

# Index

## SYMBOLS

### occurrence indicators

\*, 156  
?, 156  
+, 156

### operators

<|>, 762  
<!=>, 761  
<<+>, 128, 762  
<<\*>, 128, 762  
<,>, 761  
<//>, 763  
</>, 762  
<[]>, 763  
<<<>, 761  
<<=>, 761  
<<<>, 127, 761  
<=>, 127, 761  
<>=>, 761  
<->, 762  
<->>, 761  
<->>>, 761

## A

**abort() method (MSXML), 801**

**abs() function, 765**

**adjust-date-to-timezone() function, 766**

**adjust-date-to-timezone() function, 766**

**adjust-time-to-timezone() function, 766**

### aggregation functions

avg(), 620, 766  
max(), 224, 620, 780  
min(), 224, 620, 780  
sum(), 634, 793

**Ambroziak, Jack, 8**

**ancestor axis, 77, 763**  
**ancestor-or-self axis, 763**

<&and>, 761

**ANSEL character, 703**

**anyAtomicType, 156**

**API (Application Programming Interface), 6, 59, 85**

**Application Programming Interface (API), 6, 59, 85**

**as attribute, 63, 73, 154, 200**

**ASP page, 133, 715**

**ASP.NET, 8, 20**

### assignment statements, avoiding

avoid doing two things at once, 631  
conditional initialization, 630  
using recursion, 633

**atomization, 73**

**attribute axis, 77, 763**

**attribute name at runtime, 213**

**attribute node for an element, creating, 210**

**attribute node, 51, 204**

**attribute normalization (in XML), 137**

**attribute set for numbering, example of, 218**

**attribute set names, 217**

**attribute sets, 214**

**attribute value templates, 116, 118**

**attribute whose value is QName, creating, 210**

**attribute(), 75**

**attribute-or-top axis, 501**

**attributes of a node, 53**

**attributes of literal result element, 110**

**avg() function, 620, 766**

## B

**backwards compatibility, 123, 126, 865**

### backwards compatibility mode

comparing strings, 866  
first node rule, 865  
numeric precision, 867  
other changes, 867  
strong conversions, 74

**base URI for output documents, 59**

**base URI of a node, 53**

**base-uri() function, 766**

**basic processor. See stylesheets and schemas**

**best move, selecting (Knight's tour example), 750**

**beyond XSLT 1.0, 31**

**boolean() function, 310, 767**

**boundary whitespace, 648**

**Broadcast Markup Language, 2**

**BufferedReader object (used in example extension function), 606**

**built-in atomic types, 157**

**built-in template rules, 70, 190**

**<bullet> elements (grouping example), 291**

## calendar argument (of format-date() function)

---

### C

**calendar argument (of format-date() function), 554**  
**captured groups, 181**  
**Cascading Style Sheets (CSS), 21, 94, 519**  
**«cast as», 762**  
**«castable as», 762**  
**casting, 74**  
**CDATA sections, 57**  
**cdata-section-elements attribute (of <xsl:output>), 231, 377**  
**ceiling() function, 767**  
**character maps**  
 choosing characters to map, 233  
 limitations of, 235  
 used to comment-out elements, example of, 234  
 versus disable-output-escaping, 233  
**child axis, 77, 763**  
**child-or-top axis, 501**  
**children of a node**  
 choosing, 183  
**Clark notation (for expanded QNames), 55**  
**Clark, James, 8, 25, 28**  
**client-side script, 599, 808**  
**closure, 7, 44**  
**codepoints-to-string() function, 767**  
**collation attribute, 284, 333**  
**collations, 427, 858**  
**collection() function, 768**  
**COM object, 735**  
**command-line interface, 59**  
**command-line parameters, 85**  
**comma-separated values, 7, 589**  
**comment node, 51**  
**comment(), 75**  
**compare() function, 768**  
**compile time, 122**  
**complex type with complex content, categories of**  
 element-only content, 149  
 empty content, 149  
 mixed content, 149  
**complex type with simple content, 149**  
**computational stylesheets**  
 assignment statements, 630  
 cheating, 629  
 grouping, 641  
 programming without assignment statements, 625  
 variables, 629  
**concat() function, 117, 768**  
**conditional compilation, 122**  
**conditional expression (XPath 2.0), 35**  
**conflict resolution, 65, 71, 498**  
**constructor functions, 74**  
**contains() function, 769**  
**content of a constructed document, 258**  
**content of a constructed element, 263**  
**ContentHandler interface (SAX2), 46**

### context

dynamic context, 78  
 static context, 78  
**context item, 79**  
**context item, changing, 223, 280**  
**context position, 79, 785**  
**context size, 79, 785**  
**controlling sequence of processing, 68**  
**conversion, of types, 73**  
**converting attributes to child elements (example), 270**  
**copying nodes to and from temporary trees, 247**  
**copy-namespaces attribute, 241, 245, 246**  
**core function library, 130**  
**count() function, 141, 306, 368, 634, 769**  
**country argument (of format-date() function), 557**  
**createProcessor() method (MSXML), 737**  
**creating attributes, different ways of, 208**  
**creating multiple output files, examples of, 418, 545**  
**cross-references, making, 672**  
**CSS (Cascading Style Sheets), 94, 519**  
**current mode, 190**  
**current template rule, 185**  
**current() function, 526**  
**current-date() function, 85, 769**  
**current-dateTime() function, 769**  
**current-group() function, 283, 286, 529**  
**current-grouping-key() function, 283, 286, 523, 530**  
**current-time() function, 769**  
**customization layer, 324**  
**cycles in a graph, examples of checking for, 199, 307**

### D

**data conversion applications, 17**  
**data file, examples of creating**  
 converting from GEDCOM 5.5 to 6.0, 704  
 GEDCOM files to XML, converting, 703  
**data in tables, arranging, 298**  
**data types**  
 data types, 74  
 expressions, 76  
 global, 73  
 immutability, 73  
 local, 73  
 parameters, 74  
 temporary trees, 79  
**data() function, 727, 770**  
**data-oriented XML applications, 691**  
**date, 155**  
**Date:new() (example extension function), 113**  
**Date:toString() (example extension function), 113**  
**day-from-date() function, 770**  
**day-from-dateTime() function, 770**

## event records, creating (Family tree example)

**days-from-dayTimeDuration() function, 770**

**declaration order, 316**

**declarations**

implementor-defined declarations, 100

user-defined top-level elements, 101

XSLT-defined declarations, 99

**declarations, XSLT-defined**

<xsl:attribute-set>, 99, 214

<xsl:character-map>, 99, 229

<xsl:decimal-format>, 99, 251

<xsl:function>, 99, 300

<xsl:import>, 99, 312

<xsl:import-schema>, 99, 324

<xsl:include>, 100, 328

<xsl:key>, 100, 332

<xsl:namespace-alias>, 100, 116, 350

<xsl:output>, 100, 375

<xsl:param>, 100, 392

<xsl:preserve-space>, 100, 406

<xsl:strip-space>, 100, 432

<xsl:template>, 100, 450

<xsl:variable>, 100, 471

**declaring types, 154**

**deep copy, 249**

**deep-equal() function, 770**

**default namespace, 54, 98**

**default value of stylesheet and template**

parameters, 395

**default-collation() function, 771**

**default priority, 498**

**default-validation attribute (of xsl:stylesheet), 92, 168**

**descendant axis, 763**

**descendant-or-self axis, 763**

**design principles (of XSLT language), 29**

**disable-output-escaping attribute, 233, 460, 466**

**disable-output-escaping, using, 470**

**displaying family tree data (Family tree example)**

putting it together, 728

stylesheet, 715

**distinct-values() function, 224, 620, 642, 771**

**distributed object systems**

COM, 47

Java, 47

**<<div>>, 762**

**divide-and-conquer recursion, 634**

**Döbler, Johannes, 8**

**doc() function, 79, 771**

**document element, 88**

**document header, formatting, 655**

**document node, 49, 51**

**Document Object Model (DOM), 6, 48, 375, 494**

**document order, 76, 278**

**document order. See sorting**

**Document Style Semantics and Specification**

Language (DSSSL), 27, 627

**Document Type Definitions (DTD), 23**

**document() function to analyze stylesheet, example of, 537**

**document() function, 39, 47, 66, 532**

**DocumentBuilderFactory.newInstance() method (JAXP), 819**

**document-node() function, 75, 163**

**document-uri() function, 772**

**DOM (Document Object Model), 6, 48, 375, 494**

**DSSSL (Document Style Semantics and Specification Language), 27, 627**

**DTD (Document Type Definitions), 23**

**DTD information in the tree model, 58**

**DTD types, 58**

**duplicate attribute names, 205, 217**

**duplicate namespace declarations, 114, 350**

**dyn:evaluate() function (EXSLT), 859**

**dynamic sort keys, 430**

## E

**EDI messages, 15, 18, 37, 46**

**effective boolean value, 236, 309, 767**

**electronic data interchange message, 15, 18, 37, 46**

**element node, 51**

**element nodes within sequence constructor**

extension elements, 103

literal result elements, 103

XSLT instructions, 103

**element or attribute, 115**

**element types (in XML Schema), 148**

**element(), 75**

**element-available() function, 122, 126, 445, 542**

**element-only content, 148**

mixed content, 149

simple content, 148

**elements that may appear as children of**

<xsl:stylesheet>

implementor-defined declarations, 98

user-defined data elements, 98

XSLT-defined declarations, 98

**embedded stylesheets, 95**

**empty() function, 772**

**encoding argument (to unparsed-text() function), 588**

**encoding attribute (of xsl:output), 389**

**ends-with() function, 772**

**<<eq>> operator, 284, 761**

**error handling**

dynamic errors, 71

static errors, 71

**error() function, 773**

**escape-uri() function, 773**

**evaluate() extension function (EXSLT), 859**

**event based interface, 6**

**event records, creating (Family tree example), 709**

---

**«every»**


---

**«every», 761**  
**exactly-one() function, 773**  
**example of aggregating a list of numbers, 635**  
**example of navigational stylesheet, 617**  
**example of rule-based stylesheet, 621**  
**example of supplying parameters and output properties, 843**  
**example of transformation using files, 842**  
**example of using <xsl copy-of> for repeated output source, 247**  
**example of using a character-map to comment-out elements, 234**  
**example of using an attribute set for numbering, 218**  
**example of using an extension instruction, 134**  
**example of using client-side JScript to transform a document, 808**  
**example of using document() function to analyze stylesheet, 537**  
**example of using generate-id() to create links, 570**  
**example of using interleaved structures, 637**  
**example of using keys as cross-references, 575**  
**example of using modes, 194**  
**example of using recursion to process a sequence of nodes, 224**  
**example of using recursion to process a sequence of strings, 227**  
**examples section. See literal result elements**  
**«except» operator, 209, 762**  
**«except», 762**  
**exclude-result-prefixes attribute, 93, 439**  
**existence of system-defined functions, testing for, 565**  
**exists() function, 773**  
**expanded content, 231**  
**expanded-QName() function, 774**  
**explicit base URI, supplying, 542**  
**expressions (XPath), 76**  
**extensibility, 128, 595**  
**eXtensible Stylesheet Language (XSL), 21**  
**eXtensible Stylesheet Language Transformations (XSLT), 1**  
**extension attributes, 596**  
**extension declarations, 596**  
**extension elements, 118**  
**extension function examples**  
   calculate a square root, 602  
   freeMemory() function, 132  
   getRuntime() function, 132  
**extension functions**  
   binding, 599  
   calling extension functions, 598  
   calling external functions within a loop, 606  
   calling, 598  
   client-side script, 599  
   explicit binding, 599  
**extension instructions, 105, 133, 135**

**extension types, 596**  
**extension-element-prefixes attribute, 93, 133, 438**  
**extensions, keeping portable**  
   function-available() function, 612  
   system-property() function, 612  
**external functions within a loop, calling, 606**  
**external general parsed entity, 50**

**F**

**false() function, 774**  
**family records, creating, 706**  
**family tree**  
   case study of, 691–92  
   modeling, 692  
**family tree data, displaying, 714**  
**<FamilyRec> element (Family tree example), 706**  
**features available in later XSLT versions, testing for, 544**  
**fill-in-the-blanks stylesheets, 613**  
**final board, displaying (Knight's tour example), 745**  
**find-best-move() function (Knight's tour example), 749**  
**finding the route (Knight's tour example), 746**  
**flags (for regular expressions)**  
   i, 178  
   m, 178  
   s, 178  
   x, 178  
**floor() function, 774**  
**focus, 78**  
**fold() function (FXSL example), 198**  
**following axis, 763**  
**following-sibling axis, 77, 763**  
**for loop, 223**  
**<for>, 761**  
**formal definition of pattern semantics, 495**  
**Format section. See literal result elements**  
**format string (of xsl:number), analyzing, 364**  
**format-date() function**  
   presentation modifiers, 553  
**format-dateTime() function, 550**  
**format-number() function, 149, 251, 359, 362, 558**  
**format-time() function, 562**  
**formatting a list of names, example of, 311**  
**forwards compatibility in XSLT 1.0, 124**  
**forwards-compatibility mode, 105, 123, 566**  
**4xslt, 8**  
**from attribute (of xsl:number), 369**  
**function parameters, 74**  
**function. See programming without assignment statements**  
**functional programming, 306, 626**  
**function-available() function, 122, 125, 445, 564**

## implementor defined declarations

**FunctionCall construct (XPath), 518**  
**functions versus named templates, 304**  
**functions with uncontrolled side effects, 609**  
**FXSL library of extension functions, 198**

## G

«ge», 761  
**GEDCOM 5.5 to 6.0, converting from, 704**  
**GEDCOM 6.0 schema, 696**  
**GEDCOM data model, 692**  
**GEDCOM files to XML, converting, 703**  
**GEDCOM model, object types of**  
   contact, 695  
   group, 694  
   repository, 695  
   source, 695  
**GEDCOM parser, 703**  
**<GEDCOM> element, 706**  
**generate-id() function, 246, 291, 568**  
**generate-id() to create links, example of, 570**  
**generating an attribute conditionally, example of, 211**  
**getXMLReader() method (JAXP), 818**  
**global id parameter, 737**  
**global variables, 73, 99, 622**  
**goto statement, 625**  
**Gregor, 8**  
**group theory, 44**  
**group-adjacent attribute (of xsl:for-each-group)**  
   using, 290  
**group-by attribute (of xsl:for-each-group)**  
   using, 286  
**group-ending-with attribute (of xsl:for-each-group), 285**  
**grouping consecutive elements by name, example of, 291**  
**grouping, 641**  
**grouping-separator attribute (of xsl:number), 366**  
**grouping-size attribute (of xsl:number), 366**  
**group-starting-with attribute (of xsl:for-each-group), 284**  
 «gt», 761  
**GUI interface, 59**

## H

**handling flat XHTML documents, example of, 296**  
**handling repeating groups of adjacent elements, example of, 294**  
**head-tail recursion, 224**  
**Hello World example, 10**  
**higher order function, 200**  
**hours-from-dateTime() function, 775**  
**hours-from-dayTimeDuration() function, 775**  
**hours-from-time() function, 775**

**href attribute**  
 ?xml:stylesheet?<>, 95  
 <xsl:import>, 313  
 <xsl:include>, 329

## HTML

boilerplate generation, 592  
 files, 2, 728  
 in the browser, generating, 735  
 internal hyperlink, 682  
 outline, creating, 651  
 output method, 38, 46, 385  
 output page, 132  
 pages, 298, 715  
 tags, 621  
**HTML outline, creating, 651**  
**HTML pages from servlet, generating, 730**  
**HTML using ASP pages, generating, 735**  
**<html> element, 46, 61**  
**HTTP**  
   message, 2  
   server, 733  
**hyperlinks, in HTML output, 47**

## I

**IANA (Internet Assigned Numbers Authority), 427**  
**id attribute, 214, 437**  
**id() function, 58, 335, 775**  
**id/idref constraint, 251**  
**identity constraints (XML Schema)**  
   <xs:key>, 259  
   <xs:keyref>, 259  
   <xs:unique>, 259  
**identity template, 243**  
 «idiv», 762  
**IdKeyPattern construct, 518**  
**idref attribute, 199**  
**idref() function, 776**  
**if expression (XPath), 35, 227, 236, 310**  
 «if», 761  
**implementor defined declaration, features of**  
   binding of extension functions, 100  
   collations used for sorting, 100  
   details of result-tree serialization, 100  
   extension instructions, 100  
   localization of messages, 100  
**implementor defined declarations**  
   implicit binding, 131, 601  
   Java examples, 131  
   JavaScript example, 132  
   need for, 598  
   performance, 83, 129  
   purpose of, 129  
   side effects, 609  
   significance of, 597  
   writing, 857

## implicit-timezone() function

---

**implicit-timezone() function, 776**

**import precedence**

- determining, 314
- effect of, 316
- of template rules, example of, 321
- of variables, example of, 320

**import precedence. See xsl:import**

**importing schemas, 168, 325**

**index-of() function, 776**

**individual attributes, validating, 167**

**individual elements, validating, 164**

**individual records, creating (Family tree example), 707**

**informal definition of pattern semantics, 497**

**InfoSet, 24**

**init() method (Java servlets), 731, 737**

**initial mode, 59**

**initial sequence, 425**

**initial template, 59, 742**

**inputs and outputs, multiple, 47**

**in-scope-prefixes() function, 777**

**insert-before() function, 777**

**installing and configuring the servlet, 733**

**instance methods, 132**

**<<instance of>>, 159, 762**

**instructions**

- attribute value templates, 116
- extension instructions, 105
- literal result elements, 106
- XSLT instructions, 103

**integer, 155**

**interleaved structures, example of, 637**

**International Resource Identifier (IRI), 23**

**Internet Assigned Numbers Authority (IANA), 427**

**Internet Service Provider, 715**

**intersect operator, 200, 308, 717**

**<<intersect>>, 762**

**introspection, 58**

**invoking a transformation, 59**

**IRI (International Resource Identifier), 23**

**<<is>>, 761**

**item() function, 525**

**item(), 75**

**<item> elements, 214**

**ItemType construct, 75**

**IXMLDOMDocument and IXMLDOMDocument2 (MSXML), 801**

**IXMLDOMNode (MSXML), methods of**

- Document, 803
- selectNodes, 803
- selectSingleNode, 803
- transformNode, 803
- transformNodeToObject, 803

**IXMLDOMNode (MSXML), properties of**

- baseName, 803
- namespaceURI, 803
- nodeName, 803

nodeTypeString, 803

nodeValue, 803

prefix, 804

text, 804

xml, 804

**IXMLDOMNodeList (MSXML), methods of**

- item, 804
- nextNode, 804
- reset, 804

**IXMLDOMParseError (MSXML), properties of**

- errorCode, 804
- filepos, 804
- line, 804
- linepos, 804
- reason, 805
- srcText, 805
- url, 805

**IXPathNavigable (.NET), 813**

**IXSLProcessor object (MSXML), methods of**

- addParameter, 806
- reset, 806

**IXSLProcessor object (MSXML), properties of**

- setStartMode, 806
- transform, 806

## J

**Java**

APIs, 93

JAXP interface, 59, 133, 233, 815–850

Server Pages, 20

servlets, 20

XML processors, 815

**javax.xml.parsers.DocumentBuilder, 821**

**javax.xml.parsers.DocumentBuilderFactory method, 819**

**javax.xml.parsers.SAXParser, 818**

**javax.xml.transform.dom.DOMLocator, 824**

**javax.xml.transform.dom.DOMResult, 825**

**javax.xml.transform.dom.DOMSource, 825**

**javax.xml.transform.ErrorListener, 826**

**javax.xml.transform.OutputKeys, 827**

**javax.xml.transform.Result, 828**

**javax.xml.transform.sax.SAXResult, 828**

**javax.xml.transform.sax.SAXSource, 829**

**javax.xml.transform.sax.SAXTransformerFactory, 830**

**javax.xml.transform.sax.TemplatesHandler, 835**

**javax.xml.transform.Source, 831**

**javax.xml.transform.SourceLocator, 832**

**javax.xml.transform.stream.StreamResult, 833**

**javax.xml.transform.stream.StreamSource, 832**

**javax.xml.transform.Templates, 834**

**javax.xml.transform.Transformer, 836**

**javax.xml.transform.TransformerFactory, 839**

**javax.xml.transform.TransformerFactory ConfigurationError, 836**

**JAXP**

- benefits of, 815
- parser API, 816
- support for DOM, 819
- support for SAX, 817
- transform() method, 732, 838
- transformation API, 822

**jd.xslt processor**, 8, 37, 601

**jEdit editor**, 852

**K**

keeping extensions portable, 611

**key() function**, 280, 332, 495, 572

**key() pattern to format specific node, example of**, 520

**key/keyref constraint (XML Schema)**, 251

**keys as cross-references, example of**, 575

**keys for grouping, using**, 340, 577, 578

**keys to find nodes by value, using**, 574

**keys versus IDs**, 335

**L**

**lang attribute (of xsl:number)**, 366

**lang() function**, 777

**language argument (of format-date() function)**, 553

**last() function**, 79, 189, 288, 778

**lax validation**, 162, 207, 250, 259, 268

**lazy evaluation**, 628

**LDAP directory**, 44

**<le>**, 761

**letter-value attribute (of xsl:number)**, 366

**level attribute (of xsl:number)**, 727

**lexical QName**, 174

**libxslt**, 8

**lists, producing**, 670

**literal result elements**

- attributes, 110
- content, 108
- format, 107
- namespaces, 112
- position, 107

**load() method (MSXML)**, 801

**loadXML() method (MSXML)**, 801

**local variables**, 73

**localized messages**, 346

**local-name() function**, 55, 778

**local-name-from-QName() function**, 779

**locating a stylesheet module**, 313

**looking for cycles among attribute sets, example of**, 307

**lookup table in stylesheet, examples of**, 80, 540

**lower-case() function**, 16, 779

**<lt>**, 761

**M**

**main() method**, 844

**make-best-move() function (Knight's tour example)**, 749

**match attribute (of xsl:template)**, 60, 333, 493

**matches() function**, 182, 184, 779

**matching parentless nodes**, 500

**max() function**, 224, 620, 780

**mechanisms to test extension instruction**, 135

**media attribute**, 95

**media type (MIME type)**, 94, 588

**MEI (Music Encoding Initiative)**, 4

**method attribute (of xsl:output)**, 129, 378, 385

**methods in Java class library, examples of calling**

- Enumeration.hasMoreElements(), 134
- Enumeration.nextElement(), 134
- Properties.getProperty(), 134
- Properties.propertyNames(), 134
- System.getProperties(), 134

**Microsoft MSXML parser**, 8, 345, 735, 800

**Microsoft WD-xsl dialect**, 8, 30, 435

**Microsoft XSLT processors**, 8, 30, 799

**MIDI files**, 4

**MIME type**, 94, 588

**min() function**, 224, 620, 780

**minutes-from-dateTime() function**, 781

**minutes-from-dayTimeDuration() function**, 780

**minutes-from-time() function**, 781

**mixed content**, 149

**<mod>**, 762

**mode attribute (of xsl:apply-templates)**, 187, 190

**modeling family tree example**

- GEDCOM 6.0 schema, 696
- GEDCOM data model, 692
- schema for GEDCOM 6.0, creating, 695

**modes**, 70, 194, 454

**modular structure of stylesheet**, 84

**month-from-date() function**, 781

**month-from-dateTime() function**, 781

**months-from-yearMonthDuration() function**, 781

**Mozilla browser**, 8

**MSXML**, 8, 345, 735, 800

**Muenchian grouping**, 341, 643

**multilevel grouping by value, example of**, 288

**multiphase transformations**, 81

**multiple definitions for the same key**, 340

**multiple named keys**, 340

**multiple template rules, invoking**

- grouping-separator, 361
- grouping-size, 361
- ordinal, 361

**multiple-match example (xsl:analyze-string)**, 183

**multivalued keys**, 337

**multivalued nonunique keys, example of**, 338

**Music Encoding Initiative (MEI)**, 4

**Music Markup Language**, 4

## MusicML

---

**MusicML**, 4  
**MusicXML**, 4  
**MusiXML**, 4

## N

### name of a node, parts of

local name, 52, 262  
 namespace URI, 52, 262

### name of a parameter, 396

### name() function, 55, 204, 782

### named templates and stylesheet functions, difference between, 304

### names and namespaces, 53

#### namespace

aliasing, 115  
 attribute, 203, 326  
 declaration, 53, 230, 114  
 default, 54  
 fixup, 204, 264  
 literal result element, 112  
 node, 51, 55, 113  
 nodes, copying, 250  
 prefix, declaring, 23, 53  
 prefixes, 115  
 relative URI, 54  
 undeclarations, 23, 114  
 URI, 23, 53, 115  
 working of, 53

### namespace axis, 763

### namespaces of a node, 53

### namespaces, 22, 53

### namespace-uri() function, 782

### namespace-uri-for-prefix() function, 782

### namespace-uri-from-QName() function, 783

### NameTest construct (XPath), 509

### navigational design patterns, 619

### navigational stylesheets, 616

### <ne>, 761

### .NET

environment, 8  
 framework, 8

### Netscape browser, 8

### newDocumentBuilder() method (JAXP), 819

### newSAXParser() method (JAXP), 818

### NewsML format, 2

### nilability, 171

### node kinds

attribute, 51, 242  
 comment, 51, 243  
 document, 51, 242  
 element, 51, 242  
 namespace, 51, 243  
 processing instruction, 51, 243  
 text, 51, 242

### node(), 75, 525

### node, properties of

attributes, 53  
 base URI, 53  
 children, 53  
 name, 52  
 namespaces, 53  
 parent, 53  
 string value, 52  
 type annotation, 52  
 typed value, 52

### NodeInfo interface (Saxon), 856

### node-name() function, 204, 783

### node-set() extension function, 81, 370, 629, 859

### nodeValue() method (DOM), 606

### non-ASCII characters, 389

### non-null namespace URI, 129

### normalization

### normalize-space() function, 139, 783

### normalize-unicode() function, 784

### not() function, 784

### Novatchev, Dimitre, 198

### number format pattern result (xsl:decimal-format), 256

### number() function, 127, 362, 784

### numbering the lines of a poem, example of, 371

### numbers, formatting, 365

### numeric promotion, 73, 395

## O

### object instances, 131

### object-oriented programming language, 184

### objects in the new model

events, 693  
 families, 693  
 individuals, 693

### occurrence indicators, 75, 156

### one-or-more() function, 785

### <OPTION> element, 211

### <or>, 761

### Oracle processor, 601

### ordinal attribute (of xsl:number), 366

### output escaping, controlling, 462

### output formats, different, 45

### outputting the number (xsl:number), 367

### override attribute (of xsl:function), 302

## P

### parallel markup, 637

### <param> element, 59

### parameterized attribute value, 116

### parameters

function parameters, 74

stylesheet parameters, 74  
 template parameters, 74  
**parent axis, 763**  
**parent of a node, 52**  
**parentless nodes, matching, 500**  
**parse() method (JAXP)**  
   parser API, 816  
   support for DOM, 819  
   support for SAX, 817  
   transform() method, 732  
   transformation API, 822  
**parsing, 6, 45, 535**  
**path expression, 76**  
**PathPattern, 503**  
**PatternAxis, 508**  
**patterns containing predicates, 497**  
**patterns, meaning of, 493**  
**patterns, overview of, 493**  
**patterns, syntax of**  
   IDKeyPattern, 518  
   PathPattern, 503  
   Pattern, 502  
   PatternStep, 507  
   RelativePathPattern, 505  
**PC based Web browser, 2**  
**PDF (Portable Document Format), 2, 46**  
**percent attribute (of xsl:decimal-format), 311**  
**picture argument (of format-date() function), 551**  
**picture string, 253, 559**  
**place of XSLT in the XML family**  
   XSL and CSS, 24  
   XSLT and XML schemas, 25  
   XSLT and XML, 22  
   XSLT and XSL, 21  
**<Place> attribute (Family tree example), 711**  
**place-knight() function (Knight's tour example), 745**  
**population order, 282**  
**Portable Document Format (PDF), 46**  
**portable stylesheets, writing**  
   conditional compilation, 122  
   extensibility, 128  
   version compatibility, 123  
**position() function, 79, 111, 371, 785**  
**possible moves (Knight's tour example)**  
   finding, 747  
   trying, 749  
**Post Schema Validation Infoset (PSVI), 146**  
**precedence. See import precedence**  
**preceding axis, 763**  
**preceding-sibling axis, 77, 763**  
**preserve validation, 162, 207, 250, 259, 268**  
**preserveWhitespace property (MSXML), 811**  
**principal stylesheet module, 85**  
**priority (of template rules), 71, 190, 453**  
**priority, default, 498**  
**procedural design patterns, 619**

**procedural languages**  
   C#, 6  
   Java, 6  
   Visual Basic, 6  
**procedure call, 220**  
**processing instruction node, 51**  
**processing-instruction(), 75**  
**producing lists (XML specification example), 670**  
**programming without assignment statements, 625**  
**progressive rendering, 36**  
**promotion, numeric, 73, 395**  
**pseudo-attributes in <?xml-stylesheet?>**  
   **processing instruction**  
     alternate, 94  
     charset, 94  
     href, 94  
     media, 94  
     title, 94  
     type, 94  
**PSVI (Post Schema Validation Infoset), 146**  
**publishing information to user, process of, 20**  
**publishing static HTML, 728**  
**pull processing, 69**  
**push processing**  
   examples of, 54, 65

## Q

**QName value space, 210**  
**QName, 130, 203**  
**QName-valued**  
   attribute, creating, 210  
   content, 349  
   elements, 210

## R

**recursion to process a sequence of nodes, 224**  
**recursion to process a sequence of strings, 227**  
**recursion, 223, 306, 633**  
**recursive templates, writing, 224**  
**reflection, 58**  
**refresh() function (Family tree example), 737**  
**regex. See <xsl:analyze-string> instruction**  
**regex-group() function, 181, 184, 580**  
**regular expression syntax, 179**  
**relational database, 7**  
**RelativePathPattern, 501**  
**RelaxNG, 25**  
**remove() function, 785**  
**replace() function, 182, 184, 620, 786**  
**required attribute (of xsl:param), 301, 393**  
**resolve-QName() function, 786**  
**resolve-uri() function, 787**  
**result document, validating, 160, 259, 415**

## result tree fragments

---

**result tree fragments, 81, 249**

**result tree**

writing to, 68

**reverse() function, 787**

**Rich Text Format (RTF), 46**

**root node, 49, 673**

**root() function, 788**

**round() function, 362, 788**

**round-half-to-even() function, 788**

**round-tripping. See namespace fixup**

**RTF (Rich Text Format), 46**

**Rubik's cube, 44**

**rule based design pattern, 37, 645**

**rule based stylesheets**

advantages of, 620

features of, 620

**rules for HTML output, 385**

**rules for text output, 389**

**running the stylesheet, 11, 752**

## S

**Sablotron, 8**

**save() method (MSXML), 801**

**SAX (Simple API for XML), 6**

**SAX API, 6**

**SAX filter application, 19**

**SAX2 API specification, 46**

**SAX-compliant parser, 703**

**Saxon**

collations, 858

command line, 852

evaluate() extension, 859

expression() extension, 860

extension functions, writing, 601, 857

JAVA API, 855

installing, 12

origins, 31

processor, 8, 851–864

**Saxon from a Java application, using, 855**

**Saxon from the command line, using, 852**

**Saxon processor, 601**

**Saxon processor, invoking, 852**

**Saxon tree models, 856**

**Saxon**

installing, 12

origins, 31

**Saxon, 8, 12, 31, 851**

**saxon:expression() extension function, 860**

**SAXResult class (JAXP), 233**

**Scalable Vector Graphics (SVG), 244**

**scene.xsl stylesheet, 622**

**<SCENE> element, 195**

**schema for GEDCOM 6.0, creating, 695**

**schema information in the tree model, 58**

**schema processor, 140, 161**

**schema, adding, 868**

**schema. See also XML Schema**

**SCHEMA\_VALIDATION property (Saxon), 732**

**schema-aware XSLT processor, 58, 145, 201, 262, 705, 729**

**schema-location attribute, 325, 326**

**schemas, importing, 168, 324, 705, 715**

**<scrap> element (XML specification example), 676**

**seconds-from-dateTime() function, 789**

**seconds-from-dayTimeDuration() function, 789**

**seconds-from-time() function, 789**

**section headers, creating, 666**

**select**

attribute, 22, 73, 347, 362

expression, 189

statement, 236

**select attribute, 18, 60**

**SELECTED attribute, 211**

**selecting nodes explicitly, example of, 69**

**selectNodes method (MSXML), 22**

**self axis, 763**

**separator attribute, 201, 206**

**sequence constructor, 60, 64, 106**

**sequence number, determining, 362**

**sequence of nodes, 279**

**sequence type descriptor**

attribute() +, 156

document-node(), 156

element(), 156

node() \*, 156

node(), 156

**SequenceType syntax, 74, 301, 327, 444**

**serialization, 45, 57, 375**

**service() method (Java servlets), 732**

**set() method (MSXML), 627**

**setErrorListener() method (JAXP), 826**

**setParameter() method (JAXP), 816**

**setProperty() method, 801**

**setting out the production rules, 676**

**SGML (Standard Generalized Markup Language), 27**

**SGML-based standard, 27**

**SGML syntax, 34**

**showing the ancestors of a node, example of, 279**

**side effects, 36, 609**

**Simple API for XML (SAX), 6**

**simple key, using, 336**

**simple type definitions, 146**

**simplified stylesheet. See fill-in-the-blanks stylesheets**

**simplified stylesheets**

advantages of, 119

examples of, 14, 120

**simulating higher order functions, 198**

**single-level grouping by value, example of, 286**

**single-match example, 182**

**SOAP namespaces, 23, 114****«some», 761****sort key**

- component, 425
- specification, 425
- value, 425

**sorted sequence, 425****sorting on the result of a calculation, example of, 431****sorting the groups, 285****sorting, 189, 278****source document, 307****source document, validating, 159****<SPEAKER> element, 293****<SPEECH> element, 225****SQL script, 2****SQL Server, 9****sql:connect() function (Saxon), 603****Standard Generalized Markup Language (SGML), 27****Standard Music Description Language, 4****start node, 362****starts-with() function, 789****static method, 131****strict validation, 161, 207, 250, 259, 268****string value of a node, 52, 138****string() function, 116, 127, 138, 790****string, 155****string-join() function, 790****string-length() function, 790****strings, 116****string-to-codepoints() function, 791****«strip» 162, 207, 250****stripping whitespace nodes, effect of, 141****stylesheet**

- debugging, 712
- program, 84
- structure, 83
- tree, 115

**stylesheet design patterns**

- computational stylesheets, 625
- fill-in-the-blanks stylesheets, 613
- navigational stylesheets, 616
- rule based stylesheets, 620

**stylesheet design patterns, 613****stylesheet functions, 129, 224****stylesheet functions, using, 303****stylesheet languages**

- Cascading Style Sheets (CSS and CSS2), 24
- XSL (XSLT plus XSL Formatting Objects), 24

**stylesheet layer, 316****stylesheet module**

- overview of, 119
- principal, 85

**stylesheet namespaces**

- local, 716
- schema, 716

XHTML, 716

XSLT, 716

**stylesheet parameters, 59, 74, 397****stylesheet-prefix attribute, 354****stylesheets and schemas, 145****stylesheets, 85****Stylus Studio, 9****subroutine call, 220****subsequence() function, 791****substitution groups, 150, 151****substring() function, 116, 791****substring-after() function, 792****substring-before() function, 792****subtract-dates-yielding-dayTimeDuration() function, 793****subtract-dates-yielding-yearMonthDuration() function, 793****subtract-dateTimes-yielding-dayTimeDuration() function, 793****subtract-dateTimes-yielding-year MonthDuration() function, 793****sum() function, 634, 793****super() method (in object-oriented programming), 184, 356****SVG (Scalable Vector Graphics), 244****SVG namespace, 244****switch statement, 236****syntax of patterns**

- syntax, 7, 125
- trees and DOM, 604
- type system, 251

**system overview of XSLT, 43****System.Xml (Microsoft .NET), 812****System.Xml.Xsl interface (Microsoft .NET), 133****system-property() function, 122, 124, 125, 346****T****table of contents, example of creating, 661****tables, arranging data in, 298****targetNamespace attribute (XML Schema), 326****template bodies, 84****template parameters, 74, 398****template rules**

- built-in, 70
- choosing, 189

**temporary tree, 79, 260****temporary tree, validating, 163****Tennison, Jeni, 643****terminate attribute (of xsl:message), 343****test expression, 236, 309****testing availability of a Java method, example of, 568****testing for node-set() extensions, example of, 566****Text Encoding Initiative, 4**

## text node

---

### text node

- in a sequence constructor, 64
- whitespace-only, 64

### text output method, 46, 389

### text(), 75

### text, formatting, 667

### timezone-from-date() function, 793

### timezone-from-dateTime() function, 793

### timezone-from-time() function, 793

### <to>, 761

### token output sequence, formatting, 365

### token symbols (in XPath)

- Char, 756
- DecimalLiteral, 755
- Digit, 756
- DoubleLiteral, 756
- IntegerLiteral, 755
- QName, 756
- StringLiteral, 756
- Wildcard, 756

### tokenize() function, 16, 182, 184, 224, 794

### top-level elements, 98

### top-level processing, 705

### trace() function, 794

### Transformation API for XML (TrAX), 815

### transformation process

- built-in template rules, 70
- conflict resolution policy, 71
- controlling which nodes to process, 68
- modes, 70
- push processing, 65
- sequence constructors, 61
- template rules, 60
- transformation, invoking, 59

### transformation process. See system overview of XSLT

### transformation processor, 57

### transformation sheet. See system overview of XSLT

### transformation, invoking, 59

### Transformix, 8

### translate() function, 367, 742, 795

### TrAX (Transformation API for XML), 815

### <treat as>, 762

### tree construction process, 64

### tree model

- nodes in tree model, 51
- Saxon implementation, 856
- XML as tree, 48

### true() function, 795

### tunnel attribute (of xsl:param), 393

### tunnel parameters, 398, 490

### type annotation of a node

- copying, 250
- description, 52
- setting and using, 146, 159, 164

### type attribute, 160, 161, 201, 207

### type conversion, 73, 496

### type of the parameter, 394

### type system for XSLT

- based on XML Schema, 41

### typed value of a node, 52

## U

### unary «+», 762

### unary «-», 762

### Unicode

- characters, 334
- codepoint collation, 333
- codepoints-to-string() function, 767
- collation algorithm, 429
- normalization, 232, 383
- Private Use Area, 234
- string-to-codepoints() function, 791

### Uniform Resource Identifier (URI), 22, 533

### Uniform Resource Locators (URLs), 533

### <union>, 762

### unordered() function, 795

### unparsed-entity-public-id() function, 584

### unparsed-entity-uri() function, 585

### unparsed-text() function, 587, 611

### up-conversion, 589

### upper-case() function, 796

### URI (Uniform Resource Identifier), 22, 533

### URI identifying a namespace, 262

### URI, resolving, 533

### URIResolver class, 133, 841

### URIs as atomic values, 538

### URIs held in nodes, 535

### URI-valued attributes, 231, 389

### URLs (Uniform Resource Locators), 533

### Usage section. See literal result elements

### use attribute (of xsl:key), 333, 337

### use-attribute-sets attribute, 215, 241, 262, 308

### use-character-maps attribute, 230

### user defined attributes, 124

### user defined top-level element

- user-defined data elements, 98
- XSLT-defined declarations, 98

### use-when attribute

- using, 85, 89
- with named attribute sets, example of, 331

## V

### validate() method, 802

### validating

- individual attributes, 167
- individual elements, 164
- result document, 160
- source document, 159
- temporary tree, 163

**validating and annotating the document,**  
**259**

**validation attribute, 161, 164, 201**

**validation attribute, values of**

lax, 162  
preserve, 162  
strict, 161  
strip, 162

**value attribute, 214, 362**

**value of a constructed attribute, 206**

**variables**

context, 78  
datatypes, 74  
expressions, 76  
global, 73  
immutability, 73  
local, 73  
parameters, 74  
temporary trees, 79

**variant stylesheets (XML specification example),**  
**685–688**

**VBScript in MSXML3 stylesheet, 600**

**<vc> element (XML specification example),**  
**678**

**vendor defined attributes, 129**

**vendor extensions**

permissible, 595  
testing for, 548, 566

**vendor or third-party extensions, testing for,**  
**566**

**vendor portability, 274**

**version attribute, 123, 124, 437**

**version compatibility, 123**

**VitalType attribute (family tree example), 718**

## W

**WD-xsl, 8, 30, 435**

**Web browsers**

Internet Explorer, 30, 95, 715  
Netscape, 95, 715

**Web client, 27**

**Web content, creating**

markup, 29  
program, 29  
script, 29

**Web sites, 739**

**well-balanced fragment, 50**

**well-formed XML document, 49**

**while loop, 223**

**whitespace**

control using <xsl:text>, 461  
effect of stripping whitespace nodes, 141  
in a sequence constructor, 64  
in MSXML3, 138  
solving whitespace problems, 142

stripping, 432

whitespace nodes in stylesheet, 64, 141

**whitespace characters**

carriage return, 137  
newline, 137  
space, 137  
tab, 137

**whitespace facet in XML Schema**

collapse, 138  
preserve, 138  
replace, 138

**whitespace handling, 136**

**whitespace nodes**

in the stylesheet, 64, 141

**whitespace problems, solving**

too little whitespace, 143  
too much whitespace, 142

## X

**Xalan processor, 8, 601**

**xdt:anyAtomicType, 62**

**xdt:dayTimeDuration, 63, 76**

**xdt:untypedAtomic, 63, 76**

**xdt:yearMonthDuration, 63, 76**

**XHTML output method, 46, 388**

**XML**

as a tree, 39, 44, 48  
attributes, 44, 55  
base attribute, 539  
data, 2  
documents, 1  
element, 24  
information set (InfoSet), 24  
lang attribute, 695  
Namespaces 1.1, 23  
namespaces, 22  
output method, 46, 378  
parser, 6, 137  
Query Language, 32  
space attribute, 60  
specification, formatting, 646  
syntax, use of, 34  
tags, 621  
tree model, 56  
vocabulary, 17

**XML-based electronic commerce, 2**

**XML-based model, 19**

**XML-defined IDs, 335**

**XML document features, categories of**

debatable, 56  
definitely insignificant, 56  
definitely significant, 56

**XML envelope/payload applications, 591**

**XML parser, 6**

**XML Query Language, 28**

## XML Schema

---

**XML Schema, 25, 92, 145, 210**

**XML Schema, overview of**

elements with attributes and simple content, 148

elements with element-only content, 150

elements with mixed content, 149

processing, 8

overview of, 145

rule-based, 37

types based, 41

simple type definitions, 146

substitution groups, 151

**XML specification, formatting, 646**

**XML Spy, 9**

**XML to HTML, transforming, 1**

**XML vocabulary, 21, 210**

**XML, features of**

separating data from presentation, 2

transmitting data between applications, 2

**xml:space attribute, 49**

**XMLFilter class (JAXP), 823**

**XMLNode, 812**

**XMLSpec, case study of, 645–46**

**XPath**

cast expression, 444

data model, 147, 163, 204

doc() function, 533

engine, 305

expression language, 7

expression syntax, 130

expressions, 76, 101, 125

function call, 68, 224

function library, 765

language, 18

relationship to XSLT, 21

**XPath 2.0 requirements, 33**

**xpath-default-namespace attribute, 304, 442**

**XPathDocument (.NET), 812**

**XPathNavigator object (.NET), 813**

**XPointer, 21, 860**

**XQuery, 9, 32, 131, 206**

**xs:anyURI, 63, 76**

**xs:boolean, 75**

**xs:date, 75**

**xs:dateTime, 75**

**xs:decimal, 73, 75**

**xs:double, 73, 75**

**xs:float() constructor function, 765**

**xs:float, 73**

**xs:ID, 73**

**xs:integer() constructor function, 72**

**xs:integer, 73, 75**

**xs:QName, 76, 204**

**xs:string, 73, 75**

**xs:time, 75**

**xs:token, 150**

**<xsl:apply-imports> instruction, 184**

**<xsl:apply-templates> instruction**

changes in 2.0, 184

effect, 185

format, 184

usage and examples, 185

**xsi:nil attribute, 171**

**xsi:type attribute, 170**

**XSL (Extensible Stylesheet Language), 21**

**XSL and CSS, 24**

**XSL Formatting Objects (XSL-FO), 21**

**XSL, capabilities of**

creation of formatting constructs, 29

definition of reusable formatting macros, 29

extensible set of formatting objects, 29

formatting constructs, creating, 25

formatting of source elements, 29

source elements, formatting, 24

writing-direction independent stylesheets,

29

**XSL, history of**

beyond XSLT 1.0, 31

Microsoft WD-xsl dialect, 8, 30, 435

prehistory, 26

Saxon, 31

XQuery, 32

XSLT 2.0 and XPath 2.0, 33

**<xsl:analyze-string> instruction, 176, 227**

**<xsl:apply-imports> instruction, 184**

**<xsl:apply-templates> instruction**

changes in 2.0, 187

effect, 188

format, 187

usage and examples, 193

**<xsl:apply-templates> versus <xsl:for-each>, 194**

**<xsl:attribute> instruction**

changes in 2.0, 201

effect, 202

examples, 211

format, 201

usage, 208

**<xsl:attribute> instruction, attributes of**

name, 202

select, 202

separator, 202

type, 202

validation, 202

**xsl:attribute-set**

changes in 2.0, 214

effect, 215

examples, 217

format, 214

usage, 217

**<xsl:attribute-set> declaration, 214**

**<xsl:attribute-set> declaration, attributes of**

name, 215

use-attribute-sets, 215

**xsl:call-template**

changes in 2.0, 220  
 effect, 221  
 format, 220  
 usage and examples, 222

**<xsl:character-map> declaration**

changes in 2.0, 229  
 effect, 230  
 format, 229  
 usage and examples, 232

**xsl:choose**

changes in 2.0, 236  
 effect, 236  
 examples, 237  
 format, 236  
 instruction, 213, 227, 236, 237  
 usage, 237

**xsl:comment**

changes in 2.0, 238  
 effect, 239  
 examples, 240  
 format, 238  
 instruction, 238  
 usage, 239

**xsl:copy**

changes in 2.0, 241  
 effect, 242  
 examples, 244  
 format, 241  
 instruction, 205, 216, 240, 350

**<xsl:copy>**

attributes of, 241  
 changes in 2.0, 245  
 copy-namespaces, 241  
 effect, 246  
 example of, 244  
 format, 245  
 significance of, 243  
 type, 241  
 usage and examples, 247  
 validation, 241

**<xsl:copy-of>**

attributes of  
 copy-namespaces, 245  
 for repeated output source, example of, 247  
 instruction, 245, 350  
 select, 245  
 type, 245  
 validation, 245

**xsl:decimal-format**

attributes of  
 changes in 2.0, 252  
 declaration, 251, 359  
 decimal-separator, 252  
 digit, 253  
 effect, 253  
 examples, 255

format, 252  
 grouping-separator, 252  
 infinity, 252  
 minus-sign, 252  
 name, 252  
 NaN, 253  
 pattern-separator, 253  
 percent, 253  
 per-mille, 253  
 usage, 255  
 zero-digit, 253

**xsl:document**

changes in 2.0, 257  
 effect, 258  
 format, 257  
 instruction, 257  
 usage and examples, 260

**xsl:element**

changes in 2.0, 260  
 effect, 261  
 format, 261  
 instruction, 62, 260  
 usage and examples, 269

**xsl:exclude-result-prefixes attribute, 114****xsl:fallback**

changes in 2.0, 271  
 effect, 272  
 examples, 274  
 format, 271  
 instruction, 106, 125, 135, 271  
 usage, 273

**xsl:for-each**

changes in 2.0, 276  
 effect, 277  
 format, 277  
 instruction, 276  
 usage and examples, 279

**xsl:for-each-group**

attributes of, 281  
 changes in 2.0, 281  
 collation, 282  
 effect, 282  
 format, 281  
 group-adjacent, 282  
 group-by, 282  
 group-ending-with, 282  
 group-starting-with, 282  
 instruction, 281  
 select, 282  
 usage and examples, 286

**xsl:function**

as, 301  
 attributes of  
 changes in 2.0, 300  
 declaration, 35, 73, 131, 300  
 effect, 301  
 format, 300

## xsl:function (continued)

---

### **xsl:function (continued)**

name, 301  
 override, 301  
 usage and examples, 303

### **xsl:if**

changes in 2.0, 309  
 effect, 309  
 examples, 311  
 instruction, 102, 309  
 format, 309  
 usage, 310

### **xsl:import**

changes in 2.0, 312  
 declaration, 47, 88, 98, 312  
 effect, 313  
 examples, 320  
 format, 312  
 usage, 319

### **xsl:import-schema**

changes in 2.0, 324  
 declaration, 163, 168, 324  
 effect, 325  
 examples, 328  
 format, 325  
 usage, 326

### **xsl:include**

changes in 2.0, 328  
 declaration, 47, 88, 328  
 effect, 329  
 format, 328  
 usage and examples, 330

### **<xsl:include> and <xsl:import>, difference between, 88**

### **xsl:key**

attributes of  
 changes in 2.0, 332  
 collation, 332  
 declaration, 332  
 effect, 333  
 format, 332  
 match, 332  
 name, 332  
 usage and examples, 335  
 use, 332

### **xsl:matching-substring**

changes in 2.0, 342  
 effect, 343  
 element, 181, 182, 342  
 format, 342  
 usage and examples, 343

### **xsl:message**

changes in 2.0, 343  
 effect, 344  
 examples, 345  
 instruction, 102, 343  
 format, 343  
 usage, 344

### **xsl:namespace**

changes in 2.0, 347  
 effect, 347  
 format, 347  
 instruction, 346  
 usage and examples, 348

### **xsl:namespace-alias**

changes in 2.0, 350  
 effect, 351  
 format, 350  
 usage and examples, 352

### **<xsl:namespace-alias> instruction, 116, 350**

### **xsl:next-match**

changes in 2.0, 355  
 effect, 356  
 format, 355  
 instruction, 355  
 usage and examples, 357

### **xsl:non-matching-substring**

changes in 2.0, 359  
 effect, 359  
 element, 184, 358  
 format, 359  
 usage and examples, 359

### **xsl:number**

attributes of  
 changes in 2.0, 360  
 count, 360  
 effect, 361  
 format, 360  
 format, 361  
 from, 360  
 instruction, 359  
 lang, 361  
 letter-value, 361  
 level, 360  
 select, 360  
 usage and examples, 367  
 value, 360

### **xsl:otherwise**

changes in 2.0, 374  
 effect, 374  
 element, 237, 374  
 format, 374  
 usage and examples, 375

### **xsl:output**

attributes of  
 changes in 2.0, 375  
 effect, 377  
 examples, 391  
 format, 375  
 usage, 390

### **<xsl:output> declaration, 46, 375**

cdata-section-elements, 376, 381  
 doctype-public, 376, 382  
 doctype-system, 376, 382  
 encoding, 376, 382

**<xsl:stylesheet> element, attributes of**

- escape-uri-attributes, 376
- include-content-type, 376
- indent, 376, 383
- media-type, 376, 383
- method, 376
- name, 376
- normalization-form, 376, 383
- omit-xml-declaration, 377, 384
- standalone, 377, 384
- undeclare-namespaces, 377, 384
- use-character-maps, 377, 385
- version, 377, 385
- <xsl:output>, attributes of**
  - cdata-section-elements, 376, 381
  - doctype-public, 376, 382
  - doctype-system, 376, 382
  - encoding, 376, 382
  - escape-uri-attributes, 376
  - include-content-type, 376
  - indent, 376, 383
  - media-type, 376, 383
  - method, 376
  - name, 376
  - normalization-form, 376, 383
  - omit-xml-declaration, 377, 384
  - standalone, 377, 384
  - undeclare-namespaces, 377, 384
  - use-character-maps, 377, 385
  - version, 377, 385
- xsl:output-character element, 391**
  - changes in 2.0, 391
  - effect, 392
  - format, 392
- xsl:param**
  - changes in 2.0, 393
  - effect, 394
  - element, 99, 356, 392
  - examples, 399
  - format, 393
  - usage, 397
- <xsl:param> element, attributes of**
  - as, 393
  - name, 393
  - required, 393
  - select, 393
  - tunnel, 393
- <xsl:param> with a default value, example of, 399**
- xsl:perform-sort**
  - changes in 2.0, 405
  - effect, 405
  - format, 405
  - instruction, 405
  - usage and examples, 406
- xsl:preserve-space**
  - changes in 2.0, 406
  - effect, 407
- element, 406
- examples, 410
- format, 406
- usage, 409
- <xsl:preserve-space> declaration, 406**
- xsl:processing-instruction, 411**
  - changes in 2.0, 411
  - effect, 411
  - examples, 413
  - format, 411
  - usage, 412
- xsl:result-document**
  - changes in 2.0, 414
  - effect, 415
  - examples, 417
  - format, 414
  - usage, 417
- <xsl:result-document> instruction, 47, 414**
- xsl:sequence**
  - changes in 2.0, 420
  - effect, 420
  - element, 420
  - format, 420
  - usage and examples, 421
- <xsl:sequence> instruction, 200, 226, 420**
- xsl:sort**
  - changes in 2.0, 423
  - effect, 424
  - examples, 431
  - element, 423
  - format, 423
  - usage, 429
- <xsl:sort> attributes of**
  - case-order, 424
  - collation, 424
  - data-type, 424
  - lang, 424
  - order, 424
  - select, 424
  - stable, 424
- <xsl:sort> element, 189, 278, 423**
- xsl:strip-space**
  - changes in 2.0, 433
  - declaration, 432
  - effect, usage, and examples, 433
  - format, 433
- <xsl:strip-space> declaration, 432**
- xsl:stylesheet**
  - changes in 2.0, 434
  - effect, 437
  - format, 434
  - usage and examples, 445
- <xsl:stylesheet> element, 91, 433**
- <xsl:stylesheet> element, attributes of**
  - default-validation, 92, 168
  - exclude-result-prefixes, 92
  - extension-element-prefixes, 92

## <xsl:stylesheet> element, attributes of (*continued*)

---

### <xsl:stylesheet> element, attributes of (*continued*)

id, 92  
 <template> declaration, 103  
 <template> element, 115  
 <transform>, 98  
 <value-of> element, 102  
 <value-of> instruction, 127  
 <variable> element, 61, 99  
 <when> element, 236  
 xpath-default-namespace, 92

#### **xsl:template**

changes in 2.0, 451  
 declaration, 450  
 effect, 452  
 format, 451  
 usage and examples, 455

#### **xsl:text**

changes in 2.0, 460  
 effect, 460  
 format, 460  
 instruction, 459  
 instruction, 114, 459  
 usage, 460

#### **xsl:transform**

element, 91, 465  
 format, 465

#### **xsl:type**

attribute, 166  
 validation attribute, 166  
 version attribute, 119

#### **xsl:value-of**

changes in 2.0, 465  
 effect, 466  
 examples, 470  
 format, 465  
 instruction, 465  
 usage, 467

#### **xsl:variable**

changes in 2.0, 471  
 effect, 472  
 element, 471  
 examples, 477  
 format, 471  
 usage, 477

#### **xsl:when**

changes in 2.0, 487  
 effect, 487  
 format, 487  
 usage and examples, 488

#### **<xsl:when> element, 487**

#### **xsl:with-param**

changes in 2.0, 489  
 effect, 489  
 element, 488  
 format, 489  
 usage and examples, 490

#### **[xsl]version attribute, 273**

### **XSL-FO (XSL Formatting Objects), 21**

#### **XSLT**

engine, 19  
 forwards compatibility, 274  
 namespace, 98, 351, 438  
 overview of, 1  
 processor, 6, 47, 92, 115, 168  
 processor, core task of, 43  
 Recommendation, 46  
 significance of, 17  
 stylesheet, 8, 11  
 template rules, 116  
 type system, 72  
 uses of, 4

#### **XSLT (eXtensible Stylesheet Language Transformations), 1**

#### **XSLT 1.0 processor, 122**

#### **XSLT 1.0 stylesheet, overview of, 9**

#### **XSLT 1.0 stylesheets to XSLT 2.0, migration of, 124**

#### **XSLT 2.0 and XPath 1.0, features of, 619**

#### **XSLT 2.0 as a language**

Rule based, 37  
 types based on XML schema, 41  
 XML syntax, use of, 34

#### **XSLT 2.0 behavior and XSLT 1.0 behavior, differences between, 126**

#### **XSLT 2.0 processor, 122**

#### **XSLT 2.0 requirements, 33**

#### **XSLT 2.0 stylesheet, 15, 541**

#### **XSLT 2.0, changes in, 84, 494**

#### **XSLT 2.0, features of, 122**

#### **XSLT and SQL**

overview of, 7  
 similarities between, 7

#### **XSLT and XML**

relationship between, 18  
 schemas, 21

#### **XSLT and XML schemas, 25**

#### **XSLT and XML, relationship between, 22**

#### **XSLT and XPath processing objects (MSXML)**

IXMLDOMDocument, 800  
 IXMLDOMNode, 800  
 IXMLDOMNodeList, 800  
 IXMLDOMParseError, 800  
 IXMLDOMSelection, 800  
 IXSLProcessor, 800  
 IXSLTemplate, 800

#### **XSLT and XPath, 21**

#### **XSLT and XSL, 21**

#### **XSLT defined attributes, 129**

#### **XSLT element, 122**

#### **XSLT engine, 16**

#### **XSLT functions**

current() function, 526  
 current-group() function, 523  
 current-grouping-key(), 523

---

## zero-or-one() function

document(), 524  
 element-available(), 524  
 format-date(), 524  
 format-number(), 524  
 format-time(), 524  
 function-available(), 524  
 generate-id(), 524  
 key(), 524  
 regex-group(), 524  
 system-property(), 524  
 unparsed-entity-public-id(), 524  
 unparsed-text(), 524  
**XSLT in data conversion, role of, 2**  
**XSLT instructions. See instructions**  
**XSLT processing model, 43**  
**XSLT processor, 5, 39**  
**XSLT processor, core task of, 36**  
**XSLT Recommendation, 38**  
**XSLT stylesheet, 7, 9**  
**XSLT to XML, transformation from, 5**  
**XSLT to XML, transforming**  
   XSLT 1.0 stylesheet, 9  
   XSLT 2.0 stylesheet, 15  
   XSLT and SQL, 7  
   XSLT processors, 8  
**XSLT transformations, using**  
   data conversion, 17  
   publishing, 17  
**XSLT tree model, 6, 48**  
**XSLT version 2.0, features of**  
   extending the scope of applicability, 3

integration across the XML standards family, 3  
 tactical usability improvements, 3

**XSLT, 1****XSLT, overview of**

transforming music, 3  
 version 2.0, 3

**XSLT, system overview of**

different output formats, 45  
 multiple inputs and outputs, 47  
 trees, not documents, 44

**XSLT, use of**

data conversion applications, 17  
 publishing, 19

**XSLTC, 8****XSLT-defined declarations. See declarations****XSLT-defined elements, 129****XSLT-defined functions, 130****xt processor, 601****xt, 8****Y****year-from-date() function, 796****year-from-dateTime() function, 796****years-from-yearMonthDuration() function, 796****Z****zero-or-one() function, 796**

