
- Inherited: no
- Media: visual

- "auto" determines the value from the window (for continues media) or the size of the page
- "indefinite" determines the value from the laid-out content
  - overridden to be "auto" by a value of "indefinite" for `page-height=`
- object to which this property applies: `<simple-page-master>`
- shorthand impacting on this property: `size=`

`pause=` (7.29.19; CSS; shorthand):

- Value: [<time> | <percentage>]{1,2} | inherit
- Initial: depends on user agent
- Inherited: no
- Percentages: see descriptions of 'pause-before' and 'pause-after'
- Media: aural
- a shorthand specifying, in order, `pause-before=` and `pause-after=`
  - specifying one value applies to both properties

`pause-after=` (7.6.5; CSS; basic):

- Value: <time> | <percentage> | inherit
- Initial: depends on user agent
- Inherited: no
- Percentages: see prose
- Media: aural
- specifies the pause to be observed after speaking an object's content
- objects to which this property applies: `<basic-link>`, `<bidi-override>`, `<block>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`
- shorthand impacting on this property: `pause=`

`pause-before=` (7.6.6; CSS; basic):

- Value: <time> | <percentage> | inherit
- Initial: depends on user agent
- Inherited: no
- Percentages: see prose
- Media: aural
- specifies the pause to be observed after speaking an object's content
- objects to which this property applies: `<basic-link>`, `<bidi-override>`, `<block>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`
- shorthand impacting on this property: `pause=`

`pitch=` (7.6.7; CSS; basic):

- Value: <frequency> | x-low | low | medium | high | x-high | inherit
- Initial: medium
- Inherited: yes
- Media: aural
- specifies the average pitch (a frequency) of the speaking voice
- objects to which this property applies: `<basic-link>`, `<bidi-override>`, `<block>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`

`pitch-range=` (7.6.8; CSS; basic):

- Value: <number> | inherit
- Initial: 50
- Inherited: yes
- Media: aural

- specifies variation in average pitch
- a highly animated voice displays a high pitch range
    - i.e. one that is heavily inflected
- objects to which this property applies: `<basic-link>`, `<bidi-override>`, `<block>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`

`play-during`= (7.6.9; CSS; basic):

- Value: <uri-specification> mix? repeat? | auto | none | inherit
- Initial: auto
- Inherited: no
- Media: aural

- <uri-specification> specifies the sound played as a background while the object's content is spoken
- "`mix`" specifies the sound inherited from the parent object is mixed with the object's background <uri-specification>
- "`repeat`" specifies a short background to be repeated while the object is being spoken
- "`auto`" specifies the sound of the parent's background is to be continued
- "`none`" specifies silence for the background of the object being spoken
- objects to which this property applies: `<basic-link>`, `<bidi-override>`, `<block>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`

`position`= (7.29.20; CSS; shorthand):

- Value: static | relative | absolute | fixed | inherit
- Initial: static
- Inherited: no
- Media: visual

- Value

	`relative-position=`	`absolute-position=`
`"static"`	`"static"`	`"auto"`
`"relative"`	`"relative"`	`"auto"`
`"absolute"`	`"static"`	`"absolute"`
`"fixed"`	`"static"`	`"fixed"`

`precedence`= (7.25.16; XSL; extended):

- Value: true | false | inherit
- Initial: false
- Inherited: no
- Media: visual

- specifies whether the inline-progression-dimension of the region extends to content-rectangle of the page-reference-area ("`true`") or only to the edges incurred by the adjacent regions ("`false`")
- objects to which this property applies: `<region-after>`, `<region-before>`

`provisional-distance-between-starts=` (7.28.4; XSL; basic):

- Value: <length> | <percentage> | inherit
- Initial: 24.0pt
- Inherited: yes
- Percentages: refer to inline-progression-dimension of closest ancestor block-area that is not a line-area
- Media: visual
- see Chapter 5 Generic body constructs (page 76) for a discussion of this property
- object to which this property applies: `<list-block>`

`provisional-label-separation=` (7.28.3; XSL; basic):

- Value: <length> | <percentage> | inherit
- Initial: 6.0pt
- Inherited: yes
- Percentages: refer to inline-progression-dimension of closest ancestor block-area that is not a line-area
- Media: visual
- see Chapter 5 Generic body constructs (page 76) for a discussion of this property
- object to which this property applies: `<list-block>`

`ref-id=` (7.28.5; XSL; extended):

- Value: <idref> | inherit
- Initial: none, value required
- Inherited: no
- Media: all
- object to which this property applies: `<page-number-citation>`

`reference-orientation=` (7.20.3; XSL; extended):

- Value: 0 | 90 | 180 | 270 | -90 | -180 | -270 | inherit
- Initial: 0
- Inherited: yes (see prose)
- Media: visual
- degree values are clockwise from "`0`" degrees at the top
  - note this is a simple <integer> and not an <angle> data type value
- objects to which this property applies: `<block-container>`, `<inline-container>`, `<region-after>`, `<region-before>`, `<region-body>`, `<region-end>`, `<region-start>`, `<simple-page-master>`

`region-name=` (7.25.17; XSL; basic):

- Value: xsl-region-body | xsl-region-start | xsl-region-end | xsl-region-before | xsl-region-after | xsl-before-float-separator | xsl-footnote-separator | <name>
- Initial: see prose
- Inherited: no, a value is required
- Media: visual
- objects to which this property applies: `<region-after>`, `<region-before>`, `<region-body>`, `<region-end>`, `<region-start>`

`relative-align=` (7.13.6; XSL; extended):

- Value: before | baseline | inherit
- Initial: before
- Inherited: yes
- Media: visual

- specifies the alignment, in the block-progression-direction, between two or more areas
- objects to which this property applies: `<list-item>`, `<table-cell>`

`relative-position=` (7.12.1; CSS; extended):

- Value: static | relative | inherit
- Initial: static
- Inherited: no
- Media: visual

- "`static`" stacks the area normally
- "`relative`" positions an area as if it were stacked, but the area does not affect the position of any other area
  - any such area that breaks over a page boundary is clipped to the page and the remainder is discarded
- objects to which this property applies: `<basic-link>`, `<bidi-override>`, `<block>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-footer>`, `<table-header>`, `<table-row>`
- shorthand impacting on this property: `position=`

`rendering-intent=` (7.17.3; XSL; extended):

- Value: auto | perceptual | relative-colorimetric | saturation | absolute-colorimetric | inherit
- Initial: auto
- Inherited: no
- Media: visual

- this property is applicable primarily to color-profiles corresponding to CMYK color spaces
- the different options cause different methods to be used for translating colors to the color gamut of the target rendering device
- object to which this property applies: `<color-profile>`

`retrieve-boundary=` (7.23.4; XSL; extended):

- Value: page | page-sequence | document
- Initial: page-sequence
- Inherited: no
- Media: paged

- specifies how far "back" in the flow the formatter will look for a marker, starting with the current page
- object to which this property applies: `<retrieve-marker>`

`retrieve-class-name=` (7.23.2; XSL; extended):

- Value: <name>
- Initial: an empty name
- Inherited: no, a value is required
- Media: paged

- constrains that the `<marker>` whose children are retrieved by the `<retrieve-marker>` must have a `marker-class-name=` property value that is the same as the value of this property
- object to which this property applies: `<retrieve-marker>`

`retrieve-position=` (7.23.3; XSL; extended):

- Inherited: no
- Media: paged

- Value: first-starting-within-page | first-including-carryover | last-starting-within-page | last-ending-within-page
- Initial: first-starting-within-page

- specifies the preference for which `<marker>` children shall be retrieved by a `<retrieve-marker>`
- object to which this property applies: `<retrieve-marker>`

`richness=` (7.6.10; CSS; basic):

- Value: <number> | inherit
- Initial: 50
- Inherited: yes
- Media: aural

- specifies the richness, or brightness, of the speaking voice
- a rich voice will "carry" in a large room, a smooth voice will not
  - the term "smooth" refers to how the wave form looks when drawn
- objects to which this property applies: `<basic-link>`, `<bidi-override>`, `<block>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`

`right=` (7.5.3; CSS; extended):

- Value: <length> | <percentage> | auto | inherit
- Initial: auto
- Inherited: no
- Percentages: refer to width of containing block
- Media: visual

- see `left=` for details
- object to which this property applies: `<block-container>`

`role=` (7.4.2; XSL; basic):

- Value: <string> | <uri-specification> | none | inherit
- Initial: none
- Inherited: no
- Media: all

- provides a hint for alternate rendering agents (aural readers, etc.) as to the role of the XML element or elements that were used to construct this formatting object
- <uri-specification> specifies an RDF resource
- objects to which this property applies: `<basic-link>`, `<block>`, `<external-graphic>`, `<footnote>`, `<footnote-body>`, `<initial-property-set>`, `<inline>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<list-item-body>`, `<list-item-label>`, `<multi-case>`, `<multi-properties>`, `<multi-switch>`, `<multi-toggle>`, `<page-number>`, `<page-number-citation>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`

`rule-style=` (7.21.5; XSL; basic):

- Value: none | dotted | dashed | solid | double | groove | ridge | inherit
- Initial: solid
- Inherited: yes
- Media: visual

- "none" and "solid" are always supported
- other values may be interpreted as "solid"
- object to which this property applies: <leader>

rule-thickness= (7.21.6; XSL; basic):

- Value: <length>
- Initial: 1.0pt
- Inherited: yes
- Media: visual

- this only applies if the leader-pattern= is "rule"
- object to which this property applies: <leader>

scaling= (7.14.10; XSL; extended):

- Value: uniform | non-uniform | inherit
- Initial: uniform
- Inherited: no
- Media: visual

- specifies whether scaling needs to preserve the intrinsic aspect ratio
- objects to which this property applies: <external-graphic>, <instream-foreign-object>

scaling-method= (7.14.11; XSL; extended):

- Value: auto | integer-pixels | resample-any-method | inherit
- Initial: auto
- Inherited: no
- Media: visual

- indicates the preference in the scaling/sizing tradeoff to be used when formatting bitmapped graphics
- objects to which this property applies: <external-graphic>, <instream-foreign-object>

score-spaces= (7.28.6; XSL; extended):

- Value: true | false | inherit
- Initial: true
- Inherited: yes
- Media: visual

- specifies whether the text-decoration property shall be applied to spaces
- objects to which this property applies: <bidi-override>, <character>, <initial-property-set>, <page-number>, <page-number-citation>

script= (7.9.3; XSL; extended):

- Value: none | auto | <script> | inherit
- Initial: auto
- Inherited: yes
- Media: visual

- specifies the script used in language- and locale- coupled services
  - e.g. line-justification, line-breaking, hyphenation, etc.
- objects to which this property applies: <block>, <character>

show-destination= (7.22.9; XSL; extended):

- Value: replace | new
- Initial: replace
- Inherited: no
- Media: interactive

- specifies whether the destination resource should replace the current document view or open a new document view
- object to which this property applies: <basic-link>

size= (7.29.21; CSS; shorthand):

- Value: <length>{1,2} | auto | landscape | portrait | inherit
- Initial: auto
- Inherited: N/A [XSL:no, is optional]
- Media: visual

- a shorthand specifying, in order, `page-width=` and `page-height=`
    - specifying one value applies to both properties

`source-document=` (7.4.1; XSL; basic):

- Value: <uri-specification> [<uri-specification>]* | none | inherit
- Initial: none
- Inherited: no
- Media: all

- provides pointers back to the original XML documents used to create the XSL-FO input
- W3C Accessibility guidelines strongly encourage the use of this property either on `<root>` or on the first formatting object from each source document
- objects to which this property applies: `<basic-link>`, `<block>`, `<external-graphic>`, `<footnote>`, `<footnote-body>`, `<initial-property-set>`, `<inline>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<list-item-body>`, `<list-item-label>`, `<multi-case>`, `<multi-properties>`, `<multi-switch>`, `<multi-toggle>`, `<page-number>`, `<page-number-citation>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`

`space-after=` (7.10.6; XSL; basic):

- Value: <space> | inherit
- Initial: space.minimum=0pt, .optimum=0pt, .maximum=0pt, .conditionality=discard, .precedence=0
- Inherited: no
- Percentages: N/A (Differs from margin-bottom in CSS)
- Media: visual

- objects to which this property applies: `<block>`, `<block-container>`, `<list-block>`, `<list-item>`, `<region-body>`, `<simple-page-master>`, `<table-and-caption>`, `<table>`

`space-before=` (7.10.5; XSL; basic):

- Value: <space> | inherit
- Initial: space.minimum=0pt, .optimum=0pt, .maximum=0pt, .conditionality=discard, .precedence=0
- Inherited: no
- Percentages: N/A (Differs from margin-top in CSS)
- Media: visual

- objects to which this property applies: `<block>`, `<block-container>`, `<list-block>`, `<list-item>`, `<region-body>`, `<simple-page-master>`, `<table-and-caption>`, `<table>`

`space-end=` (7.11.1; XSL; basic):

- Value: <space> | <percentage> | inherit
- Initial: space.minimum=0pt, .optimum=0pt, .maximum=0pt, .conditionality=discard, .precedence=0
- Inherited: no
- Percentages: refer to inline-progression-dimension of closest ancestor block-area that is not a line-area
- Media: visual

- objects to which this property applies: `<basic-link>`, `<character>`, `<external-graphic>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<page-number>`, `<page-number-citation>`, `<title>`

`space-start=` (7.11.2; XSL; basic):

- Value: <space> | <percentage> | inherit
- Initial: space.minimum=0pt, .optimum=0pt, .maximum=0pt, .conditionality=discard, .precedence=0
- Inherited: no
- Percentages: refer to inline-progression-dimension of closest ancestor block-area that is not a line-area
- Media: visual
- objects to which this property applies: `<basic-link>`, `<character>`, `<external-graphic>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<leader>`, `<page-number>`, `<page-number-citation>`, `<title>`

`span=` (7.20.4; XSL; extended):

- Value: none | all | inherit
- Initial: none
- Inherited: no
- Media: visual
- objects to which this property applies: `<block>`, `<block-container>`

`speak=` (7.6.11; CSS; basic):

- Value: normal | none | spell-out | inherit
- Initial: normal
- Inherited: yes
- Media: aural
- specifies the manner by which text will be rendered aurally
- "none" suppresses the aural rendering to zero time (unlike setting the volume to zero)
- objects to which this property applies: `<basic-link>`, `<bidi-override>`, `<block>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`

`speak-header=` (7.6.12; CSS; basic):

- Value: once | always | inherit
- Initial: once
- Inherited: yes
- Media: aural
- specifies whether table headers are spoken before every cell, or only before a cell when that cell is associated with a different header than the previous cell
- objects to which this property applies: `<basic-link>`, `<bidi-override>`, `<block>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`

`speak-numeral=` (7.6.13; CSS; basic):

- Value: digits | continuous | inherit
- Initial: continuous
- Inherited: yes
- Media: aural

- specifies how numerals are spoken
- objects to which this property applies: `<basic-link>`, `<bidi-override>`, `<block>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`

`speak-punctuation=` (7.6.14; CSS; basic):

- Value: code | none | inherit
- Initial: none
- Inherited: yes
- Media: aural

- specifies how punctuation is spoken
- note that "`borge`" is not an available option
- objects to which this property applies: `<basic-link>`, `<bidi-override>`, `<block>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`

`speech-rate=` (7.6.15; CSS; basic):

- Value: <number> | x-slow | slow | medium | fast | x-fast | faster | slower | inherit
- Initial: medium
- Inherited: yes
- Media: aural

- <number> specifies the speaking rate in words per minute (varying somewhat by language)
- objects to which this property applies: `<basic-link>`, `<bidi-override>`, `<block>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`

`src=` (7.28.7; XSL; basic):

- Value: <uri-specification> | inherit
- Initial: none, value required
- Inherited: no
- Media: visual

- objects to which this property applies: `<color-profile>`, `<external-graphic>`

`start-indent=` (7.10.7; XSL; basic):

- Value: <length> | <percentage> | inherit
- Initial: 0pt
- Inherited: yes
- Percentages: refer to inline-progression-dimension of containing reference-area
- Media: visual

- objects to which this property applies: <block>, <block-container>, <list-block>, <list-item>, <region-body>, <simple-page-master>, <table-and-caption>, <table>

starting-state= (7.22.10; XSL; extended):

- Value: show | hide
- Initial: show
- Inherited: no
- Media: interactive

- specifies the <multi-case> that can be initially displayed
- object to which this property applies: <multi-case>

starts-row= (7.26.15; XSL; extended):

- Value: true | false
- Initial: false
- Inherited: no
- Media: visual

- object to which this property applies: <table-cell>

stress= (7.6.16; CSS; basic):

- Value: <number> | inherit
- Initial: 50
- Inherited: yes
- Media: aural

- specifies the height of "local peaks" in the intonation contour of a voice
- objects to which this property applies: <basic-link>, <bidi-override>, <block>, <character>, <external-graphic>, <initial-property-set>, <inline>, <instream-foreign-object>, <leader>, <list-block>, <list-item>, <page-number>, <page-number-citation>, <table-and-caption>, <table>, <table-body>, <table-caption>, <table-cell>, <table-footer>, <table-header>, <table-row>, <title>

suppress-at-line-break= (7.16.3; XSL; extended):

- Value: auto | suppress | retain | inherit
- Initial: auto
- Inherited: no
- Media: visual

- "auto" will suppress a U+0020 space character if it is first or last in a line
- "suppress" and "retain" will act accordingly with the given character
- object to which this property applies: <character>

switch-to= (7.22.11; XSL; extended):

- Value: xsl-preceding | xsl-following | xsl-any | <name>[<name>]*
- Initial: xsl-any
- Inherited: no
- Media: interactive

- specifies which <multi-case> object this <multi-toggle> will switch to when evoked
- object to which this property applies: <multi-toggle>

table-layout= (7.26.16; CSS; extended):

- Value: auto | fixed | inherit
- Initial: auto
- Inherited: no
- Media: visual

- object to which this property applies: <table>

table-omit-footer-at-break= (7.26.17; XSL; extended):

- Value: true | false
- Initial: false
- Inherited: no
- Media: visual

- specifies if a table whose last area is not at the end of an area produced by the table should end with the content of the `<table-footer>` formatting object or not
- object to which this property applies: `<table>`

`table-omit-header-at-break=` (7.26.18; XSL; extended):

- Value: true | false
- Initial: false
- Inherited: no
- Media: visual

- specifies if a table whose first area is not at the beginning of an area produced by the table should start with the content of the `<table-header>` formatting object or not
- object to which this property applies: `<table>`

`target-presentation-context=` (7.22.12; XSL; extended):

- Value: use-target-processing-context | <uri-specification>
- Initial: use-target-processing-context
- Inherited: no
- Media: interactive

- specifies the limited context in which the resource should be presented if the external destination is a resource of a processed structured media type for which a limited presentational context makes sense
- object to which this property applies: `<basic-link>`

`target-processing-context=` (7.22.13; XSL; extended):

- Value: document-root | <uri-specification>
- Initial: document-root
- Inherited: no
- Media: interactive

- specifies the root of a virtual document that the processor preparing the new presentation should process if the external destination is a resource of a processed structured media type
- object to which this property applies: `<basic-link>`

`target-stylesheet=` (7.22.14; XSL; extended):

- Value: use-normal-stylesheet | <uri-specification>
- Initial: use-normal-stylesheet
- Inherited: no
- Media: interactive

- specifies the stylesheet to be used for processing the target resource
- object to which this property applies: `<basic-link>`

`text-align=` (7.15.9; CSS; basic):

- Value: start | center | end | justify | inside | outside | left | right | <string> | inherit
- Initial: start
- Inherited: yes
- Media: visual

- <string> applies only to `<table-cell>` and specifies that which cells in a table column will align
- `"left"` and `"right"` are interpreted respectively as `"start"` and `"end"`
- objects to which this property applies: `<block>`, `<external-graphic>`, `<instream-foreign-object>`, `<table-and-caption>`

`text-align-last=` (7.15.10; XSL; extended):

- Inherited: yes
- Media: visual

- Value: relative | start | center | end | justify | inside | outside | left | right | inherit
- Initial: relative
- "`left`" and "`right`" are interpreted respectively as "`start`" and "`end`"
- object to which this property applies: `<block>`

`text-altitude`= (7.27.4; XSL; extended):

- Value: use-font-metrics | <length> | <percentage> | inherit
- Initial: use-font-metrics
- Inherited: no
- Percentages: refer to font's em-height
- Media: visual
- specifies the "height" to be used for the ascent above the dominant baseline
- objects to which this property applies: `<block>`, `<character>`, `<leader>`, `<page-number>`, `<page-number-citation>`

`text-decoration`= (7.16.4; CSS; extended):

- Value: none | [[underline | no-underline] || [overline | no-overline] || [line-through | no-line-through] || [blink | no-blink]] | inherit
- Initial: none
- Inherited: no, but see prose
- Media: visual
- this is not inherited but descendant boxes are formatted with the same decoration
- the color of the decorations remains the same even if descendent elements use different colors
- objects to which this property applies: `<character>`, `<initial-property-set>`, `<inline>`, `<page-number>`, `<page-number-citation>`

`text-depth`= (7.27.5; XSL; extended):

- Value: use-font-metrics | <length> | <percentage> | inherit
- Initial: use-font-metrics
- Inherited: no
- Percentages: refer to font's em-height
- Media: visual
- specifies the "depth" to be used for the descent below the dominant baseline
- objects to which this property applies: `<block>`, `<character>`, `<leader>`, `<page-number>`, `<page-number-citation>`

`text-indent`= (7.15.11; CSS; basic):

- Value: <length> | <percentage> | inherit
- Initial: 0pt
- Inherited: yes
- Percentages: refer to width of containing block
- Media: visual
- positive values indent the start edge, negative values outdent the start edge with a hanging indent
- object to which this property applies: `<block>`

`text-shadow`= (7.16.5; CSS; extended):

- Inherited: no, see prose
- Media: visual

- Value: none | [<color> || <length> <length> <length>? ,]* [<color> || <length> <length> <length>?] | inherit
- Initial: none

- specifies a comma-separated list of shadow effects to be applied to the text of the element
- a shadow effect is the color used as a basis the effect, the horizontal distance to the right (positive) or left (negative) of the text, the vertical distance below (positive) or above (negative) the text, and optionally a blur radius
- objects to which this property applies: `<character>`, `<initial-property-set>`, `<leader>`, `<page-number>`, `<page-number-citation>`

`text-transform`= (7.16.6; CSS; extended):

- Value: capitalize | uppercase | lowercase | none | inherit
- Initial: none
- Inherited: yes
- Media: visual

- this is deprecated in XSL-FO due to "severe internationalization issues"
- objects to which this property applies: `<character>`, `<initial-property-set>`, `<page-number>`, `<page-number-citation>`

`top`= (7.5.2; CSS; extended):

- Value: <length> | <percentage> | auto | inherit
- Initial: auto
- Inherited: no
- Percentages: refer to height of containing block
- Media: visual

- see `bottom`= for details
- object to which this property applies: `<block-container>`

`treat-as-word-space`= (7.16.7; XSL; extended):

- Value: auto | true | false | inherit
- Initial: auto
- Inherited: no
- Media: visual

- specifies if the character shall be treated as a word space ("`true`") or as a normal letter ("`false`")
- object to which this property applies: `<character>`

`unicode-bidi`= (7.27.6; CSS; extended):

- Value: normal | embed | bidi-override | inherit
- Initial: normal
- Inherited: no
- Media: visual

- specifies the opening of an additional level of embedding of characters for the bidirectional algorithm
- using "`embed`" provides for nesting directionality for blocks of Unicode characters separated by characters with weak or neutral directionality, without impacting on those characters that have a defined directionality
- using "`bidi-override`" ignores any defined directionality and imposes direction on the enclosed characters
- object to which this property applies: `<bidi-override>`

`vertical-align`= (7.29.22; CSS; shorthand):

- Value: baseline | middle | sub | super | text-top | text-bottom | <percentage> | <length> | top | bottom | inherit
- Initial: baseline
- Inherited: no
- Percentages: refer to the 'line-height' of the element itself
- Media: visual

- values specify the vertical positioning of inline-level constructs
- Value

	`alignment-baseline=`	`alignment-adjust=`	`baseline-shift=`	`dominant-baseline=`
`"baseline"`	`"baseline"`	`"auto"`	`"baseline"`	`"auto"`
`"top"`	`"before-edge"`	`"auto"`	`"baseline"`	`"auto"`
`"text-top"`	`"text-before-edge"`	`"auto"`	`"baseline"`	`"auto"`
`"middle"`	`"middle"`	`"auto"`	`"baseline"`	`"auto"`
`"bottom"`	`"after-edge"`	`"auto"`	`"baseline"`	`"auto"`
`"text-bottom"`	`"text-after-edge"`	`"auto"`	`"baseline"`	`"auto"`
`"sub"`	`"baseline"`	`"auto"`	`"sub"`	`"auto"`
`"super"`	`"baseline"`	`"auto"`	`"super"`	`"auto"`
<percentage>	`"baseline"`	<percentage>	`"baseline"`	`"auto"`
<length>	`"baseline"`	<length>	`"baseline"`	`"auto"`

`visibility=` (7.28.8; CSS; extended):

- Value: visible | hidden | collapse | inherit
- Initial: visible
- Inherited: yes
- Media: visual

- specifies whether the boxes generated by an element are rendered even while affecting layout
- objects to which this property applies: `<block>`, `<character>`, `<inline>`, `<leader>`, `<page-number>`, `<page-number-citation>`, `<table-body>`, `<table-column>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`

`voice-family=` (7.6.17; CSS; basic):

- Value: [[*specific-voice* | *generic-voice*],]* [*specific-voice* | *generic-voice*] | inherit
- Initial: depends on user agent
- Inherited: yes
- Media: aural

- *generic-voice* values are voice families from the set "`male`", "`female`" and "`child`"
- *specific-voice* values are specific instances that may be recognized by the rendering agent
  - e.g. "`comedian`", "`romeo`", "`juliet`", etc.
- objects to which this property applies: `<basic-link>`, `<bidi-override>`, `<block>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`

`volume=` (7.6.18; CSS; basic):

- Value: <number> | <percentage> | silent | x-soft | soft | medium | loud | x-loud | inherit
- Initial: medium
- Inherited: yes
- Percentages: refer to inherited value
- Media: aural

- specify the median volume of the waveform
- objects to which this property applies: `<basic-link>`, `<bidi-override>`, `<block>`, `<character>`, `<external-graphic>`, `<initial-property-set>`, `<inline>`, `<instream-foreign-object>`, `<leader>`, `<list-block>`, `<list-item>`, `<page-number>`, `<page-number-citation>`, `<table-and-caption>`, `<table>`, `<table-body>`, `<table-caption>`, `<table-cell>`, `<table-footer>`, `<table-header>`, `<table-row>`, `<title>`

`white-space=` (7.29.23; CSS; shorthand):

- Value: normal | pre | nowrap | inherit
- Initial: normal
- Inherited: yes
- Media: visual

- specifies a set of properties that would otherwise be individually specified
- Value `linefeed-treatment=white-space-collapse=white-space-treatment=wrap-option=`
  `"normal" "treat-as-space" "true" "ignore-if-surrounding-linefeed" "wrap"`
  `"pre" "preserve" "false" "preserve" "no-wrap"`
  `"nowrap" "treat-as-space" "true" "ignore-if-surrounding-linefeed" "no-wrap"`

`white-space-collapse=` (7.15.12; XSL; extended):

- Value: false | true | inherit
- Initial: true
- Inherited: yes
- Media: visual

- specifies that a space, tab or carriage return character is ignored if it follows another such character or immediately precedes a linefeed character
- object to which this property applies: `<block>`

`white-space-treatment=` (7.15.8; XSL; extended):

- Value: ignore | preserve | ignore-if-before-linefeed | ignore-if-after-linefeed | ignore-if-surrounding-linefeed | inherit
- Initial: ignore-if-surrounding-linefeed
- Inherited: yes
- Media: visual

- specifies the behavior of space, tab and carriage return characters
- object to which this property applies: `<block>`

`widows=` (7.19.7; CSS; basic):

- Value: <integer> | inherit
- Initial: 2
- Inherited: yes
- Media: visual

- specifies the minimum number of lines of a block that must be left at the top of a page
- object to which this property applies: `<block>`

`width=` (7.14.12; CSS; basic):

- Value: <length> | <percentage> | auto | inherit
- Initial: auto
- Inherited: no
- Percentages: refer to width of containing block
- Media: visual

- negative values are illegal
- does not apply to non-replaced inline-level elements
- objects to which this property applies: `<block-container>`, `<external-graphic>`, `<inline>`, `<inline-container>`, `<instream-foreign-object>`, `<table>`, `<table-caption>`, `<table-cell>`

`word-spacing=` (7.16.8; CSS; extended):

- Value: normal | <length> | <space> | inherit
- Inherited: yes
- Initial: normal
- Media: visual

- specifies inter-word spacing behavior in addition to the default between words
- negative values are allowed
- objects to which this property applies: `<bidi-override>`, `<character>`, `<initial-property-set>`, `<leader>`, `<page-number>`, `<page-number-citation>`

`wrap-option=` (7.15.13; XSL; basic):

- Value: no-wrap | wrap | inherit
- Inherited: yes
- Initial: wrap
- Media: visual

- specifies how line-breaking of the content is to be handled
- "`no-wrap`" will cause lines longer than the width of the content-rectangle to be considered an overflow condition on the reference-area
- objects to which this property applies: `<block>`, `<inline>`, `<page-number>`, `<page-number-citation>`

`writing-mode=` (7.27.7; XSL; basic):

- Value: lr-tb | rl-tb | tb-rl | lr | rl | tb | inherit
- Inherited: yes (see prose)
- Initial: lr-tb
- Media: visual

- "`lr`" is a shorthand for "`lr-tb`"
- "`rl`" is a shorthand for "`rl-tb`"
- "`tb`" is a shorthand for "`tb-rl`"
- objects to which this property applies: `<block-container>`, `<inline-container>`, `<region-after>`, `<region-before>`, `<region-body>`, `<region-end>`, `<region-start>`, `<simple-page-master>`, `<table>`

`xml:lang=` (7.29.24; XSL; shorthand):

- Value: *country-language* | inherit
- Inherited: yes
- Initial: not defined for shorthand properties
- Media: visual

- *country-language* is a language and/or country value in conformance with RFC-3066
- recognized as a shorthand for `country=` and `language=`

`z-index=` (7.28.9; CSS; extended):

- Value: auto | <integer> | inherit
- Inherited: no
- Initial: auto
- Media: visual

- constructs with higher index numbers are "in front" of those with lower index numbers and show through the constructs that are "behind"
- object to which this property applies: `<block-container>`

Annex E - Sample tool information

- Introduction - Sample questions for vendors

Sample questions for vendors

Annex E - Sample tool information

Answers to the following questions may prove useful when trying to better understand an XSL-FO product offering from a vendor. The specific questions are grouped under topical questions. This by no means makes up a complete list of questions as you may have your own criteria to add, nonetheless, they do cover aspects of XSLT that may impact on the stylesheets and transformation specifications you write.

- how is the product identified?
    - what is the name of the XSL-FO processor in product literature?
    - which version of XSL-FO is supported?
    - to which email address or URL are questions forwarded for more information in general?
    - to which email address or URL are questions forwarded for more information specific to the answers to these technical questions?
- what are the details of the implementation and invocation of the XSL-FO processor?
    - which hardware/operating system platforms support the processor?
    - which character sets are supported for the input file encoding?
    - what is the XSLT processor used?
        - can it be replaced with another XSLT processor?
    - what is the XML processor used?
        - can it be replaced with another XML processor?
        - does the XML processor support minimally declared internal declaration subsets with only attribute list declarations of `ID`-typed attributes?
        - does the XML processor support XML Inclusions (Xinclude)?
        - does the XML processor support catalogues for public identifiers?
        - does the XML processor validate the source file?
            - can this be turned on and off?
    - can the processor be embedded in other applications?
        - can the processor be configured as a servlet in a web server?
    - is the source code of the processor available?
        - in what language is the processor written?
    - for Windows-based environments:
        - can the processor be invoked from the MSDOS command-line box?
        - can the processor be invoked from a GUI interface?
        - what other methods of invocation can be triggered (DLL, RPC, etc.)?
        - can error messages be explicitly redirected to a file using an invocation parameter (since, for example, Windows-95 does not allow for redirection of the standard error port to a file)?

Sample questions for vendors (cont.)

- which features of XSL-FO are supported?
  - does the processor respect the inherent writing directions of Unicode characters and the Unicode bidirectional text algorithm?
  - which constructs are absent or only partially supported?
  - what graphics formats does the product support?
    - does the formatter support relative URI values for external graphic resources?
    - does the formatter require a protocol specification in the URI?
    - are vector-based formats rendered at device resolution?
    - example of photographic formats
      - JPEG (supports "lossy" compression)
    - examples of pixel formats
      - BMP (Windows)
      - PNG (see `http://www.libpng.org/` and `http://burnallgifs.org` for information)
      - GIF (requires LZW licensing from `http://www.unisys.com`)
    - examples of vector formats
      - SVG (Scalable Vector Graphics)
      - WMF (Windows Meta File)
      - EPS (Encapsulated PostScript)
      - CGM (Computer Graphics Metafile)
  - which color schemes are supported?
- which features of EXSLFO are supported?
  - see `http://www.exslfo.org` for details on community-defined extensions to the XSL-FO vocabulary that may or may not be supported by vendors
- how are particular features implemented?
  - at what conformance level is each feature implemented?
  - what is the thickness of each of the border width named values?
  - which writing directions are supported?
  - which languages are supported for hyphenation?
  - which fonts are supported and how are fonts supported?
  - which ligatures are recognized for adjacent characters?
  - which properties and states of `<basic-link>` are supported and how?
    - does the user interface have a "back" function to retrace steps?

Sample questions for vendors (cont.)

- what convenience features are implemented?
    - can the resulting file be fragmented for ease of management or transmission?
    - can the result be packaged in groups of different page sizes (to support fold-outs)?
    - can different hoppers be specified for the printer for different paper selections?
- what extension formatting objects and properties are implemented?
    - are there any vendor-defined extensions to the vocabulary?
    - what namespace URI is used to identify the vocabulary?
- what output formats are supported?
    - PDF?
        - are internal links supported?
        - are external links supported?
        - is there an extension for turning on the PDF security bit in the result?
        - is there an extension for defining the "General Info" fields for title, subject, author, etc.
    - PostScript?
    - Windows GDI?
    - PCL?
    - TeX?
    - RTF?
    - TIFF?
    - other?
- what instream foreign object vocabulary namespaces are supported?
    - SVG?
    - MathML?
    - other?

Where To Go From Here?

Conclusion - Practical Formatting Using XSL-FO

The work on XSL and XSLT continues:

- all XSLT, XPath and XSL-FO are now full W3C Recommendations
- long list of future feature considerations already being examined for new releases of the technology
- new products are continually being announced
- feedback is necessary from users like you!
    - use the XSL mail lists to contribute:
        - `http://www.mulberrytech.com/xsl/xsl-list/`
        - `http://groups.yahoo.com/group/XSL-FO`
        - `http://lists.w3.org/Archives/Public/www-xsl-fo/`
    - contact the XSL editors with comments about the specification:
        - `xsl-editors@w3.org`

Colophon

Conclusion - Practical Formatting Using XSL-FO

These materials were produced using structured information technologies as follows:

- authored source materials
  - content in numerous XML files maintained as external general entities for a complete prose book that can be made into a subset for training
    - specification of applicability of constructs for each configuration
      - 45- and 90-minute lecture, half-, full-, two- and three-day lecture and hands-on instruction, and book (prose) configurations
    - an XSLT transformation creates the subset of effective constructs from applying applicability to the complete file
    - content from other presentations/tutorials included semantically (not syntactically) during construct assembly
  - customized appearance engaged with marked sections and both parameter and general entities
    - different host company logos and venue and date marginalia
    - changing a single external parameter entity to a key file includes suite of files for given appearance
- accessible rendition in HTML
  - an XSLT stylesheet produces a collection of HTML files using Saxon for multiple file output
  - mono-spaced fonts and list-depth notation conventions assist the comprehension of the material when using screen-reader software
- printed handout deliverables
  - an XSLT stylesheet produces an instance of XSL formatting objects (XSL-FO) for rendering
  - XPDF `http://www.foolabs.com/xpdf` extracts raw text from PDF files for the back-of-the-book index methodology published as a free resource by Crane Softwrights Ltd.
  - XEP by RenderX `http://www.renderx.com` produces PostScript from XSL-FO
  - GhostScript `http://www.GhostScript.com` produces PDF from PostScript
  - the iText `http://itext.sf.net` PDF manipulation library for Java is used for page imposition by a custom Python `http://www.python.org` program running under the Jython `http://www.jython.org` environment

Obtaining a copy of the comprehensive tutorial

Conclusion - Practical Formatting Using XSL-FO

This comprehensive tutorial on XSL-FO is available for subscription purchase and free preview download:

- "Practical Formatting Using XSL-FO (Extensible Stylesheet Language Formatting Objects)" Fifth Edition - 2003-02-12 - ISBN 1-894049-13-6
  - the free download preview excerpt of the publication indicates the number of pages for each topic
- the cost of purchase includes all future updates to the materials with email notification
  - the materials are updated after new releases of the W3C specifications
  - the materials are updated after incorporating comments gleaned during presentations and from feedback from customers
- available in PDF
  - formatted as 1-up or 2-up book pages per imaged page
  - dimensions in either US-letter or A4 page sizes
  - available as either single sided or double sided, full-page or half-page bound
- accessible rendition available for use with screen readers
- free preview download includes full text of first three chapters and two useful annexes
- site-wide and world-wide staff licenses (one-time fee) are available

See `http://www.CraneSoftwrights.com/links/trn-20040212.htm` for more details.

Feedback

- the unorthodox style has been well-accepted by customers as an efficient learning presentation
- feedback from customers is important to improve or repair the content for future editions
- please send suggestions or comments (positive or negative) to `feedback@CraneSoftwrights.com`

US Government employee purchase

- US Government employees (not contractors) are entitled to obtain their personal prepaid copies at no charge from a government intranet location
- visit the Crane web site for details

Practical Formatting Using XSL-FO (Extensible Stylesheet Language Formatting Objects)

Crane Softwrights Ltd.
http://www.CraneSoftwrights.com

Fifth Edition - 2004-02-12 - ISBN 1-894049-13-6

Practical Formatting Using XSL-FO

Table of contents
Indexed by slide number

Practical Formatting Using XSL-FO

Index

Q

R

Y

Z