

Index

A

abbreviated notation, XPath, 189–191
absolute location steps, XPath, 189
Abstract Factory design patterns, 42
Abstract Syntax Notation 1.
 See ASN.1
access controls
 declarative, 550, 551, 552–553
 policies, 524
 programmatic, 550, 551, 553–554
 server architecture, 554–556
Active Server Pages (ASP), 297
algorithms, Dijkstra's, 252, 256
ANSI standards *versus* W3C recommendations, 17
Apache Axis, 458–489
Apache Cocoon, 298
 basics, 344
 XML documents
 goals for handling, 344–346
 integrating and multichanneling, 346–351
Apache SOAP, 453–458
Apache Xerces (in text). *See* Xerces
APIs (application programming interfaces). *See also* DOM; JAX-RPC; JAXM; JAXP; JDBC; JTA; SAX; Servlet
 ease of using Web applications, 7
 application data structure mapping from XML document structure with XSLT, 243–246
 to isomorphic XML document structure, 236, 237–243
 application programming interfaces.
 See APIs

 application-specific query languages, 355
 applications. *See* Web applications
 ASCII character encoding, 83
 ASN.1 (Abstract Syntax Notation 1)
 versus XML, 12
 ASP (Active Server Pages), 297
 Attr DOM interface, 46
 attributes, DOM trees, 94–95
 authentication
 clients, 526–527
 servers, 525–526
 axes, XPath, 189–191
 Axis (Apache), 458–489

B

B2B (business-to-business) applications, 1
 CORBA rare usage, 411
 document-centric messaging, 411–412
 security concerns, 521, 537
 SOAP usage, 426
 support by J2EE, 298
 emergence, 295
 retrieving information with HTTP and HTML, 7–9
 transition from B2C to B2B, 7–9
 XML documents, 10–11
 XML messaging, demand increase, 412
B2C (business-to-consumer) applications
 emergence, 295
 retrieving information with HTTP and HTML, 7–9

 support by J2EE, 298
 transition from B2C to B2B, 7–9
 XML documents, 10–11
BEA's WebLogic, 295, 296
Bean-Managed Persistence (BMP), EJB entity beans, 403, 406
bind/publish/find operations. *See* Web services
BLOB (Binary Large Object) datatype, 361
body of envelopes, 410–411
Boolean type, XPath, 191
 location step functions, 192
 object handling, 198–199
built-in templates, XSLT, 207
business entities and services, UDDI, 492–494
business-to-business applications.
 See B2B applications
business-to-consumer applications.
 See B2C applications

C

Candidate Recommendations, W3C, 17
canonicalization, XML Digital Signature, 538–542
cascading models, Servlet and JSP, 337, 338
Castor XML data binding, 574–579
 pros and cons, 579–581
CD-ROM contents, 611
 running sample programs, 2
CDATASection DOM interface, 47
CDF metadata format, 16
certificates. *See* Digital certificates
CGI (Common Gateway Interface), 6

- chaining models, Servlet and JSP, 337, 338
- character encodings
 - charset parameter, 84
 - internationalization, 83–84
 - UTF-8 and UTF-16, 77
 - versus* character sets, 13
- character length, databases, 361
- character sets, Unicode, 13
 - versus* character encodings, 13
- CharacterData DOM interface, 47
- charset parameter, 84
- child elements, whitespace recommendations, 80
- child nodes, DOM trees, 89
 - creating/appending, 62–67
 - processing order, 90–91
- choke points, access controls, 551
- CLOB (Character Large Object)
 - datatype, 361
- co-occurrence constraints, RELAX NG schema, 290–292
- Cocoon (Apache), 298
 - basics, 344
 - XML documents
 - goals for handling, 344–346
 - integrating and multichanneling, 346–351
- COM (Component Object Model), 14
 - SOAP functions, 458
- Comment DOM interface, 48
- Common Gateway Interface. *See* CGI
- Common Object Request Broker. *See* CORBA
- Component Object Model. *See* COM
- connectivity. *See* interoperability
- Container-Managed Persistence (CMP), EJB entity beans, 402–403, 406
- containers
 - EJB and Web, 402
 - JSP (JavaServer Pages), 331–334
 - Servlet, 302–303
- ContentHandler SAX interface, 48–52
 - problems with characters() methods, 51, 109–113
- context nodes, XPath, 189
- CORBA (Common Object Request Broker)
 - IDL, 411
 - RPC standards, 408
 - SOAP, 426
 - XML, 13
- Core DOM feature, 123–124, 127, 128
- Crimson parser (JAXP), lack of support for DOM traversal, 98
- Cross-site Scripting. *See* CSS
- crypto service providers, security, 533
- cryptography architecture
 - SSL/TLS, 533–536
- CSS (Cross-site Scripting)
 - problem in serialization, 77
 - Servlet, 307–310
- D**
- data binding, 1, 165. *See also* datatypes
 - defined, 561
 - generating Java programs from schemas
 - JAXB, 562–566
 - pros and cons, 573
 - Relaxer, 567–573
 - generating XML documents from
 - Java classes, 573–574
 - Castor XML, 574–579
 - pros and cons, 579–581
 - SOAP encoding, 582–584
- data format language features of XML, 12
- data models, schema languages, 592–601
- databases. *See also* RDBMSs
 - DB2, CD-ROM contents, 2
- datatypes. *See also* data binding; types
 - RDBMSs, mapping and modeling, 361–362
 - XML Schema, 288–289
 - basics, 270–272
 - facets, 272–276
- date and time datatypes, 270–272
- DB2 database, 355
 - CD-ROM contents, 2
- DCM (document-centric messaging), 411–412
 - JAXM, 459
 - SOAP, 422
- DCOM (Distributed Component Object Model)
 - interoperability, 411
 - XML, 13
- decentralized Web applications to distributed applications, 13–14
- declarative access controls, 550, 551, 552–553
- deep copies, nodes, 92
- DefaultHandler SAX interface, 48
- Deferred DOM feature, 123–124, 127, 128
- deployment descriptors
 - EJB, 405
 - Servlet, 306–307
 - distributable, 329
- digital certificates
 - client-based authentication, 528–530
 - public-key infrastructure, 528–530
 - SSL/TLS, 525–526
- digital signatures
 - XML Security Suite for Java, signing, 542–547
 - XML Security Suite for Java, verifying, 547–550
- Dijkstra’s algorithm, 252, 256
- distributed Web applications
 - history, 407–408
 - interoperability, 408–409
 - to decentralized applications, 13–14
 - XML messaging, document-centric, 411–412
- DMZs (Demilitarized Zone)
 - firewalls, SSL/TLS, 536–537
 - SOAP, 421–422
- Document Definition Types. *See* DTDs
- Document DOM interface, 47
 - factory methods, 62–63
 - implementation classes, 61
- Document Object Model. *See* DOM
- document order, XPath, 190
- document-centric messaging. *See* DCM
- DocumentFragment DOM interface, 47
- DocumentType DOM interface, 47
- DOM (Document Object Model), 44
 - Deferred, Non-deferred, and Core DOM features, 123–124, 127, 128
 - DOM trees/nodes
 - child nodes, 89
 - child nodes, creating/appending, 62–67
 - child nodes, deleting, 99–100
 - child nodes, manipulating, 92–94
 - child nodes, processing order, 90–91

- creating new nodes, 91–92
 - moving nodes between documents, 100
 - namespaces, 67–70
 - parent nodes, 88
 - previous siblings, 89
 - processing order, 90–91, 97–98
 - serializing, 75–78
 - serializing, defined, 60
 - serializing, with XMLSerializer package, 74–75
 - shallow/deep node copying, 92
 - status, accessing/updating, 86–87
 - structure, 87–88
 - traversing, 97–98
 - validating generation (alternate technique), 70–74
 - entity references, deleting, 96–97
 - namespaces
 - adding declarations, 102–108
 - automatic no namespace declarations, 101–102
 - basics, 100–101
 - pros and cons
 - development efficiency, combining XPath and DOM, 226–231
 - execution efficiency, 225
 - XML document conversion, comparison of SAX/DOM and XSLT, 231–233
 - tree-based interfaces, 45–46, 45–48
 - validating generation, 60
 - versus* SAX, 44, 55
 - converting DOM trees to SAX events, 128–134
 - converting SAX events to DOM trees, 134–141
 - memory, 120–124
 - speed, 125–128
 - DOMException DOM interface, 48
 - DOMImplementation DOM interface, 48
 - DTDHandler SAX interface, 48
 - DTDs (Document Type Definitions), 18
 - compatibility datatypes, 270–272
 - document scanners, Xerces2, 179
 - general purpose schema language, 603
 - internationalization, 83–84
 - namespaces, validating, 144–146
 - RELAX NG schema, mimicking DTDs
 - comments, 285–286
 - constructs, 286–287
 - declarations, attribute-list, 284–285
 - declarations, element type, 282–284
 - versus* XML Schema, 259–260
 - XML document parsing, valid documents, 29–30
 - XML Schema mimicking DTDs
 - all constructs, 268–270
 - attribute-list declarations of DTDs, 264–266
 - comments, 266–268
 - XNI Interface, 169
 - dynamic Web applications, transition from static to dynamic contents, 6–7
- E**
- EAI (Enterprise Application Integration), 407
 - Web services, 466, 517–519
 - EAR (Enterprise Application Archive), 517
 - EDI (Electronic Data Interchange)
 - emergence, 408
 - EJB (Enterprise JavaBeans), 296
 - containers, 402
 - RDBMS access, 401–406
 - security architecture, 557
 - SOAP, 426
 - functions, 458
 - Electronic Data Interchange (EDI)
 - emergence, 408
 - Element DOM interface, 47
 - encoding schemes. *See* Character encoding
 - encoding, SOAP, 422–425
 - encryption, XML, 558–559
 - engines (SOAP)
 - Apache Axis, 458–489
 - Apache SOAP, 453–458
 - examples
 - application for travel reservations, 427–434
 - header processing, 448–453
 - transport layers, abstracting to support JMS, 434–447
 - transport layers, intermediary support, 447–448
 - Enterprise Application Archive. *See* EAR
 - Enterprise Application Integration (EAI), 407
 - Enterprise JavaBeans. *See* EJB
 - entities
 - caching in memory, 148–150
 - entity beans (EJB), 402–403
 - entity resolution, 146–147
 - managing in Xerces2, 179
 - Entity DOM interface, 47
 - EntityReference DOM interface, 47
 - envelopes, headers and body, 410–411
 - SOAP, 421–422
 - processing steps, 448–452
 - error handling, 31–33
 - errors *versus* fatal errors, 34
 - error reporter, Xerces2, 179
 - ErrorHandler SAX interface, 48
 - event handlers (SAX), 48–55. *See also* SAX
 - filters, 114–117
 - eXcelon, 355
 - exception handling, 27–29
 - Extensible Markup Language. *See* XML
 - Extensible Stylesheet Language Transformations. *See* XSLT
- F**
- facets, XML Schema datatypes, 272–276
 - using, 288–289
 - Factory Method design patterns, 42
 - factory methods, Document DOM interface, 62–63
 - feature/property mechanism, SAX, 120
 - filters, SAX, 119
 - using, 114–117
 - writing, 117–119
 - find/publish/bind operations. *See* Web services
 - firewalls, SSL/TLS, 536–537
- G**
- general notation, XPath, 189–191
 - graph structures, mapping from XML documents, 251–257
 - GUIs (graphical user interfaces), Web application ease of use, 7
- H**
- handlers. *See* Event handlers
 - handling errors. *See* Error handling
 - handling exceptions. *See* Exception handling

- hash tables, mapping from XML documents, 247–251
 - headers of envelopes, 410–411
 - SOAP, 421–422
 - HTML (Hypertext Markup Language)
 - HTML parser, 165
 - MVC model, 200
 - retrieving Web application information, 7–9
 - XSLT, 200
 - HTTP (Hypertext Transfer Protocol)
 - client authentication, 526–527
 - HTTP Content-Type headers, 314
 - SOAP, 421
 - authentication, 458
 - Web applications, 6
 - retrieving information, 7–9
 - Web services, not required, 463
 - XML messaging, 409
 - Hypertext Markup Language. *See* HTML
 - Hypertext Transfer Protocol. *See* HTTP
- I**
- IBM
 - MQSeries, 410
 - Web services architecture, 465
 - WebSphere, 295, 296
 - IDL (Interface Definition Language)
 - CORBA, 411
 - defined by OMG, 408
 - DOM language bindings, 46
 - IIOP (Internet Inter ORB Protocol), 410
 - instructions, XSLT templates, 204–205, 211
 - Interface Definition Language. *See* IDL
 - intermediary, SOAP, 420–422
 - International Telecommunications Union-Telecommunication Standardization Sector. *See* ITU-T
 - internationalization, XML declarations, 83–84
 - Internet Inter ORB Protocol. *See* IIOP
 - Internet resources. *See also* Appendix B
 - XML and Java
 - free downloads, 1
 - in-depth/latest information, 2
 - interoperability of Web applications, 11–13
 - distributed Web applications, 408–409
 - tightly *versus* loosely coupled applications, 411–412, 426
 - SOAP, 426
 - iPlanet (Sun), 295, 296
 - ISO i0646. *See* Unicode character set
 - ISO standards *versus* W3C recommendations, 17
 - ISO/IEC schema, alternatives to XML Schema, 260
 - isomorphic XML documents, mapping to application data structures, 236, 237–243
 - ITU-T (International Telecommunications Union-Telecommunication Standardization Sector), digital certificates, 525
- J**
- J2EE (Java 2 Enterprise Edition), 296, 297
 - Java. *See also* Servlet
 - DOM language bindings, 46
 - Internet resources
 - free downloads, 1
 - in-depth/latest information, 2
 - Web site (official), 2
 - Java 2 Enterprise Edition (J2EE), 296, 297
 - Java 2 Software Development Kit. *See* SDK
 - Java API for XML Messaging. *See* JAXM
 - Java API for XML Processing, JAXP
 - Java API for XML RPC. *See* JAX-RPC
 - Java classes, data binding, pros and cons, 579–581
 - Java Community Process (JCP), 296
 - JAXP, 211
 - Java Cryptography Architecture. *See* JCA
 - Java Cryptography Extension. *See* JCE
 - Java Database Connectivity (JDBC), 354. *See also* Appendix D
 - Java Message Service. *See* JMS
 - Java Naming and Directory. *See* JNDI
 - Java Runtime Environment. *See* JRE
 - Java Secure Socket Extension. *See* JSSE
 - Java Specification Request. *See* JSR
 - Java Transaction API. *See* JTA
 - JavaBeans, 14
 - JavaServer Pages. *See* JSP
 - JAX-RPC (Java API for XML RPC), 459–460
 - WSDL, 490–491
 - JAXB data binding, 562–566
 - JAXM (Java API for XML Messaging), 459–460
 - JAXP (Java API for XML Processing)
 - creating
 - DOM Document objects, 61
 - DOM trees, 66–67
 - Crimson parser, lack of support for DOM traversal, 98
 - defined by JCP (Java Community Process), 211
 - SAX events, XSLT translations, 223–224
 - XML document parsing with
 - Xerces, 40–43
 - programming patterns, 55–58
 - versus* Xerces native API, 43
 - JCA (Java Cryptography Architecture), 533
 - JCE (Java Cryptography Extension), 533
 - JCP (Java Community Process)
 - JAXB data binding, 562
 - JAXP, 211
 - JDBC (Java Database Connectivity), 354. *See also* Appendix D
 - Jing, 292
 - JMS (Java Message Service), 402, 403
 - SOAP, 421
 - transport layers, 434–447
 - JNDI (Java Naming and Directory), 402
 - JRE (Java Runtime Environment)
 - SDK platforms, 2
 - JSP (JavaServer Pages), 297
 - basics, 330–331
 - combining with Servlet, 337–343
 - containers, 331–334
 - document-centric model, 337
 - expressions, 332
 - returning XML documents, 333–334
 - Cross-site Scripting countermeasures, 334–335
 - serializing documents, 335–337
 - syntax, 331
 - JSR (Java Specification Request) Web services, 466
 - JSSE (Java Secure Socket Extension), 533–536
 - JTA (Java Transaction API), 402

K

- key pairs, authentication, 528, 531–533
- keywords, XML documents, 226–227

L

- language bindings. *See* DOM
- LexicalHandler interface *versus* other interface, 166
- licensing, Xerces XML processor, 22
- listings
 - Apache Cocoon, 346–351
 - data binding
 - Castor XML, 574–578, 580
 - JAXB, 563–567
 - Relaxer, 568–573
 - SOAP, 583
 - DOM trees/nodes
 - converting from SAX events, 135–139
 - converting to SAX events, 129–132
 - creating, 64–65
 - creating classes, Album, 160
 - creating classes, Artist, 160–161
 - creating classes, DOM utility, 161–163
 - creating classes, MusicCollection, 158–159
 - creating classes, Title, 161
 - creating, with JAXP API, 66–67
 - creating, with namespaces, 68–70
 - document factory, 163–164
 - DOMConstructor usage, 139–140
 - DOMReader usage, 132–134
 - factory methods of Document interface, 62–63
 - namespaces, correcting problems, 103–107
 - print memory for parsing, 122–123
 - program displaying artists, 164–165
 - EJB, RDBMS access, 403–405
 - JAXP parser/processor, 40–43
 - programming patterns, 57–58
 - JSP
 - combining with Servlet, 339–343
 - returning XML documents, 334, 335, 336–337
 - simple example, 330–331

- mapping to application-specific structures with XSLT
 - transformed purchase order, 244–245
 - XSLT transformation script, 245–246
- mapping to graph structures
 - Edge class, 254
 - graph represented as XML, 252–253
 - ShortestPath class, 254–256
 - ShortestPath class, main() method, 256–257
 - Vertex class, 253
- mapping to hash tables
 - Config class, 249–251
 - configuration file, 248
- mapping to isomorphic XML document structure
 - Customer class, 238
 - Customer class, unmarshal() method, 241
 - Item class, 238
 - Item class, unmarshal() method, 241–242
 - purchase order document, 237
 - PurchaseOrder class, 238
 - PurchaseOrder class, main() method, 242–243
 - PurchaseOrder class, unmarshal() method, 239–240
 - TypeConversion class, 240–241
- mapping to/from XML documents and RDBMSs tables, 358–359, 367–382
 - flat representation, 365–367
 - nested representation, 365
- programming in Java
 - SAX event handler, 217–219, 224
 - SAX event handler, translating using XSLT, 220–221
 - stylesheets, 221–222
- RELAX NG schema
 - co-occurrence constraints, 290–292
 - mimicking DTDs, 282–286
 - namespaces, 289–290
- SAX
 - converting DOM trees to SAX events, 129–132
 - converting SAX events to DOM trees, 135–139
 - event tracing, 48–55
 - filters, 117–118
 - memory, 121–122
 - speed, print parsing for SAX and DOM, 125–127
 - text processing, 111–113
- schema languages
 - as data models, 595–596, 599
 - as syntactic constraints, 586–587, 589–590
 - RDF Schema, 607
 - RELAX Namespace, 608
- security, access controls, 552, 553
- Servlet, 300–305
 - Cross-site scripting, 307–309
 - RDBMS access, 398–400
 - XML documents, client communicating with servlet, 326–327
 - XML documents, processing shopping carts, 321–324
 - XML documents, receiving, 312–313
 - XML documents, receiving requests from clients, 315–320
- SOAP
 - examples, 413–418, 419–420
 - JAXM, 459–460
 - RPC messages, 423–424
 - XML Schema, 424–425
- SOAP engines
 - Apache SOAP, 454–457
 - application for travel reservations, 428–434
 - header processing, 448–452
 - transport layers, abstracting to support JMS, 434–447
 - transport layers, intermediary support, 447–448
- sockets
 - streams, unwrapping input, 154–155
 - streams, wrapping output, 154
- SSL/TLS, JSSE, 535–536
- UDDI, 493–495
- UDDI and WSDL, applying to dynamic e-business, 510–516
- UDDI4J, 496, 498–506
- WSDL
 - IDL, 476, 477–478
 - SOAP messaging, 468–470, 473–474
 - WSTK, 483–489

- listings (*cont.*)
 - Xerces parsers/processors
 - DOM interface, 157–165
 - invalid documents, 33–34
 - namespaces, 35–36, 36–38
 - not well-formed documents, 28–29
 - programming patterns, 56–57
 - valid documents, 30–33
 - valid documents, with DTDs, 29–30
 - well-formed documents, 25–28
 - XML Schema, 38–39
 - XMLDocumentHandler interface, 167–168
 - Xerces2, non-validating configurations, 181–185
 - XML Digital Signature, 540–547
 - verifying with XML Security Suite for Java, 548–550
 - XML documents
 - whitespace, 78
 - whitespace, removing ignorable, 81–82
 - whitespace, with `xml:space` attribute, 80–82
 - XML parsers/processors
 - DTDs, modified music collection, 145
 - entities, caching in memory, 148–150
 - entities, resolution, 147–148
 - namespaces, prefixes, 146
 - XML Schema
 - datatype and facet usage, 288–289
 - facets, 272–275
 - mimicking DTDs, 261–268
 - namespaces, 277–279
 - XPath
 - Keyword index file, 228–230
 - mapping between XML documents and RDBMSs, 382–398
 - simple program, 195–196
 - SmartDoc example showing keywords, 226–227
 - XML document for XPathTest, 197
 - XSLT processors
 - I/O, calling XSLT for I/O, 212–213
 - I/O, using DOM for I/O, 214–215
 - XSLT stylesheets
 - complex example, 209–210
 - improved example, 205–206
 - overview of structure, 203
 - XSLT translations
 - XML to XHTML, 200–202
 - XML to XHTML, output, 210–211
 - literal strings, XSLT templates, 204–205, 211
 - LiveLink's OpenText, 354, 355
 - location steps, XPath, 188–189
 - functions, 192–193
 - login modules, access controls, 555–556
 - loosely coupled integrated applications, 14
 - lossy interfaces, 166
- M**
- mapping methods
 - basics, 363
 - examples, 367–382
 - from XML documents to tables (RDBMSs)
 - basics, 357–360
 - datatype mapping, 361
 - datatype modeling, 361–362
 - defining primary keys, 360–361
 - designing tables, 360
 - representation
 - element *versus* attribute, 367
 - nested and flat, 364–367
 - to application data structures and XML documents
 - application-specific mapping, 243–246
 - isomorphic mapping, 236, 237–243
 - versus* XPath method, 362–363
 - marshaling XML documents, 236
 - memory
 - DOM trees/nodes
 - print memory for parsing, 122–123
 - speed, print parsing for SAX and DOM, 125–127
 - validating, 70–73
 - versus* SAX, 120–124
 - entity caching, 148–150
 - Message Driven Bean (MDB), EJB
 - entity beans, 403
 - metadata, 15–16
 - Microsoft
 - COM (Component Object Model), 14
 - .NET Web services architecture, 465
 - MIME specification, HTTP Content-Type headers, 314
 - MQSeries (IBM), 410
 - Multi-Schema Validator, 292
 - MVC (Model-View-Controller)
 - Model, XSLT, 200
- N**
- NameNodeMap DOM interface, 48
 - namespaces, 19–20
 - DOM trees, 67–70
 - adding declarations, 102–108
 - automatic no namespace declarations, 101–102
 - basics, 100–101
 - prefixes, 20
 - RELAX NG, 289–290
 - SAX, 113–114, 119
 - SAX parsing, 51
 - URLs, 20
 - validating with DTDs, 144–146
 - XML documents, 19–20
 - parsing with Xerces, 35–38
 - XML Schema, 277–279
 - XPath, 193–195
 - .NET (Microsoft) Web services, 465
 - node DOM interface, 47
 - basis for all nodes, 85
 - node-set type, XPath, 191
 - location step functions, 192
 - object handling, 198–199
 - NodeList DOM interface, 48
 - nodes, DOM trees, 45
 - child nodes, 89
 - creating/appending, 62–67
 - deleting, 99–100
 - manipulating, 92–94
 - processing order, 90–91
 - reverse order, 91
 - creating new nodes, 91–92
 - moving nodes between documents, 100
 - namespaces, 67–70
 - parent nodes, 88
 - previous siblings, 89
 - serializing, 75–78
 - serializing, defined, 60
 - serializing, with XMLSerializer package, 74–75
 - shallow/deep node copying, 92
 - status, accessing/updating, 86–87
 - structure, 87–88
 - traversing, 97–98

validating generation (alternate technique), 70–74
 Non-deferred DOM feature, 123–124, 127, 128
 non-validating configurations, Xerces2, 181–185
 non-validating XML processors. *See* Well-formed XML documents
 Notation DOM interface, 47
 notation, XPath, 189–191
 Notes, W3C, 17
 number type, XPath, 191
 location step functions, 192
 object handling, 198–199
 numeric datatypes, 270–272

O

OASIS (Organization for Advancement of Structured Information Standards)
 alternatives to XML Schema, 260
 RELAX NG schema origin, 281–282
 object handling, XPath, 198–199
 Object Management Group. *See* OMG
 Object Services, defined by OMG, 408
 Object-Oriented Database (OODB), 355
 objects, XPath, 191–193
 ObjectStore, 355
 OMG (Object Management Group)
 DOM language bindings, 46
 formation to standardize RPCs, 408
 OODB (Object-Oriented Database), 355
 OpenText (LiveLink), 354, 355
 Oracle, 355
 Organization for Advancement of Structured Information Standards. *See* OASIS
 OSD metadata format, 16

P

parent nodes, DOM trees, 88
 parsing XML documents. *See* XML parsers/processors
 payloads. *See* body of envelopes
 PDAs (Personal Digital Assistants), XSLT usage, 199
 PDFs (Portable Document Formats), XSLT, 200
 PKI (public-key infrastructure)
 digital certificates, 528–530

prefixes, namespaces, 20
 primary keys, RDBMSs, 360–361
 primitive types, XPath, 191
 location step functions, 192
 object handling, 198–199
 private keys, authentication, 528, 531–533
 processing rules
 DOM tree nodes, 90–91
 SOAP, 421
 XSLT, 207–208
 ProcessingInstruction DOM interface, 47
 processors. *See* XML processors
 programmatic access controls, 550, 551, 553–554
 programming languages, interworking with schema languages, 601–602
 Proposed Recommendations, W3C, 17
 provider architecture, security, 533
 public keys, authentication, 528, 531–533
 public-key infrastructure. *See* PKI
 publish/find/bind operations. *See* Web services

Q

query languages. *See* Application-specific query languages; SQL; XPath; XQuery

R

RBAC (role-based access controls), 552
 RCPs (Remote Procedure Calls), JAX-RPC, 459
 RDBMSs (relational database management systems)
 database access
 with EJB, 401–406
 with Servlet, 398–400
 interworking with schema languages, 602
 lack of Ja2EE support, 298
 mapping from XML documents to tables
 basics, 357–360, 363
 datatype mapping, 361
 datatype modeling, 361–362
 defining primary keys, 360–361
 designing tables, 360
 examples, 367–382

representation
 element *versus* attribute, 367
 nested and flat, 364–367
 retrieving stored data, 355–357
 sorting data, 354–355
 three-tier applications, 353
 RDF metadata format, 16
 RDF Schema, 606–607
 Recommendations, W3C
 document levels, 17
 versus international standards, 17
 relational database management systems. *See* RDBMSs (relational database management systems)
 relative location steps, XPath, 189
 RELAX Core, origin of RELAX NG, 282
 RELAX Namespace, 608–609
 RELAX NG schema, 604–605
 alternatives to XML Schema, 260
 co-occurrence constraints, 290–292
 mimicking DTDs
 comments, 285–286
 constructs, 286–287
 declarations, attribute-list, 284–285
 declarations, element type, 282–284
 namespaces, 289–290
 origin, 281–282
 resources, 292–293
 validators, 292
 Relaxer, data binding, 567–573
 Remote Method Invocation. *See* RMI
 Remote Procedure Calls. *See* RPCs
 resources. *See* Appendix B
 RELAX NG schema, 292–293
 XML and Java
 free downloads, 1
 in-depth/latest information, 2
 XML Schema, 281
 result trees, XSLT, 200
 RMI (Remote Method Invocation), 411
 J2EE component, 402
 role-based access controls (RBAC), 552
 RPCs (Remote Procedure Calls)
 emergence as technology, 408
 loosely *versus* tightly coupled applications, 411–412, 426
 SOAP, 422
 XML relationship, 411–412

S

- SAX (Simple API for XML)
 - accessing XML documents, 21
 - ContentHandler interface
 - problems with characters()
 - methods, 109–113
 - versus* other interfaces, 166
 - conversions
 - DOM trees to SAX events, 128–134
 - SAX events to DOM trees, 134–141
 - event-driven API, 48–55
 - events, XSLT translations to other SAX events, 216–223
 - alternate methods, 223–224
 - filters, 114–117
 - writing, 117–119
 - interfaces changes between versions 1 and 2, 120
 - namespaces, 113–114
 - new features, 119–120
 - pros and cons
 - development efficiency, combining XPath and DOM, 226–231
 - execution efficiency, 225
 - XML document conversion, comparison of SAX/DOM and XSLT, 231–233
 - versus* DOM, 44, 55
 - memory, 120–124
 - speed, 125–128
 - W3C history, 44
 - SAX2 (Simple API for XML, version 2). *See* SAX
 - SAXException class, 27
 - schema languages
 - as data models, 592–601
 - as syntactic constraints, 586–591
 - general purpose
 - DTD, 603
 - RELAX NG, 604–605
 - Schematron, 605–606
 - XML Schema, 603–604
 - interworking
 - with programming languages, 601–602
 - with RDBMSs, 602
 - special purpose
 - RDF Schema, 606–607
 - RELAX Namespace, 608–609
 - Schematron, 605–606
 - SDK (Java 2 Software Development Kit)
 - downloading from official Web site, 2
 - platforms, 2
 - testing all programs in text, 22
 - Secure Sockets Layer. *See* SSL
 - Secure Sockets Layer/Transport Layer Security. *See* SSL/TLS
 - security. *See also* Access controls; EJB; SSL/TLS; Web services; XML Digital Signature
 - B2B (business-to-business) major concern, 521
 - policies/countermeasures, 523
 - requirements, 522
 - access control policies, 524
 - communication, 523–524
 - Security Assertion Markup Language, 560
 - Security Services, defined by OMG, 408
 - serializing DOM trees, 75–78
 - defined, 60
 - versus* println() method, 76–77
 - XMLSerializer package, 74–75
 - server architecture, access controls, 554–556
 - Server-Side Includes. *See* SSI
 - Service-Enabled EAR, 517
 - Servlet
 - basics, 298–299
 - combining with JSP, 337–343
 - containers, 302–303
 - Cross-site scripting, 307–310
 - deploying, 306–307
 - emerging technologies, 296, 297
 - J2EE component, 402
 - lifecycle, 306
 - programming language-centric model, 337
 - RDBMS access, 398–400
 - returning XML documents, 299–305
 - state management, 328
 - instances, 329–330
 - patterns, 328–329
 - XML documents
 - client communicating with Servlet, 324–327
 - processing shopping cart, 320–324
 - receiving, 310–313
 - receiving requests from clients, 313–320
 - servlet engines. *See* Tomcat Java servlet engine
 - Set-Cookie headers, 320–321
 - setValidating() method, 42
 - seven-layer OSI model, XML messaging, 409
 - SGML (Standard Generalized Markup Language)
 - native databases, 354
 - XML development, 16–17
 - shallow copies, nodes, 92
 - Simple API for XML. *See* SAX
 - Simple Mail Transfer Protocol. *See* SMTP
 - Simple Object Access Protocol. *See* SOAP
 - SMTP (Simple Mail Transfer Protocol)
 - XML messaging, 409, 410
 - SOAP (Simple Object Access Protocol), 1
 - access controls, 558
 - current status, 467
 - data binding, 582–584
 - DMZs, 421–422
 - emerging technologies, 296
 - encoding, 422–425
 - engines
 - Apache Axis, 458–489
 - Apache SOAP, 453–458
 - example, application for travel reservations, 427–434
 - example, header processing, 448–453
 - example, transport layers, abstracting to support JMS, 434–447
 - example, transport layers, intermediary support, 447–448
 - examples, 413–418
 - header, 421–422
 - intermediary, 420–422
 - JAX-RPC, 459–460
 - JAXM (Java API for XML Messaging), 459–460
 - message paths, 420–422
 - origin, 412–413
 - processing rules, 421
 - pros and cons for using, 426
 - XML Digital Signature and SSL/TLS usage, 558
 - sockets
 - basics, 151
 - problems, 152
 - partial/inadequate solutions, 152–153
 - streams within streams, 153–155

- wrapping output streams, classes, 155
 - Software AG's Tamino database server, 354
 - sorting RDBMSs, XML documents, 354–355
 - spoofing, defined, 151
 - SQL (Structured Query Language), 356–357
 - SSI (Server-Side Includes), 297
 - SSL (Secure Sockets Layer), SOAP functions, 458
 - SSL/TLS (Secure Sockets Layer/Transport Layer Security) authentication
 - clients, 526–527
 - servers, 525–526
 - client configuration, 531
 - cryptography architecture
 - JCA, 533
 - JCE, 533
 - JSSE, 533–536
 - firewalls, 536–537
 - server configuration, 530–531
 - SOAP, 558
 - Standard Generalized Markup Language. *See* SGML
 - standards *versus* recommendations, 17
 - static Web applications, transition from static to dynamic contents, 6–7
 - string datatypes, 270–272
 - string type, XPath, 191
 - location step functions, 192
 - object handling, 198–199
 - Structured Query Language. *See* SQL
 - stylesheets, XSLT, 201–211, 208–211
 - processing steps, 208
 - Sun Microsystems
 - iPlanet, 295, 296
 - Sun One Web services, 465
 - superdocuments, 153
 - symbol table, Xerces2, 179
 - syntactic constraints, schema languages, 586–591
- T**
- tables. *See also* RDBMSs
 - mapping from XML documents, 246–247
 - hash tables, 247–251
 - Tamino (Software AG) database server, 354
 - technology mode, UDDI, 492, 494–495
 - templates, XSLT, 200
 - literal strings and instructions, 204–205, 211
 - Text DOM interface, 48
 - three-tier applications, 7
 - RDBMSs, 353
 - with EJB, 401–406
 - with Servlet, 398–400
 - XML messaging, 409
 - Tomcat Java servlet engine, 320
 - CD-ROM contents, 2
 - Transaction Services, defined by OMG, 408
 - Transport Layer Security. *See* SSL/TLS
 - tree-based API, DOM, 45–46
 - TreeWalker visualization tool, 78
 - TREX, origin of RELAX NG, 282
 - types, XPath, 191. *See also* datatypes
 - location step functions, 192–193
- U**
- UDDI (Universal Description, Discovery, and Integration)
 - applying to dynamic e-business, 508–516
 - basics, 492–495
 - current status, 467
 - UDDI registry, 491
 - registering WSDL, 506–508
 - UDDI4J programming, 495–506
 - Unicode character set, 13
 - supporting non-Unicode encodings, 83
 - Uniform Resource Identifiers. *See* URIs
 - Universal Description, Discovery, and Integration. *See* UDDI
 - unmarshaling XML documents, 236, 238–240
 - URI datatypes, 270–272
 - URIs (Uniform Resource Identifiers)
 - embedding XPath, 188
 - namespaces, 20
 - US-ASCII character encoding, 83
 - UTF-32 encodings, 84
 - UTF-8 and 16 encodings, 77, 83–84
- V**
- validation, XML documents, 18–19, 22
 - parsing with Xerces, 29–34
 - versus* parsing well-formed documents, 34–35
 - validating generation, DOM trees, 60
 - alternate technique, 70–74
 - validity constraints, 80
 - with specific grammars, 150–151
 - VBRELAXNG, 292
 - VCs (validity constraints), 18
- W**
- W3C (World Wide Web Consortium)
 - DOM history, 44
 - recommendations
 - document levels, 17
 - versus* international standards, 17
 - XML Schema, 260, 277
 - XPath, 187
 - XSLT, 200
 - SAX history, 44
 - SOAP potential as standard, 413
 - XML Digital Signature, 538
 - XML, Web site with official/latest documents, 2, 17
 - Web Application Server, 1
 - Web applications
 - defined, 6
 - distributed applications to decentralized to applications, 13–14
 - HTTP protocol, 6
 - interoperability, 11–13
 - retrieving information with HTTP and HTML, 7–9
 - static to dynamic contents, 6–7
 - transition from B2C to B2B, 7–9
 - XML documents, 10–11
 - Web services, 15
 - Web containers, 402
 - Web messaging. *See also* XML messaging
 - Web publishing
 - XML use, 16–17
 - Web services, 1. *See also* EAI; JAX-RPC; SOAP; UDDI; UDDI4J; WSDL; WSDL4J
 - basics, 463–466
 - defined, 409
 - dynamic integration, 15
 - EAI, 517–519
 - emerging technologies, 296
 - security, 557–558
 - specifications being proposed, 559–560
 - Web Services Description Language. *See* WSDL
 - Web Services Toolkit. *See* WSTK
 - WebLogic (BEA), 295, 296
 - WebSphere (IBM), 295, 296

- well-formed XML documents,
 - 18–19, 22
 - parsing with Xerces, 25–29
 - versus* valid documents, 25–29
 - whitespace, XML documents, 78–83
 - Working Drafts, W3C, 17
 - World Wide Web Consortium. *See* W3C
 - WSDL (Web Services Description Language)
 - applying to dynamic e-business, 508–516
 - basics, 467–474
 - current status, 467
 - IDL, 474–478
 - JAX-RPC, 490–491
 - registering WSDL, 506–508
 - WSDL4J, 481–490
 - WSTK, 478–481
 - WSDL4J (WSDL for Java), 481–490
 - WSTK (Web Services Toolkit), 478–481
- X**
- Xalan XSLT processor, XSLT programming in Java, 211
 - XDR (XML Data Reduced), SOAP basis, 412
 - Xerces Native Interface. *See* XNI
 - Xerces XML processor
 - Apache Xerces Web site
 - downloading latest versions, 22
 - licensing, Xerces processor, 22
 - CD-ROM contents, 2
 - DOM interface, 157–165
 - extended options, 155–157
 - extended properties, 157
 - licensing information, 22
 - platforms/hardware, 23
 - setup, 22–24
 - support for DOM tree traversal, 98
 - versions, 23
 - XML document parsing
 - DOM (Document Object Model), 45–48
 - JAXP, 40–43
 - JAXP *versus* Xerces native API, 43
 - namespaces, 35–38
 - SAX, 48–55
 - SAX, ContentHandler interface methods, 51
 - SAX, *versus* DOM, 55
 - valid, 29–34
 - valid *versus* well-formed, 34–35
 - well-formed, 25–29
 - XML Schema, 38–39
 - XML Schema *versus* DTDs, 40
 - XMLComponent interface, 169
 - XMLComponentManager interface, 169–170
 - XMLParserConfiguration interface, 171–177
 - basics, 170–171
 - configuration responsibilities, 177–178
 - XNI, 165
 - basics, 166–168
 - DTDs, 169
 - hierarchy, 167
 - Xerces2 XML parser/processor
 - basis in XNI, 165
 - non-validating configurations, 181–185
 - parser configurations, 179–180
 - standard components, 178–179
 - dependencies, 179
 - DTD document scanners, 179
 - entity manager, 179
 - error reporter, 179
 - symbol table, 179
 - XML document scanners, 179
 - XHTML (Extensible HTML), 200–202
 - XInclude processor, 166
 - XKMS (XML Key Management Specification), 560
 - XML (Extensible Markup Language), 1
 - configuration files, 16
 - CORBA, 13
 - data format language features, 12
 - DCOM, 13
 - DTDs, 18
 - encryption, 558–559
 - internationalization, 83–84
 - Internet resources
 - free downloads, 1
 - in-depth/latest information, 2
 - metadata for searches, 15–16
 - MVC model, 200
 - Recommendation 1.0
 - namespaces, 19–20
 - Unicode character set, 13
 - validity constraint definitions, 18
 - validity constraints, 80
 - whitespace, 78
 - RPC relationship, 411–412
 - SGML, 16–17
 - Unicode character set, 13
 - validity, 18–19
 - W3C
 - standards, recommendation document levels, 17
 - Web site with official/latest documents, 2, 17
 - Web of Web applications, 11
 - Web publishing, 16–17
 - well-formed documents, 18–19
 - XML application server
 - background, 295
 - basics, 297–298
 - need for common Web application building framework, 296–297
 - XML Data Reduced. *See* XDR
 - XML Digital Signature
 - basics, 538
 - canonicalization, 538–542
 - SOAP, 558
 - XML Security Suite for Java
 - digital signatures, signing, 542–547
 - digital signatures, verifying, 547–550
 - XML document scanners, Xerces2, 179
 - XML documents
 - DOM trees
 - creating objects, 61–63
 - creating without reading XML documents, 60
 - creating/appending child nodes, 62–67
 - namespaces, 67–70
 - serializing, 75–78
 - serializing, with XMLSerializer package, 74–75
 - validating generation (alternate technique), 70–74
 - embedding XPath, 188
 - mapping
 - to application-specific data structures with XSLT, 243–246
 - to graph structures, 251–257
 - to hash tables, 247–251
 - to isomorphic application data structure, 236, 237–243
 - to tables, 246–247
 - mapping to tables (RDBMSs)
 - basics, 357–360, 363
 - datatype mapping, 361
 - datatype modeling, 361–362
 - defining primary keys, 360–361
 - designing tables, 360
 - examples, 367–382
 - representation
 - element *versus* attribute, 367
 - nested and flat, 364–367

- marshaling and unmarshaling, 236, 238–240
- parsing with Xerces
 - DOM tree-based interfaces, 45–48
 - JAXP, 40–44
 - JAXP *versus* Xerces native API, 44
 - namespaces, 35–38
 - SAX, 48–55
 - SAX, ContentHandler interface methods, 51
 - SAX, *versus* DOM, 55
 - valid, 29–34
 - valid *versus* well-formed, 34–35
 - well-formed, 25–29
 - XML Schema, 38–39
 - XML Schema *versus* DTDs, 40
- RDBMSs
 - retrieving stored data, 355–357
 - sorting data, 354–355
- transition if Web applications
 - from B2C to B2B, 10–11
- validity, 18–19
- well-formed, 18–19
- whitespace, 78–83
- XML Key Management Specification (XKMS), 560
- XML messaging. *See also* Web messaging
 - basics, 409–410
 - document-centric, 411–412
 - envelopes, body and header, 410–411
 - role in Web services, 463
 - SOAP, 1
 - DMZs, 421–422
 - emerging technologies, 296
 - encoding, 422–425
 - envelopes/body and header, 418–420
 - examples, 413–418
 - header, 421–422
 - intermediary, 420–422
 - JAX-RPC, 459–460
 - JAXM, 459–460
 - message paths, 420–422
 - origin, 412–413
 - processing rules, 421
 - pros and cons for using, 426
 - SOAP engines
 - Apache Axis, 458–489
 - Apache SOAP, 453–458
 - example, application for travel reservations, 427–434
 - example, header processing, 448–453
 - example, transport layers, abstracting to support JMS, 434–447
 - example, transport layers, intermediary support, 447–448
- XML parsers/processors. *See also* Specific parsers/processors
 - entity caching in memory, 148–150
 - entity resolution, 146–147
 - namespace validation with DTDs, 144–146
 - parsing/generating XML documents, 21–22
 - programming hints, 143
 - validation with specific grammars, 150–151
- XML Schema
 - advanced features, 280–281
 - alternative schemas, 260–261
 - co-occurrence constraints, lack of support, 290
 - datatypes
 - basics, 270–272
 - facets, 272–276
 - using, 288–289
 - general purpose schema language, 603–604
 - mimicking DTDs
 - all constructs, 268–270
 - attribute-list declarations of DTDs, 264–266
 - comments, 266–268
 - element type declarations of DTDs, 261–264
 - namespaces, 277–279
 - resources, 281
 - versus* DTDs, 259–260
 - XML document parsing with Xerces, 38–39
 - XML Schema *versus* DTDs, 40
- XMLComponent interface, 169
- XMLComponentManager interface, 169–170
- XMLDocumentHandler interface, 167–168
 - versus* other interfaces, 166
- XMLFilter SAX interface, filters
 - using, 114–117
 - writing, 117–119
- XMLParserConfiguration interface, 171–177
- XMLReader interface, 50–51
- XMLSerializer package, 74–75
- XMLSerializer SAX interface, 114
- XNI (Xerces Native Interface), 165
 - basics, 166–168
 - basis for Xerces2, 165
 - DTDs, 169
 - hierarchy, 167
- XPath
 - basics, 187–188
 - examples, 195–197
 - expressions, Tamino, 354–355
 - mapping between XML documents and RDBMSs, 362, 382–398
 - namespaces, 193–195
 - objects and types, 191–193
 - handling, 198–199
 - pros and cons
 - development efficiency, combining XPath and DOM, 226–231
 - execution efficiency, 225
 - XML document conversion, 231–233
 - retrieving stored data, 356
 - syntax, 188–191
- XPointer, 188
- XQuery, 356
- XSLT (Extensible Stylesheet Language Transformations)
 - basics, 199–200
 - mapping to application-specific data structures and XML documents, 244–246
 - processing steps, 207–208
 - programming in Java
 - basics, 211–212
 - DOM, 213–215
 - JAXP, 212–213
 - SAX events, translating to other SAX events, 216–223
 - SAX events, translating to other SAX events, JAXP, 223–224
 - pros and cons
 - development efficiency, combining XPath and DOM, 226–231
 - execution efficiency, 225
 - XML document conversion, comparison of SAX/DOM and XSLT, 231–233
 - stylesheets, 201–211
 - syntax and semantics, 200
 - templates, literal strings and instructions, 204–205, 211
 - W3C recommendation, 200
 - XML documents, transversing with XSLT processors, 206
 - XPath use, 188

