

Index

- ABC, vii
- abs(), 191
- abs operator, 45
- __abs__ method, 123
- absolute positioning, 344
- abstract classes, 242
- AbstractAction, 459, 460
- AbstractButton, 307, 377
- AbstractDataModel class, 421–422
- abstractions, 3
- Abstract Windows Toolkit. *See* AWT
- ActionEvent, 376
- actionPerformed(), 321, 459
- ActiveX, 4
- add(), 295, 449
- __add__ method, 122, 123, 127, 454
- addActionListener, 227
- addAddress(), 434–435
- __addAddress_Clicked(), 334
- AddAddressDialog, 330, 332–333
- addComponents, 365, 367
- addListDataListener(), 427
- addListSelectionListener(), 325
- addNode(), 453–454
- address book application
 - adding addresses in, 334
 - adding dialog to, 330–333
 - adding event handlers to, 321–322
 - adding main window for, 323–329
 - adding menus to, 342–343
 - adding table model to, 424–443
 - adding text fields to, 320–321
 - adding toolbar to, 333
 - code for, 168–172, 370–371
 - converting into applet, 527–532
 - database support in, 496–512
 - DDL support in, 501–502
 - files in, 158–172
 - input form for, 320–323
 - integrating with database, 506–511
 - layout of, 362–371
 - modularizing, 496–498
 - properties in, 498–501
 - removing addresses from, 335–336
 - running in applet form, 531–532
 - streamlining using *pickle*, 178–180
 - testing, 166–168
 - using, 511
- address class, database-aware, 508–511
- address table
 - creating, 503–504
 - inserting records into, 504–505
 - reading from and writing to, 506
- AddressForm, 322–323, 325–326
- AddressFormPane, 330
- AddressMain, 425
 - accommodating to AddressModel, 440–443
- AddressModel, 425–426
 - code for, 434–440
 - helper methods in, 426–433
 - inserting into AddressMain, 434–435
 - testing, 433–434
- addSeparator(), 339
- addTableModelListener(), 420, 421, 423, 428
- addTreeModelListener(), 446, 455
- adjusting text, 211–212
- AdjustmentEvent, 376
- aggregation, 110
- alignment, FlowLayout and, 345–348
- ALTER TABLE statement, 473–474
- alternation, 578–579
- and operator, 47

606 Index

- `__and__` method, 123
- `append()`, 34, 127
- `applet` tag, 520–521
- `AppletBrowser`, 522–523, 532–533, 534–536
- `AppletContext`, 533
- applets
 - compiling, 236, 518–519
 - as container, 515–516
 - converting application into, 527–532
 - customization of, 521–522
 - described, 513–514
 - embedding in HTML page, 519–520
 - lifecycle management for, 516–518
 - reading, 524–527
 - stub of, 522
 - using, 522–524
 - working with, 514–515
- `AppletStub`, 533
- `AppletViewer`, 521
- `apply()`, 201
- `archive` tag, 519
- arguments
 - defining, 92–93
 - keyword, 92
 - passing of, 11–12
 - positional, 92–93
 - variable number of, 95–98
- arithmetic operators, 44–45
 - use with sequences, 54
- `ArithmeticError` class, 147
- `array()`, 221, 223, 253
- array Java type, 215, 216, 217
 - and `jarray`, 220–222
 - and methods, 222–224
- assembly language, 3
- assignments, 28
- `atof()`, 207
- `atoi()`, 207
- `atol()`, 207
- `AttributeError` class, 147–148
- attributes, of object, 111, 112
 - getting, setting, and deleting, 117–119
- `autoexec.bat`, 264
- `available()`, 250
- AWT (Abstract Windows Toolkit), 227, 279
 - advantages of, 280
 - design patterns in, 280
 - translating to JFC, 424
- background, setting, 297
- backreferences, 577–578
- base class, 232
- basic types, 215
- batch files, 519
- `BeanShell`, 550
- behavior, of object, 112–113
- `BIGINT` data type, 478
- `BINARY` data type, 478
- binary-oriented classes, 242
- binary streams, 250–259
- `Bistro`, 550
- `BIT` data type, 478
- bitwise operators, 51, 53
- boolean Java type, 215, 216, 217
- Boolean values, 46, 94
- `Boolean` wrapper, 217
- `BorderLayout`, 282, 303, 320, 356–361
 - advantages of, 355
- `bounds()`, 294
- `BoxLayout`, 349–351
- `break` statement, 69–70
- browsers, 513
 - `applet` tag and, 520–521
 - cautions about, 524
 - simulation of, 517
- buffer, 223
- `BufferedInputStream`, 256
 - using, 257–259
- `BufferedOutputStream`, 256–257
- `BufferedReader`, 247, 248, 249
- `BufferedWriter`, 247, 248
- buffering classes, 248–249
- buffering, of streams, 242
- built-in functions, 76
 - advantages of, 182
 - listed, 183–205
- built-in objects, 220
- `Button`, 214, 302–303
- buttons
 - adding to frame, 281, 282
 - radio, 307–308
 - toggle, 307
 - working with, 301–303
- byte Java type, 215, 216, 217
- byte streams, 242
- `Byte` wrapper, 217
- `ByteArrayInputStream`, 277, 278
- `ByteArrayOutputStream`, 277
- bytecode, 4, 203
- C++, 2, 3
- caching, 247
- `__call__` method, 122
- call stack, 132
- `callable()`, 189
- callable object, 252
- canonical path, 264
- `canRead()`, 262
- `canWrite()`, 262, 263
- `capitalize()`, 207

- capwords(), 207
- cardinality, 111
- case change functions, 207–208
- catch statement, 146
- catchall exception handler, 134–136
- cells, working with, 430–431
- center(), 212
- chaining, of streams, 242
- __changeDictKey(), 427, 432
- __changeList(), 427, 432
- CHAR data type, 478
- char Java type, 215, 216, 217
- Char wrapper, 217
- character class, 576–577
- character streams, 242
- CharArrayReader, 278
- CharArrayWriter, 278
- checkboxes
 - adding to frame, 282
 - working with, 305–307
- children(), 446, 448
- chr(), 183, 255
- class, 98
- class-based exceptions, 130
 - user-defined, 142, 147
- class hierarchy, 119–121, 242
- classes, 10
 - abstract, 242
 - acting like dictionaries, 125–127
 - acting like event sources, 230
 - acting like numbers, 123
 - base, 232
 - compiling, in Java, 236–239
 - defined, 98
 - encapsulation of, 104–105
 - exploring, 99–100
 - functions for, 198–200
 - helper, 387
 - instances of, 100–104
 - I/O, 242–243
 - methods used on, 116–117
 - nesting of, 85
 - OOP view of, 111
 - in Python, 19
 - in Python vs. Java, 555–557
 - subclassing, 231–234
- ClassInstance Java type, 217
- clauses, 66
- clear method, 125
- close(), 243, 250, 251
- cmp(), 190, 195, 196–197
- __cmp__ method, 116, 117, 167–168
- code base, 525
- code blocks, 11, 85
- code reuse, 119
- coerce(), 184–185
- cohesion, 110
- collections, 13–15
 - functions for, 196–197
 - types of, 33–38
- color_changed(), 393
- colors, working with, 297–298
- columns, working with, 430
- commentChar(), 270, 272
- comments, 24
 - importance of, 166
 - overuse of, 327
 - proper use of, 427
- commit(), 331
- comparing, 116–117
- comparison operators, 46, 48–49, 66
- compile(), 203–204, 579
- compilers, defined, 2
- Component, 377
- ComponentEvent, 376
- components
 - adding to containers, 363–366
 - alignment of, 345–348
 - in Java, 280, 296
- componentShown, 380
- compound statement, 68
- concatenation, 7–8, 54
- concrete, 244
- conditional operators, 50–51
- CONSTRAINT statement, 474, 476
- constraints
 - defining, 362–363
 - in GridBagLayout, 361–362
- constructors, in Java, 219–220
 - and subclassing, 232
- Container, 377
- ContainerEvent, 376
- containers, 280
 - EventFrame and, 381–382
- containment, 110
- content pane, 214
- contentPane, 321
- context, of applet, 522
- continue statement, 70–71
- control flow, 15
 - conditional execution, 64–73
 - example of, 75–82
 - sequential execution, 73–75
- conversion functions, 183–187
 - for strings, 206–207
- copy method, 125
- CORBA (Common Object Request Broker Architecture), 4
- CORBABeans, 4
- COS naming service, 523

608 Index

- count(), 35, 69, 78, 127, 209, 471
- coupling, 104, 110, 231
- CPython, 5
 - library for, 546–547
 - types in, 218
- CREATE INDEX statement, 473
- CREATE TABLE statement, 473, 501
- createConnection(), 502–503
- CreateFrame(), 287
- createShape(), 412
- createStatement(), 503, 508
- createTable(), 501–502, 503
- current directory, 265
- Cursor, 281

- Data Definition Language. *See* DDL
- data models, 312
- data structures, organization of, 159–160
- data types, SQL, 477–483
- database-aware dictionary, 507–508
- databases, 463–464
 - accessing data in, 470–472
 - changing data in, 472
 - inserting data in, 468–470
 - removing data from, 472
 - sample JDBC session, 464–467
 - URLs for, 467
 - using, 467–472
- DataInput, 259
- DataInputStream, 260
 - using, 260–261
- Data Manipulation Language. *See* DML
- DataOutputStream, 260
 - using, 261
- DATE data type, 478
- DateOutput, 259
- DDL (Data Definition Language), 463
 - field creation in, 478–483
 - statements in, 473–475
 - supporting, 501–502
 - using, 475–477
- debug utility, 253–256
- debugging, 386
- decimal system, 206
- DECIMAL data type, 478
- declaration, of variable, 29
 - implicit, 7
 - in Java, 30
 - in Visual Basic, 29
- Decorator design pattern (Gamma), 250, 280, 318, 419
- del, 192
- del(), 191, 193
- __del__ method, 104, 113, 114
- delattr(), 191–192
- __delattr__ method, 118

- delete(), 262
- DELETE DML statement, 486–487
 - using, 492–493, 494, 495
 - __delitem__ method, 125
- Delphi, exception handling in, 145
 - __delslice__ method, 127
- design patterns, 280, 419
- Design Patterns* (Gamma), 119, 231, 250, 344
- destructor, 104, 113
- dialog, adding, 330–333
 - __dict__ method, 118
- dictionaries
 - database-aware, 507–508
 - function of, 37–38
 - nested, 80
 - in Python, 13
 - String %, 60–62
 - using, 38
- dir(), 105, 154, 187
- directory, current, 265
- dirty flags, 507, 508–509
- dispose(), 291, 373
- distributed object model, 4
- __div__ method, 123
- divmod(), 190
- __divmod__ method, 123
- divmod operator, 45
- DML (Data Manipulation Language), 483
 - statements in, 483–487
 - using, 487–493
- document strings, 24–26, 166
- DOTALL, 584
- DOUBLE data type, 478
- double Java type, 215, 216, 217
- Double wrapper, 217
- draw_filled(), 384, 402
- draw_outline(), 384, 402
- drawing program
 - code for, 395–400
 - DrawShapes* module, 386–393, 395–414
 - Shapes* module, 383–386, 393–395
- drawOval(), 385
- drawRect(), 385
- drawRubberShape(), 389
- drawShape(), 412
- __drawShape(), 389–390
- DrawShapes class, 387, 391–393
 - event handler for, 393
- DrawShapes* module
 - adding text capability to, 400–409
 - classes defined by, 386–393
 - code for, 395–400
 - evaluation of, 409–411
- drawString(), 373
- DROP statement, 475
- dropdown list, 309

- dump(), 175
- dumps(), 177
- dynamic typing, 551
- elif, 16
 - clause, 67–68
- else, 16
 - clause, 66, 71–73
 - in exception handling, 136–137
- encapsulation, 110, 412
 - using private variables, 104–105
- end(), 589–590
- endpos property, 590
- Enough Rope to Shoot Yourself in the Foot*, 327
- EOFError class, 148
- eoISignificant(), 270
- equality, vs. identity, 140, 167
- equals(), 262
- errors
 - distinguished from exceptions, 130
 - in regular expressions, 582–583
 - sequence, 130
 - syntax, 130
 - of type Exception, 141
- escape(), 582
- escape characters, special, 577
 - listed, 594–595
- escape sequences, 40
 - using, 40–41
- eval(), 204–205
- event-driven programming, 373
- event handling, 226, 283–285
 - adding to application, 321–322
 - in different versions of Windows, 285
 - for Frame and JFrame, 286–291
 - in Java, 227–231
 - Java code for, 290–291
 - in menus, 337–341
 - in Python, 291–293
- event listener, 230
- event notification
 - in ListModel, 428
 - in tree model, 453–455
- event object, 285
- event properties, 226–227
- EventFrame, 378–380
 - container events with, 381
 - key events with, 382
 - mouse events with, 382
 - window events with, 380–381
- except, 17
 - clause, 133–134
- exception handling, 17, 129
 - catchall, 134–136
 - and interfacing with other systems, 136
 - in Java and other settings, 146
 - most-to-least specific, 144–146
 - syntax of, 132–150
- Exception class, 141, 146
- exceptions, 6
 - class-based, 130
 - classes and instances as, 140–146
 - dangers of, 130–132
 - defined, 129
 - distinguished from errors, 130
 - “garden variety,” 146–150
 - hierarchy of, 143
 - learning from, 150
 - matching handlers to, 132
 - strings to raise, 137
 - user-defined, 142, 147
- execfile(), 205
- executeQuery(), 491
- executeUpdate(), 469, 472, 491, 503
- exists(), 262, 263
- expandtabs(), 212
- expressions, 6
 - characteristics of, 27–28
 - defined, 28
 - regular, 574–603. *See also* Regular expressions
 - in spreadsheets, 27
- fields, 59
 - creation of, using DDL, 478–483
- file modes, 153–154
- file object, 153
- file path, 25
 - changing, 299
- File Java class, 261–262
 - methods in, 262
 - using, 262–266
- FileInputStream, 251
- FileOutputStream, 251, 276
- FileReader, 244, 246–247, 249
- files, 152
 - creating with FileWriter, 244–246
 - in Java, 242–243
 - length of, 245
 - methods for, 154–158
 - operations for, 153–158
 - reading, 165–166
 - reading with FileReader, 246–247
 - support for, 160–161
 - writing out, 164–166
 - writing to, 161–164
- FileWriter, 244–246
- fill_clicked(), 393
- fillOval(), 385
- fillRect(), 385
- filter(), 201
- finally, 17
 - in exception handling, 140

610 Index

- find(), 208
- finding functions, 208–210
- fireContentsChanged(), 428
- fireIntervalAdded(), 428
- fireIntervalRemoved(), 428
- fireTableCell(), 423
- first class object, 100, 113, 227
- FirstIndex, 308
- flag property, 587
- flags
 - dirty, 507
 - finding value of, 587
 - for format directives, 58
 - for regular expressions, 583, 584
- float(), 183–184
- float Java type, 215, 216, 217
- Float type, 7, 31, 216, 478
- Float wrapper, 217, 219
- Float(x) operator, 46
- floating-point numbers, 6
- FloatingPointError class, 147
- floatValue(), 216, 219
- FlowLayout, 282, 344–349
- flush(), 243, 251, 256
- FocusEvent, 376
- focusGained, 380
- font_changed(), 406
- fonts
 - finding, 298
 - implementation of, 404–406
 - working with, 298–299
- for, 15, 16–17
 - loops, 73–75
- foreground, setting, 297
- foreign keys, 474–475, 476
- format directives, 56–58
 - flags for, 58
 - use with numbers, 59
- formatSQLStr(), 505
- formatting
 - of numbers, 59–60
 - of strings, 43, 55–62
- Frame, 283
 - class hierarchy for, 293–294
 - component functionality in, 294
- frames, in Java, 280
 - adding components to, 281–282
 - setting mouse cursor of, 281
- fromAddress(), 509, 510
- fromBounds(), 412
- FTP protocol, 527
- functional decomposition, 427
- functional programming, 200–203
- functions, 10, 66–67
 - built-in, 76
 - calling, 91–93
 - defined, 2
 - defining, 91–92
 - public, 220, 221
 - in Python, 18
- garbage-collected, defined, 113
- get(), 125, 224
- getAbsolutePath(), 262
- getAddress(), 331
- getAddressAt(), 434
- getAddressAtRow(), 426
- getAddressAttributeAtColumn(), 426
- __getAddressAttributeAtColumn(), 431
- getAddressByName(), 434
- getAddresses(), 164, 171–172
- getAllowsChildren(), 446, 448
- getattr(), 193
- __getattr__ method, 118
- getBytes(), 222, 223
- getCanonicalPath(), 262
- getChars(), 222, 223
- getChild(), 446, 455
- getChildAt(), 446, 448
- getChildCount(), 446, 448, 455
- getColumnClass(), 420, 423, 428, 430
- getColumnCount(), 420, 422, 428, 430
- getColumnName(), 420, 423, 428, 430
- getElementAt(), 427
- getEventsInfo(), 227–229, 284, 308
- getGraphics, 373
- getIndex(), 446, 448
- getIndexOfChild(), 446, 455
- getInt(), 470
- __getitem__ method, 125
- getLastPathComponent(), 458
- getMean(), 77–78, 94–95, 562–563
- getMedian(), 79–80, 564
- getMode(), 78–79, 563
- getName(), 224, 262, 449
- getNodePathToRoot(), 454–455
- getOutputStream(), 528
- getParent(), 262, 446, 448
- getPath(), 262
- getRange(), 75–76, 562
 - fine-tuning, 76
- getRect(), 384, 385, 402
- getRoot(), 446, 455
- getRowCount(), 420, 422, 428
- getSize(), 427
- __getslice__(), 127
- getState(), 331
- getString(), 470
- getter/setter methods, 118
- getValueAt(), 420, 423, 428, 430, 431
- GIF images, using as icons, 300
- global namespace, 107

- global variables, 107–108, 295
- globals, 204
- globals(), 188
- granularity, 143
- Graphical User Interfaces. *See* GUIs
- graphics, 372–375
 - adding text to, 400–409
 - common events, 376–377
 - drawing program example, 383–415
 - event frame for, 378–383
- graphics objects, 372
- graphics resource, disposing of, 373
- GridBagConstraints, 361–363
- GridBagLayout
 - characteristics of, 361
 - constraints in, 361–362
 - example of use of, 362–370
 - parameters for, 362
- GridLayout, 320, 351–355
- group(), 588–589, 596
- groupdict(), 589
- groupindex property, 588
- groups(), 589, 596
- GUI implementation, 557–559
 - in Java, 560
 - in Python, 559–560
 - sample, 319–336
- GUIs
 - building, 279
 - example of building, 319–336
- handle_keyPressed(), 407
- handle_keyTyped(), 408
- handle_mousePress(), 407
- handle_mouseRelease(), 413–414
- handleEvent(), 287
- hasattr(), 193–194
- hash(), 191
- __hash__ method, 116, 117
- hash value, 185
- hashing, 116–117
- hashtable, 185
- haskey method, 125
- hasMoreElements(), 447
- helper classes, 387
- helper methods, 423, 502–503
- hex(), 183
- hexadecimal system, 183, 206
- hierarchy, class, 119–121, 242
- high-level languages, 3
- HotSpotClient, 543
- house price example, 75–81
- HTTP protocol, 527
- I/O classes, 242
- icons
 - Action and AbstractAction and, 460–461
 - working with, 299–300
- id(), 191
- IDE, 224
- identity, vs. equality, 140, 167
- identity functions, 191
- identity operator, 102, 186
- if, 15, 16, 64–66
 - using, 97
- IGNORECASE, 583, 584
- immutability, 88–89
- immutable types, 15
 - defined, 36
 - conversion of, 185
- implementation, 119
- implicit declaration, 7
- import statement, 86–90
 - forms of, 87
- in operator, 49
- index(), 127, 209
- [index] operator, 55
- IndexError class, 148
- indexes, 8
- indexing, 8
- inheritance, 111
 - forms of, 119
 - limits of, 119
 - multiple, 121
 - single, 267, single, 293
- __init__ method, 103–104, 113, 114, 116
 - purpose of, 432
- __init__toolbar, 334
- INNER JOIN, 486
- input focus, 405
- input streams
 - parsing of, 275
 - tokenizing of, 269–276
- input-output (I/O) functions, 200
- InputStream, 242, 243, 250
- insert(), 35, 127
- INSERT DML statement, 483–485
- insertAddress, 504
- Instance Java type, 217
- instances
 - of class, 100–103
 - creating and destroying, 113–144
 - defined, 25, 111
 - methods associated with, 103–104
 - string representation of, 114–116
- Instant Basic, 550
- InstantDB, 464
 - limitations of, 475, 476, 494
 - setting up, 466–467
 - SQL subsets in, 472
 - type support in, 481–482
- instantiation, 101
- int Java type, 215, 216, 217

612 Index

- int(), 46, 183, 184
- Integer type, 7, 30, 31, 32, 219, 478
- Integer wrapper, 217, 217
- Integrated Development Environments.
 - See IDE
- interactive mode, 5, 386
 - switching to, 12
- interfaces, 119, 230, 242
 - defining, 105
 - OOP, 110
- interpreters, defined, 2
- intrinsic functions, 182–205
- intrinsic operations, 76
- introspection, 86
- invalidate(), 350
- IOException class, 148
- is, 102, 140
- is operator, 48
- is not operator, 48
- isAbsolute(), 262
- isCellEditable(), 420, 423, 428, 430, 431
- isDirty(), 508, 509
- isFile(), 262
- isInfinite(), 216, 218
- isinstance(), 198, 199
- isLeaf(), 446, 448, 455
- isSubclass(), 198, 200
- ItemEvent, 376
- items(), 38
- items method, 125
- iteration, 14

- jar (Java Archive) files, 519
- jarray*, 220–222
- Java programming language
 - classes in, 555–557
 - command language in, 550
 - compared to Python, 242, 555–573
 - constructors in, 219–220
 - data types in, 478
 - default package of, 239
 - design patterns in, 231
 - documentation for, 213
 - embedding Jython in, 239, 572–573
 - event handling in, 290–291
 - exception handling in, 145
 - flexibility of, 4
 - menus in, 336
 - networking APIs in, 241
 - properties in, 224–227, 498–499
 - relationship to Python, 3, 4–5, 19–20
 - and scripting, 549, 552–553
 - streams in, 241–278
 - subclasses in, 231–234
 - types in, 215–218
- Java APIs, 214–215
- Java runtime, installation of, 537–538
- Java Virtual Machine. *See* JVM
- java.awt, compatibility with JFrame, 283
- java.io*, 242
- java.net.URL, 526
- JavaBeans, 4
 - compiling, 236
 - events in, 224, 229–231
 - introspection feature of, 231
 - properties of, 224
- JavaScript (Rhino), 550
- JButton, 301–302, 320, 376, 378
 - properties shared with Button, 302–303
 - using, 214
- JCheckBox, 305–306, 377
 - using, 306–307
- JCheckBoxMenuItem, 377
- JComboBox, 309–310, 377
- JComponent, 296
- JDBC (Java Database Connectivity), 464
 - data types in, 478, 480
 - helper methods in, 502–503
 - programming with, 467–472
 - sample session in, 464–467
- JDK (Java Development Kit), 537
 - download from Blackdown, 542–543
 - download from Sun, 543–544
 - installation of, 538–540
 - testing installation of, 545–546
- JFC (Java Foundation Classes), 279
 - advantages of, 280, 424
 - design patterns in, 280
- JFrame, 320, 378
 - class hierarchy for, 294
 - component functionality in, 294
 - design flaw of, 283
 - event handling using, 283–293
 - lineage of, 282–283
 - using, 214, 281–283
- jinfo*, 227
- JLabel, 296
- JList, 308–309, 377
 - working with, 310–312
- JList_AddressMode1.py
 - code for, 312–314
 - examination of, 315–318
- JMenuBar, 336–337
- JMenuItem, 341, 377
- JOIN ANSI SQL statement, 483
- join(), 211
- joinfields(), 211, 247
- joining strings, 211
- JPanel, 295–296
- jpeg images, using as icons, 300

- JPopupMenu, 341–342
- JPython (Jython), 4
 - embedding in Java, 572–573
 - installation on Linux, 542–547
 - installation in Windows environment, 540–541
 - library for, 546–547
 - reconnecting to session, 468–469
 - running, 547
 - and scripting, 549–551, 552–554
 - types in, 218
- JRadioButton, 307–308
- JRadioButtonMenuItem, 341
- JScrollBar, 377
- JScrollPane, 318, 419
- JTable, 312, 416–417
 - default table model, 417–418
 - getting and setting table values, 418–419
- JTextField, 303–305, 320, 377
- JToggleButton, 307
- JToolBar, 458–459
 - actions and, 459–462
- JTree, 312
 - constructor of, 444
 - data model sharing and, 451
 - event handling and, 456–457
 - example of use of, 457–458
 - function of, 444–445
 - model interface, 446–453
- JVM, 153–154, 467, 537
- jythonc, 236
- Jython, installation of, 540 *See also* JPython

- key events, 382
- key/value pairs, 37, 38
- KeyError class, 148
- KeyEvent, 376
- KeyListener, 382
- keyPressed, 382, 407–408
- keyReleased, 382
- keys
 - in dictionary, 37
 - foreign, 474–475, 476
 - primary, 474
- keys(), 38, 125
- keyTyped, 382, 408
- keyword arguments, 92
 - variable number of, 96
- keywords, 61
 - and Python built-in functions, 182

- LabelFor, 300
- labels, adding to frame, 281, 296
- LastIndex, 308
- LastModified(), 262, 263

- late-bound polymorphism, 122–127
- layout management, 281
- layout managers, 343–344
 - example of use of, 362–370
 - types of, 344–362
- leaf node, 445–446
- LEFT JOIN, 486
- len(), 55, 196, 284
- __len__ method, 125
- length(), 262
- lifecycle management, 516
- lineno(), 269
- Linux, Jython on, 542–547
- List, 308
- list(), 185, 186, 262
- listener, 285
- ListEnumeration class, 447–448
- ListModel
 - event notification in, 428
 - implementing, 427–428
 - mapping to, 426–427
- lists
 - described, 33–34
 - dropdown, 309
 - properties of, 308
 - in Python, 9–10, 13
 - as sequences, 37
 - using, 34–35
- ListSelectionEvent, 309
- ListSelectionListener, 325
- literals, 24, 39
 - identifying, 39–40
 - numeric, 41–42
- ljust(), 212
- loadDriver(), 502
- loadFromFile(), 505
- LOCALE, 584
- locals, 204
- locals(), 188
- logical operators, 46, 47, 66
 - and Boolean returns, 49–50
- long(), 183, 184
- Long(x) operator, 46
- Long Java type, 215, 216, 217
- Long type, 31
- Long wrapper, 217
- LONGBINARy data type, 478, 480
- LONGVARCHAR data type, 478, 480
- LookupError class, 147
- Lorenzen, Jaysen, 542, 574
- low-level languages, 3
- lower(), 207
- LowerCaseMode(), 269
- __lshift__ method, 123
- lstrip(), 211

614 Index

- macros, 1–2
- main block, 85, 503
- main method, 10
- main module, 423
- main window
 - adding event handlers to, 324–326
 - adding names to, 324
 - creating, 323–324, 326–329
- makeDirs(), 262
- map(), 202, 252–253, 255–256
- mark(), 243, 250, 256, 257
- markDirty(), 508–509
- markSupported(), 243, 256
- match(), 580, 597
 - syntax of, 585
- match objects, 578
 - methods for, 588–590
 - properties for, 590–591
- max(), 55, 196
- memory, using Java streams with, 277–278
- menubars, in Java, 280, 336–337
- menus
 - bar, 280, 336–337
 - event handling in, 337–341
 - popup, 341–342
- metacharacters, listed, 591–593
- methods, 10. *See also* functions
 - arrays and, 222–224
 - defining and calling, 91–93
 - getting return values from 216
 - passing arguments to, 215–216
 - subclassing with, 232–234
- Microsoft Access, 464
 - data types not supported by, 480
 - foreign keys in, 475
 - referential integrity in, 475, 494
 - SQL subsets in, 472
 - type support in, 482
- min(), 55, 196
- mkdir(), 262
- mnemonics, 300
- __mod__ method, 123
- Model View Controller (MVC), 280, 310
 - in JTree and JTable, 312
- modes, file, 153–154
- modularity
 - adding to application, 496–498
 - assessing, 498
 - importance of, 412
- modules, 10, 18
 - attributes of, 85
 - functions for, 191–195, 197–198
 - importation of, 90
 - manipulation of, 86–90
 - search path for, 89–90
- modulus, 44, 55–56
- most-to-least specific order, 144–146
- mouse cursor, setting, 281
- mouse events, 382–383
 - mouseClicked, 382
 - mouseDragged, 382
 - mouseEntered, 382
 - MouseEvent, 376
 - mouseExited, 382
 - mouseMoved, 382
 - mousePressed, 382, 388
 - mouseReleased, 382, 390
 - __mul__ method, 123
- MULTILINE, 584
- multiline statements, 27
- multiple inheritance, 121
- multiplier characters, 577
- mutable, 185
- MutableTreeNode, 445
- MVC. *See* Model View Controller

- NameError class, 149
- namespace operators, 187–189
- namespaces, 86
 - global, 107
- __neg__ method, 123
- nested dictionaries, 80
- nesting, 85
- new, 101
- newline(), 248
- newline character, 155
 - getting rid of, 156–157
- next(), 470
- nextElement(), 447
- nextToken(), 269, 271, 272
- None type, 31, 32–33
- __nonzero__ method, 116, 117
- not operator, 47
- not in operator, 49
- numbers
 - conversion of, 183–185
 - formatting of, 59–60
 - Python treatment of, 6–7
- numeric conversion operators, 46
- NUMERIC data type, 478
- numeric functions, 190–191
- numeric literals, 41–42
- numeric types, 12–13
 - ranges of, 31–32

- object instance, 175
- object wrappers, 216
 - mappings for, 217
 - tables and, 418
- Object Management Group (OMG), 4

- Object Pascal, exception handling in, 145
- object-oriented programming (OOP)
 - defined, 109–110
 - inheritance in, 119–121
 - objects and classes in, 110–118
 - polymorphism in, 121–128
 - in Python, 19
- Object-Oriented Design* (Booch), 231, 344
- ObjectOutputStream, 276, 319
- objects, 12
 - built-in, 220
 - callable, 252
 - characteristics of, 112–113
 - conversion of, 186
 - described, 110–111
 - encapsulation of, 110
 - persisting, 276–277
 - in Python, 113
- Observer/Observable design pattern, 280
- oct(), 183
- octal system, 183, 206
- ODBC (Open Database Connectivity), 464
 - driver for, 464
- okayHandler, 321
- OLE controls (OCXs), 224
- open(), 200
- openStream(), 525, 528–531
- operations
 - class and instance, 191–196, 198–200
 - file, 153–158
 - identity, 191
 - intrinsic, 76
 - module, 197–198
 - numeric, 190
 - sequence and collection, 196–197
- operators
 - arithmetic, 44–45
 - bitwise, 51
 - comparison, 46 48–49, 66
 - conditional, in Java and other languages, 50–51
 - in expressions, 27, 44
 - logical, 46 47 49–50, 66
 - numeric conversion, 46
 - precedence of, 52, 54
 - sequence, 54–55
 - shift, 51, 53
- __or__ method, 123
- or operator, 47
- ord(), 183, 184, 252
- ordinaryChar(), 270
- ordinaryChars(), 270
- organization, 93–98
 - classes, 98–105
 - code blocks and namespaces, 85–86
 - functions and methods, 91–93
 - globals, 107–108
 - importance of, 83–85
 - modules, 86–91
 - packages, 106–107
- OUTER JOIN, 486
- OutputScreen, 237, 238, 239
- OutputStream, 233, 237, 238, 242, 243, 250, 251
- Oval class, 383, 385
- oval_pressed(), 392, 393
- OverflowError class, 149
- overloading, 233

- pack(), 214, 282
- packages, 18
 - uses of, 106–107
- paint(), 373, 374–375, 384
- PaintBox, 387, 388–391, 406–409
 - fine-tuning of, 411–414
- panels, in Java, 280
 - context of, 294–295
 - creating, 295–296
- parent directory, 263
- parseNumbers(), 269
- parsing, 20–21
 - in Java, 571–572
 - in Python, 570
- parsing strings, 211
- path
 - canonical, 264
 - file, 299
- path separators, 265
- pattern property, 588
- persisted data, 154
- pickle* module, 172–174
 - example of use of, 178–180
 - file created by, 174–175
 - writing out objects with, 175–177
 - writing to strings using, 177
- point_changed(), 406
- polylines, 372
- polymorphism, 121, 242
 - described, 303
 - late-bound, 122–127
 - power of, 126
 - typed, 122, 127–128
- pop(), 127
- popup menus, 341–342
- __pos__ method, 123
- pos property, 590
- positional arguments, 92–93
- pow(), 190
- __pow__ method, 123
- precedence, of operators, 52, 54
- prefetching, 247
- primary keys, 474

616 Index

- primary prompt, 5
- primitive types, 215
 - typecodes and, 221
- print(), 249, 250
- printEventProperty(), 229, 284–285
- printf(), 95, 96–97
- println(), 249, 250
- PrintWriter, 249–250
- private variables, 104–105
- programming, 1
 - efficient, 83–84
 - functional, 200–203
 - and organizing code, 84–85
 - scripting vs., 548–549
- properties, 498–499
 - event, 226–227
 - in Java, 225–227
 - in JavaBeans, 224
- property file, 499–501
- public functions, 220, 221
- Publish design pattern, 280
- pushBack(), 269
- PyArray, 220
- Python. *See also* Cpython; JPython
 - advantages of, 554
 - basic functions with, 6–10
 - classes in, 555–557
 - compared with Java, 242, 555–573
 - described, 3
 - data types in, 478
 - history of, vii
 - installation of, 5
 - interactive mode of, 5–6
 - as main program, 10–12
 - reasons for learning, 2–3
 - relation to Java, 4–5, 19–20
 - scripting in, 552–554
 - starting with, 5–6
 - Web server in, 524–525
- quoteChar(), 269
- __radd__ method, 122, 123–124, 127
- radio buttons, 307–308
- raise, 137–139
- RandomAccess File
 - methods in, 266–267
 - modes for, 267–268
- range(), 73–74
- raw string, 575–576
- raw_input(), 69, 200
- re property, 591
- re.compile, 602
- read(), 154, 157, 158, 162, 243, 246, 250
- readAddresses(), 164, 319, 320, 506, 511
- readBoolean(), 259
- readByte(), 259
- readBytes(), 259
- readChar(), 259
- readDouble(), 259
- Reader, 242
 - methods in, 243
- readFloat(), 259
- readFully(), 259
- readInt(), 259
- readLine(), 154, 155, 156, 162, 248, 259
- readLines(), 154, 156
- readLong(), 259
- readShort(), 259
- readUnsignedByte(), 259
- readUnsignedShort(), 259
- readUTF(), 259, 261
- ready(), 243
- REAL data type, 478
- rect_pressed(), 392, 393
- Rectangle class, 383, 385
- reduce(), 202
- referential integrity, 475, 493–494
 - achieving, 494–495
 - support for, 494
- regular expressions
 - error properties, 582–583
 - example of use of, 574–575
 - flags for, 583, 587–588
 - functions for, 579–582, 584–587
 - grouping and backreferences by, 577–578
 - match objects of, 588–591
 - pattern characteristics of, 575–579
 - using, 601–603
- reload(), 197–198
- remove(), 34, 69, 127, 295, 509, 510
- removeAddressByName(), 434, 435
- __removeAddress_Clicked(), 335
- removeComponents, 365
- removeListDataListener(), 427
- removeTableModelListener(), 420, 421, 423, 428
- removeTreeModelListener(), 446, 455
- renameTo(), 262
- replace(), 209
- replaceability, polymorphism and, 121
- reportStatistic(), 80–81, 564–565
 - using, 81–82
- repr(), 160–161, 186
- __repr__ method, 114, 115–116, 320
- reset(), 243, 250, 256, 258
- resetSyntax(), 269, 275
- return values, mappings for, 217
- reuse, of code, 119
- reverse(), 34, 35, 127
- rfind(), 208

- Rhino, 550
- RIGHT JOIN, 486
- rindex(), 209
- rjust(), 212
- root node, 445
- round(), 191
- rows, working with, 430
- __rshift__ method, 123
- rstrip(), 211
- __rsub__ method, 122
- rubberbanding, 388
- runReport(), 565
- runtime, 38

- sandbox, 514
- __save__ method, 160
- scaffolding code, 162, 386, 433
- Scheme, 550
- scripts, 2, 4
 - choosing language for, 551–552
 - execution of, 85
 - Java and, 549
 - Jython and, 549–551
 - vs. programming, 548–549
- scroll panes, 419
- search path, 89–90
- search(), 579–580, 597
 - syntax of, 583, 585
- seek(), 154, 157–158
- SELECT DML statement, 483, 485
 - using, 490–492
- self argument, 99, 103
- self-documenting, 431
- separators, 247
- sequence errors, 130
- sequence operators, 54–55
- sequences, 13–15, 33, 37
 - of characters, 596–601
 - conversion of, 185–186
 - functions for, 196–197
 - using arithmetic operators with, 54
- Serializable, 276
- servlets, compiling, 236
- set(), 224
- __setAddressAttribute(), 431–432
- __setAddressAttributeAtColumn(), 427
- setattr(), 194
- __setattr__ method, 118
- setFontName(), 408
- setFontPoint(), 409
- __setitem__ method, 125
- setName(), 449
- setParent(), 449
- setShapeType(), 412
- __setslice__ method, 127
- setStub(), 522
- setter methods, 508
- setValueAt(), 420, 423, 428, 431
- Shape class, 383
- ShapeButton class, 386
- Shapes class, 383, 385–386
- Shapes module, 383–386
 - code for, 393–395
- ShapeTool class, 410–411
- shift operators, 51, 53
- short Java type, 215, 216, 217
- Short wrapper, 217
- showParameters(), 523, 524
- SimpleModel class, 447, 452–453
- SimpleNode class, 448–451
- single inheritance, 267, 293
- Skij, 550
- skip(), 243, 250, 256, 257
- skipBytes, 259
- slashSlashComments(), 270
- slashStarComments(), 270
- slice
 - range of, 9
 - setting, 250
- slice notation, 8–9
- SMALLINT data type, 478
- Smalltalk, 550
- sort(), 34, 35, 217
- source code, 25
- span(), 590
- specialization, 119
- split(), 21, 210, 580–581
 - syntax of, 585–586
- splitfields(), 210
- splitting strings, 210
- spreadsheets, expressions in, 27
- SQL (Structured Query Language), 463
 - data types in, 477–483
 - programming with, 467–472
 - subsets of in InstantDB and Microsoft Access, 472
- StandardError class, 146–147
- start(), 589–590
- statements, 64
 - compound, 68
 - defined, 2, 26
 - multiline, 27
 - subordinate, 65
 - syntax of 26–27
- static, 217
- static mode, debugging in, 386
- static typing, 551
- statistics application example, 562–565
 - in Java, 565–569
 - in Python, 565
- StatusBox class, 387
- stdev(), 471

618 Index

- str(), 183, 186
- __str__ method, 163
- Strategy design pattern, 280
- streams, 241
 - binary, 250–259
 - buffering of, 242
 - chaining of, 242
 - conceptualizing, 242
 - and memory, 277–278
 - persisting objects with, 276
 - tokenizing of, 269–276
 - working with, 250–261
- StreamTokenizer
 - methods in, 269
 - parsing Python using, 273–276
 - using, 270–273
- String % dictionary, 60–62
- String Java type, 215, 216, 217
- __string__ method, 114, 115
- string module, 247
- string property, 590
- String type, 30
- strings, 7
 - case change, 207–208
 - conversion of, 183–184, 206–207
 - document, 24–26
 - finding, 208–210
 - formatting of, 43, 55–62
 - manipulating, 210–212
 - parsing of, 20–21, 569–572
 - in Python, 13
 - to raise exceptions, 137
 - raw, 575–576
- strip(), 211
- stripping strings, 211
- Structured Query Language. *See* SQL
- stub, 522
- sub(), 581
 - syntax of, 586
- __sub__ method, 122, 123
- subclassing, 231–232
 - with Java constructors, 232
 - with Java methods, 232–234
- subn(), 581–582
 - syntax of, 586–587
- subordinate statement, 65
- Subscribe design pattern, 280
- substrings, finding, 208
- suite, 65
 - separation of, 66–68
- sum(), 471
- superclass, 228, 283
- swapcases(), 207
- Swing, 279
 - advanced topics in, 416–462
 - advantages of, 383
 - widgets in, 280
- symbols, definitions. *See* page 620
- syntax, 6
- syntax errors, 130
- sys module, 90
- table models
 - default, 417–418
 - examples of, 422–443
 - using, 420–421
- table values, getting and setting, 418–419
- TableModel, 420–421
 - implementing, 428–430
- tables
 - adding, 425
 - and object wrappers, 418
- Tcl, 550
- tell(), 154, 157–158
- terse, defined, 2–3
- testing, of program, 166
- testOutScreen, 239
- text
 - adding to graphics program, 400–409
 - Python treatment of, 7–8
- Text class, 401–404
 - code for, 403–404
 - implementation of, 406–409
 - support for, 404–406
- text fields
 - adding to application, 320–31
 - adding to frame, 282
 - in Java, 280
 - working with, 303–305
- text-oriented classes, 242
- text_pressed(), 405
- text_read_write(), 320
- TextEvent, 376
- TIME data type, 478
- TIMESTAMP data type, 478
- TINYINT data type, 478
- titlebar, 296
- toByteArray(), 278
- toggle buttons, 307
- toolbars
 - adding, 333–334
 - in Java, 280
- ToolBox class, 386
- toolTip, 296
- toString(), 262, 270, 449
- treeCollapsed, 456
- treeExpanded, 456
- TreeNode, 445
- trees, 444
 - event handling and, 456–457
 - event notification and, 453–455
 - implementation of, 446–453
 - methods for, 455
 - model of, 445–446

- treeStructureChanged(), 454
- treeWillCollapse, 456
- treeWillExpand, 456
- truth testing, 116–117
- try, 17
 - block, 144
 - in exception handling, 132–133
- try ... except, 132–133, 164, 493
- try ... finally, 132–133, 164, 493
- tuple(), 185, 186
- tuples, 13
 - formatting, 56
 - as sequences, 37
 - using, 36–37
- type(), 12–13, 189–190
 - in Jython vs. Python, 32
- type checking functions, 189–190
- type signatures, 233
- Type type, 31, 32
- typecodes, 221
- typed polymorphism, 122, 127–128
- TypeError class, 149
- types
 - assignment of, 12
 - basic, 215
 - collection, 13–15, 33–38
 - in CPython vs. JPython, 218
 - determining at runtime, 38–39
 - in Java, 215–219
 - numeric, 12–13, 31–32
 - primitive, 215, 221
 - in Python, 216–217
 - of variable, 7, 30–33
- typing, dynamic vs. static, 551

- Unicode, 215, 242
 - described, 268–269
 - reading and writing of, 261
- update(), 509, 510
- UPDATE DML statement, 486
 - using, 494–495
- update method, 125
- upper(), 207–208
- URLs, working with in interactive session, 525–527
- user-defined class-based exceptions, 142, 147
- userlist module, 127

- validate(), 350
- valueChanged, 456
- ValueError class, 150
- ValueForPathChanged(), 446, 455
- ValueIsAdjusting, 308
- valueOf(), 216, 218, 219, 222, 223
- values
 - defined, 6
 - in dictionary, 37
- values(), 38
- values method, 125
- VARBINARY data type, 478
- VARCHAR data type, 478
- variable scope, 86
- variables, 6
 - declaration of, 29–30
 - global, 107–108, 295
 - in expressions, 27
 - implicit declaration of, 7
 - in modules, 29
 - naming of, 30
 - private, 104–105
 - type of, 7
- vars(), 188–189
- VBX, 224
- VERBOSE, 584, 599
- Visual Basic, 4
 - exception handling in, 145
 - VBXs in, 224
- void, 224

- Web server, Python, 524–525
- WHERE, 484
- while, 15, 16, 69
 - loop, 160
- whitespace, 27
- whiteSpaceChars(), 270
- widgets, 280
- wildcards, 576
- Window, 377
- window events, 380–381
- WindowActivated, 286, 380
- WindowClosed, 286
- WindowClosing, 286, 380
- WindowDeactivated, 285, 286, 380
- WindowDeiconified, 286
- WindowEvent, 376, 380
- WindowEventListener, 292–293
- WindowIconified, 286, 380
- WindowListener, 287
- WindowOpened, 286, 380
- wordChars(), 270, 271
- write(), 154, 155, 156, 162, 238, 249, 243, 251, 259
 - overriding, 234, 235–236
- writeAddress(), 434, 435
- writeAddresses(), 165, 319, 320, 506, 527
- writeBoolean(), 259
- writeByte(), 259
- writeBytes(), 259, 268, 269
- writeChar(), 259
- writeChars(), 259, 268, 269
- writeDouble(), 259
- writeFloat(), 259
- writeInt(), 259
- writeLines(), 154

620 Index

- writeLong(), 259
- Writer, 242
 - methods in, 243
- writeShort(), 259
- writeUTF(), 259, 261, 268–269

- __xor__ method, 123
- xrange(), 17, 74–75

- ZeroDivisionError class, 151
- zeros(), 221, 246, 254, 258
- zfill() 212
- zip files, 519

- “ flag, 58
- # flag, 58
- 0 flag, 58
- flag, 58
- operator, 45
- / operator, 45
- [:] operator, 55
- . metacharacter, 591
- != operator, 48
- ? metacharacter, 592
- ?? metacharacter, 593
- \$ metacharacter, 592
- + flag, 58
- + metacharacter, 592
- + operator, 45
- +? metacharacter, 593
- *, to define variable number, 95
- * metacharacter, 592
- * operator, 45
- *? metacharacter, 593
- **, to declare keyword arguments, 96
- ** operator, 45

- ^ bitwise operator, 53
- ^ metacharacter, 591, 601
- < operator, 48
- << shift operator, 53
- <= operator, 48
- <> operator, 48
- == operator, 48, 140
- > operator, 48
- >= operator, 48
- >> shift operator, 53
- % operator, 45, 56
- %c format directive, 58
- %d format directive, 57, 59
- %e format directive, 57, 59
- %f format directive, 58, 59
- %g format directive, 58
- %i format directive, 57, 59
- %o format directive, 57
- %s format directive, 57
- %u format directive, 57
- %x format directive, 57
- & bitwise operator, 53
- | bitwise operator, 53
- | metacharacter, 596–601
- ~ bitwise operator, 53
- \<No.> escape character, 594
- \A escape character, 594
- \b escape character, 594
- \B escape character, 594–595
- \d escape character, 595
- \D escape character, 595
- \s escape character, 595
- \S escape character, 595
- \w escape character, 595
- \W escape character, 595
- \Z escape character, 594