Why Is This Book Necessary?

For years we have been enthusiastically pounding the table, so to speak, saying that scripting is “where it’s at!” for WebSphere® administrators. All the while, we (along with our students) have wished for a book that would enable more people to use Jython1 for their scripting needs. We finally got together and set out to write one, and we hope this book addresses not only our own desires and demands but yours as well.

Websphere’s robust and versatile scripting facility differentiates the IBM® WebSphere Application Server product from competitors in the Java™ EE server marketplace. As convenient and friendly as the Integrated Console can be, the true power for administrators is in scripting. This is probably evident to anyone who has had to configure something on multiple machines or perform the same configuration numerous times.

IBM’s support for scripting that is built into the WebSphere platform is staggering. As you delve into it, you discover all sorts of hidden capabilities and ease-of-use features and quickly develop a greater understanding of how WebSphere fits together. Were we to deliver a book containing but a single page on each aspect of the available script objects, you’d need a forklift to take it home. Instead, we’ve hopefully provided you with a book that helps prepare you for a journey of discovery. We have documented and explained the scripting concepts, the core objects, and many of our favorite techniques, while demonstrating some new ones of our own.

---

1 Jython is an implementation of the Python programming language that executes on a Java Virtual Machine.
Chapter 1 Introduction

If you will, think of WebSphere as a healthy, bio-diverse, coral reef, rich in wildlife. In this book, we teach you to master the core concepts necessary to explore the reef and introduce you to many of our favorite reef denizens. Afterward you will be prepared to explore more of the reef on your own, discovering for yourself more of the richness that IBM has built into WebSphere. Once you’ve mastered the core concepts, self-discovery becomes important; each new product layered on WebSphere (for example, WebSphere Enterprise Service Bus and WebSphere Process Server) and each new version adds more and more scripting capabilities. So in the way of the ancient parable, we will not only give you some fish, but also teach you to fish.

We hope that you enjoy the book...and the journey.

—Bob, Kevin, and Noel

About the WebSphere Application Server Product

The WebSphere Application Server is a large and complex product. As such, it is function-rich and can be configured and used in numerous ways. But you probably know that already. The kind of person likely to pick up this book and consider buying it is someone who has an understanding of what the WebSphere Application Server is and just how challenging its administration can be. It is also likely that you have either tried administering the product or might be wondering how to administer it using scripting. If you’ve gone down this road on your own, you have probably had some challenging moments (alright, hours) attempting to get your scripts to “behave.” That is what this book is all about.

This book is not for the novice WebSphere Application Server administrator. It doesn’t define or explain J2EE™ or an application server. Definitions and explanations of foundational topics are available elsewhere (see the online documentation2). This book is focused on the topic of scripting for the WebSphere Application Server environment.

In writing this book, we spent a great deal of time trying to understand this beast and what is required to administer a WebSphere Application Server using the examples that exist in the available documentation. As a WebSphere Application Server technical support analyst and team lead, and as instructors, we have also spent a great deal of time sharing our knowledge of this topic with others. That, too, is what this book is about—sharing some of what we have learned with those who are interested in learning.

This Book and Its Organization

We tried to organize this book in an easy-to-understand manner. To begin, we discuss Jython.3 Chapters 2 through 5 describe Jython with enough detail that those unfamiliar with it should be able to readily “pick up” the language and use it effectively.

2 http://www.IBM.com/software/webservers/appserv/was/library/.

3 Python is a programming language, and Jython is an implementation of that language. We won’t worry about this distinction and will continue to refer to it as the Jython programming language for simplicity’s sake.
It should be noted that this book is not meant to be an introduction to programming or even an introduction to the Jython programming language. We could easily fill hundreds more pages on just the topic of “Programming with Jython.” There are lots of books, papers, and websites that discuss the syntax and semantics of Python and therefore the Jython programming language. This book is not intended to replace nor duplicate the information available elsewhere. If you already feel comfortable with Jython (or more precisely Python) as a programming language, you might be able to skip the chapters that describe Jython and begin with Chapter 6.

For those readers who might be less familiar with Jython as a programming language, this book attempts to present the rules, characteristics, and properties of Jython in a logical order, so as to build a solid foundation of information. A little is presented at time and then revisited and expanded upon, providing reinforcement through repetition. Additionally, each chapter is filled with working examples to help you better understand not only the Jython code, but the wsadmin scripting objects as well. Almost all of the examples described are available from the IBM Press website for this book and are provided to minimize the need to search other sources for useful examples.

Chapter 6, “wsadmin,” explores the wsadmin command in detail and explains the command and its parameters thoroughly. A number of the parameters don’t seem to get a lot of use. That may be because when you first start using wsadmin, you can quickly get to a subset of the parameters that you use “all the time.” For many tasks, this is a reasonable approach to getting the job done. However, in so doing, you might have forgotten or not even realize that some wsadmin parameter exists that might make your life easier. This chapter will help broaden your understanding of this command. You never know, you may even find something that you can use right away.

Chapter 7, “Introduction to Admin Objects,” is a must-read because it explains how to configure wsadmin in order to use many of the examples shown later in the book. The wsadmin properties that are shown in this chapter (specifically the changing of the profiles directive) provide an environment upon which many of the later examples depend.

Chapter 8, “The AdminConfig Object,” is the beginning of the explanation of the wsadmin scripting objects. Specifically, it explains the AdminConfig scripting object in great detail. Many administrative scripts deal primarily with the AppServer configuration use and depend upon this scripting object for the vast majority of these manipulations. A number of useful examples are provided to demonstrate just how helpful this scripting object can be.

Chapter 9, “The AdminControl Object,” describes the AdminConfig scripting object in similar detail. This object is used by scripts that need to manipulate active AppServer objects (MBeans). So this chapter is full of useful information for those types of scripts.

---

4 Please note, however, that the version of Jython that is provided and supported by the wsadmin utility does not include all of the features and facilities available in the “latest and greatest” version of Python. So, keep this in mind as you are writing your wsadmin scripts.

Chapter 10, “The AdminApp Object,” covers the AdminApp scripting object in detail. This object is used to list, view, install, uninstall, and modify AppServer applications. As such, this chapter explains how to perform these operations using the AdminApp object methods.

Chapter 11, “The AdminTask Object—Server Management,” is where we begin the description of some of the AdminTask scripting object methods. The AdminTask object is enormous. Additionally, as is explained in this chapter, the methods included vary based upon a number of factors. The scope (as in breadth and depth) of this object is huge. It includes hundreds and hundreds of methods. As such, there is no way for it to be adequately covered in a single chapter. In fact, Chapters 12 through 15 largely deal with AdminTask object methods. That’s not all they cover, but it is at the core of each of these chapters.

Chapter 12, “Scripting and Security,” deals with scripting and security. As such, it addresses a number of security-related items. Anyone who needs to administer an enterprise application server should be familiar with the topics described in this chapter.

Chapter 13, “Databases and Authorization,” covers databases and authorization. Even though the configuration and administration of database-related resources can seem overwhelming, this chapter presents these topics in a simple fashion. You discover the easy way to configure the database-related resources (“the plumbing,” if you will) required for interactions with a database. These explanations include descriptions about the properties you can control and those that are automatically configured for you. The chapter then moves into exploring and manipulating the more complicated aspects and properties of database-related resources and then finishes with a detailed explanation of commonly used database and authorization AdminTask methods.

Chapter 14, “Messaging,” is all about messaging. This complicated topic is explored in detail but explained simply. The chapter starts by discussing messaging basics and then adds a discussion of security. It ends by explaining the more commonly used messaging AdminTask methods in great detail.

Chapter 15, “Administering Web Services,” is all about Web services. So what exactly are Web services? The IBM online documentation has this to say about Web services:

Web services are self-contained, modular applications that can be described, published, located, and invoked over a network. They implement a services-oriented architecture (SOA), which supports the connecting or sharing of resources and data in a very flexible and standardized manner. Services are described and organized to support their dynamic, automated discovery and reuse.

This chapter describes Web services and explains how they should be managed. It also discusses policies, policy sets, bindings and even the use of keystores for the security-related aspects of Web services.

Lastly, Chapter 16, “Version 7.0 Scripting Libraries,” is about the scripting object libraries that are included in version 7 of the AppServer product. These Jython libraries demonstrate some techniques for the management and manipulation of AppServer entities. Some people find the programming interface provided by some of the wsadmin scripting objects difficult to understand and even harder to work with. These libraries present another approach and provide methods that use simpler parameter lists to some of these scripting object methods.
Index

A
abs( x ) method, 77
accessing
dictionaries, 14
sequence elements, 12-13
addGroupToBusConnector-
Role method, 338
addGroupToDefaultRole
method, 338
addGroupToDestinationRole
method, 338
addGroupToTopicRole
method, 339
addGroupToTopicSpaceRoot-
Role method, 340
adding
content, 209
hosts, 258
libraries, 395
resources to security
domains, 262
addNodeGroupMember
command, 224
addPartialAppToAnApp-
WithUpdateCommand() method, 406
addPolicyType command, 382
addSIBusMember method, 318-320
addSIBWSInboundPort
method, 376
addSIBWSOutboundPort
method, 376
addUpdateSingleModuleFile-
ToAnAppWithUpdate-
Command() method, 405
addUserToBusConnectorRole
method, 338
addUserToDefaultRole
method, 338
addUserToDestinationRole
method, 339
addUserToTopicRole
method, 340
addUserToTopicSpaceRoot-
Role method, 340
AdminApp command, 355
AdminApp object, 199
application installation
methods, 204-208
editing, 208-210
methods, 199-212
updating, 208-210
AdminApp.getDeployStatus()
method, 204
AdminApp.isAppReady()
method, 200
AdminApp.updateInteractive(
) method, 208
AdminApplication objects
AdminAuthorizations
modules, 428-430
AdminClusterManage-
tment modules, 430-432
AdminJ2C modules, 412-414
AdminJDBC modules,
414-416
AdminJMS modules,
416-423
AdminNodeGroupManagement modules, 445-446
AdminNodeManagement modules, 446-447
AdminResources modules, 423-427
AdminServerManagement modules, 432-445
business-level applications, 409-412
scripting libraries, 399-409
AdminApplication.getAppDeploymentTarget() method, 404
AdminAuthorizations modules, 428-430
AdminBLA scripting library files, 409-412
AdminClusterManagement modules, 430-432
AdminConfig object, 149
config ID, 150-155
configuration types, 152-153
containment paths, 151
create/modify methods, 160-162
document manipulation methods, 164
methods, 164-165
overview of, 149-150
show/tell methods, 155-159
verification, 162-163
AdminConfig.create() method, 211
AdminConfig.getid() method, 151
AdminConfig.hasChanges() method, 162
AdminConfig.list() method, 152
AdminConfig.parents() method, 160
AdminConfig.remove() method, 162
AdminConfig.reset() method, 352
AdminConfig.save() method, 352
AdminConfig.show() method, 153
AdminConfig.showall() method, 156
AdminConfig.types() method, 160
AdminConfig.validate() method, 163
AdminControl object, 167
attributes, 174-181
environment information, 167-172
MBean support methods, 172-173
methods, 181-182
methods*_imx, 183-184
names, 173-174
AdminControl.startServer() method, 168
administration objects, 129-134
documentation, 145-147
Help, 134-145
security, enabling, 249-250
Web services, 352
exporting WSDL, 355
KeyManager commands, 377
KeySet commands, 381-382
KeyStore commands, 378-381
listing, 353-354
navigating, 353-355
overview, 346-350
policy sets, 355-365, 367-371
PolicySetManagement group, 382-390
references, 373-376
runtimes, 351-352
SIBWebServices group, 374-377
SOAP, 347
topics, 371-372
WebServicesAdmin group, 390-392
WS-I profiles, 346-347
WSDL, 347
wsadmin program, 113-115
commands, 126-128
connecting, 120
defining scripting languages, 122
evironment initialization, 116-120
JVM initialization, 116
options, 115
profile script files, 123-126
tracing, 122
usage information, 116
AdminJ2C modules, 412-414
AdminJDBC modules, 414-416
AdminJMS modules, 416-423
AdminLibHelp module, 447
AdminNodeGroupManagement modules, 445-446
AdminNodeManagement modules, 446-447
AdminReports Command Group command, 226
AdminResources modules, 423-427
AdminServerManagement modules, 432-445
AdminTask methods, 145
databases. See databases
server management, 199-200
command, 209-212
creating clusters, 205-209
examples, 201-202
JVM methods, 216-217
JVM properties, 218-219
JVM system properties, 217-218
methods, 202-205
references, 223-239
template-related commands, 214-216
z/OS methods, 220-223
AdminTask.listPolicySets() method, 356
AdminTask.listServerTypes() method, 203
AdminTask.listWebServices-Operations command, 354
AdminTask.modifySIB* methods, 312
AdminUtilities module, 447
advanced settings, databases, 284-291
aliases, J2C, 242-249
AllAuthenticated group, 311
append( item ) method, 67
applications
AdminApp object, 199
application installation methods, 204-208
ingestion, 208-210
methods, 199-204, 210-212
updating, 208-210
business-level, 409-412
deleting, 407-408
exporting, 406
Federated registries, 260-261
messaging, 301-302
creating buses, 303-308
deleting buses, 308-310
references, 317-327
security, 310-317
terminology, 303
names, 203
policy sets, 356
security, 241-242
configuring, 249-253
enabling, 249-250
J2C, 242-249
Java, 253-255
servers, 2
starting, 408-409
stopping, 408-409
updating, 404-406
wsadmin program, 113-115
commands, 126-128
connecting, 120
defining scripting languages, 122
environment initialization, 116-120
JVM initialization, 116
options, 115
profile script files, 123-126
tracing, 122
usage information, 116
applying config IDs, 153-155
arbitrary function parameters, 55-58
archive files, 116
arithmetic operators, 18
assert statements, 35-36
assignment statements, 16, 28-30
augmented, 32
packing/unpacking, 30-31
slices, 34
attachments, policy sets, 361-371
attributes
AdminControl object, 174-181
MBean Help object, 138-140
missing, 155
policy, 360
RW, 177
augmented assignment statements, 32
authentication
data entry, creating, 244
LDAP failover, 255-260
authorization
AdminAuthorizations library module, 428-430
buses, 313
databases, 277
advanced settings, 284-291
overview of, 277-280
references, 291-300
troubleshooting, 280-284
autoloading scriptLibraries, 395
availability of objects, 133

B
backslash (\) character, 9
beanDescriptions.py, 143
beanInformation.py, 144
index

bindings
  assignment statements, 30
  deleting, 365
  policy sets, 362-371
  slices, 32-33
bitwise operators, 21
Boolean operators, 21
break statements, 36
Browser security role, 311
built-in
  constants, 66
  data types, 67
  exceptions, 93-94
  functions, 77-86
buses
  creating, 303, 306-308
  deleting, 308-310, 326
  messaging, 303
  modifying, 329
  security, 263, 311-314, 337
  viewing, 334
business-level applications, 409-412
busMembers element, 305

C
callable( object ) method, 78
capabilities, 1
capitalize() method, 71
CellStatus.py, 171
center( width ) method, 71
chains, 315
changeClusterShortName() method, 220
changeServerGenericShortName() method, 220
changeServerSpecificShortName() method, 220
characters, escape sequences, 9
chr( i ) method, 78
classes
  hierarchies, 93
  statements, 63-66
ClassNotFoundException, 282
clear() method, 69
client-side policy
  attachments, 368
cloning security domains, 261
clusters
  AdminClusterManagement
    library module, 430-432
    creating, 205-209
code conventions, 26
commands
  AdminApp, 355
  AdminTask.listWebServicesOperations, 354
  command line options, 117
  configureSingleSignon, 269
copySecurityDomain, 271
copySecurityDomainFromGlobalSecurity, 270
create, 210
createAuthDataEntry, 265
createGroup, 273
createSecurityDomain, 270
createUser, 273
deleteAuthDataEntry, 265
deleteGroup, 274
deleteSecurityDomain, 271
deleteUser, 273
dir(), 395
duplicateMembershipOfGroup, 274
duplicateMembershipOfUser, 274
getActiveSecuritySettings, 266
getSingleSignon, 269
getUserRegistryInfo, 269
IdMgrRepositoryConfig
  group, 275
isAppSecurityEnabled, 275
isJACCEnabled, 275
isSingleSecurityDomain, 275
JACCUtilityCommands
  group, 275
KeyManager, 377
KeySet, 381-382
KeyStore, 378-381
listAuthDataEntries, 265-266
listInterceptors, 270
listRegistryGroups, 272
listSecurityDomains, 271
listSecurityRealms, 273
manageprofiles, 203
SecurityConfigurationCommands group, 264-270
SecurityDomainCommands group, 270-271
SecurityRealmInfoCommands group, 272-273
servers, 209-216
serverStatus, 172
setAdminActiveSecuritySettings, 266-267
setAppActiveSecuritySettings, 267-268
setGlobalSecurity, 274
unsetAppActiveSecuritySettings, 268-269
WIMManagementCommands group, 273-274
WizardCommands group, 274-275
wsadmin programs, 126-128
comments, 26, 116
collection operators, 22
compile( patternString [, flags ] ) method, 109
compile( string, filename, kind ) method, 78
complex( real[, imag ] ) method, 78
components, buses
  deleting, 326
  modifying, 329
  viewing, 334
compound statements, 41
  for, 44-46
  if, 42
  loop, 42
  try, 47-48
  while, 42-43
config ID, 150-155
configureSessionManagement
  ForAnApplication() method, 408
configureSingleSignon
  command, 269
configuring
  AdminConfig objects, 149-150
    config ID, 150-155
    configuration types, 152-153
    containment paths, 151
    create/modify methods, 160-162
  document manipulation
    methods, 164
  methods, 164-165
  show/tell methods, 155-159
  verification, 162-163
applications
  Java 2 security, 253-255
  security, 249-253
databases, 277-280
  advanced settings, 284-291
references, 291-300
  troubleshooting, 280-284
  hosts, 258-259
IdMgrRepositoryConfig
  group, 275-276
J2C (JAAS) aliases), 243-244
JACCUtilityCommands
  group, 275
messaging, 301-302
  creating buses, 303-308
  deleting buses, 308-310
  references, 317-327
  security, 310-317
  terminology, 303
permissions, 314
policy set bindings, 365
SecurityConfiguration-
  Commands group, 264-270
SecurityRealmInfo-
  Commands group, 272-273
servers, 205-209
  commands, 209-212
JVM methods, 216-217
JVM properties, 218-219
JVM system properties,
  217-218
  references, 223-239
  template-related
    commands, 214-216
z/OS methods, 220-223
SIBs, 307
strings, 74-77
WIMManagement-
  Commands group, 273-274
WizardCommands group,
  274-275
ConnectionsPool, 283
connections
  leaks, tracing, 284
  messages, 121
wsadmin program, 120
Connector security role, 311
connectSIBWSEndpoint-
  Listener method, 375
constants, built-in, 66. See also literals, 8-15
constructors, finding, 142-143
constructs, OOP, 63
  built-in
    constants, 66
    data types, 67
    functions, 77-86
    class statements, 63-66
    dictionary methods, 69-71
    instantiation, 66
    list methods, 67-68
    string methods, 71-77
containers, EJB, 351
containment paths, 151
content, adding, 209
continue statements, 36
core flow, 39-41
conventions, code, 26
convertToCluster.py, 164
copy() method, 69
copyBinding command, 389
copying
  bindings, 363
  slices, 32-33
copyPolicySet command, 388
copySecurityDomain
  command, 271
copySecurityDomainFrom-
  GlobalSecurity command, 270
count( item ) method, 67
count( substring[, start
  [ , end ] ] ) method, 71
create commands, 210
create() method, 160-162
createApplicationServer() command, 234
createApplicationServer() method, 209
createApplicationServerTemplate command, 235
createApplicationServerTemplate() method, 214
createAuthDataEntry command, 265
createCluster command, 226
createClusterMember command, 228
createCoreGroup command, 224
createDataSource, 292
createGenericServer() method, 209
createGenericServerTemplate() method, 214
createGroup command, 273
createJDBCProvider, 291
createKeyManager command, 378
createKeySet command, 381
createKeyStore command, 379
createNodeGroup command, 224
createPolicySet command, 383
createPolicySetAttachment command, 383
createProxyServer() method, 204, 209
createProxyServerTemplate() method, 214
createSecurityDomain command, 270
createServerType command, 238
createServerType() method, 202-203
createSIBDestination method, 320-322
createSIBEngine method, 322-323
createSIBJMSActivationSpec method, 325-326
createSIBJMSConnectionFactory method, 323-324
createSIBJMSQueue method, 324-325
createSIBJMSSTopic method, 325
createSIBus method, 317-318
createSIBWSEndpointListener method, 375
createSIBWSInboundService method, 374
createSIBWSOutboundService method, 374
createUser command, 273
createWebServer() method, 209
createWebServerTemplate() method, 214
Creator security role, 311
customizing databases, 284-291
J2C (JAAS) aliases, 245
references, 291-300
wsadmin programs, 115
environment initialization, 116-120
JVM initialization, 116
usage information, 116
sequences, accessing elements, 12-13
strings, 8-11
tuples, 11
databases authorization, 277
advanced settings, 284-291
references, 291-300
troubleshooting, 280-284
overview of, 277-280
DataSources, 278-280,
296-298
security, 263
DCS (Distribution and Consistency Services), 315
DDL (data definition language), 406
default failure actions, scripting libraries, 398
default function parameters, 53-62
defining scripting languages, 122
definitions classes, 66
functions, 49-50
del statements, 36
deleteAuthDataEntry command, 265
deleteCluster command, 230
deleteClusterMember command, 230
deleteCoreGroup command, 226
deleteGroup command, 274
deletePartialAppToAppWithUpdateCommand() method, 406
deleteSecurityDomain command, 271
deleteServer command, 236

D
data definition language (DDL), 406
data types, 6-7
built-in, 67
dictionaries, 14
lists, 12
numbers, 7
deleteServer() method, 210
deleteServerTemplate() method, 214
deleteSIBDestination method, 328-329
deleteSIBJMSActivationSpec method, 327
deleteSIBJMSConnectionFactory method, 327
deleteSIBJMSQueue method, 327
deleteSIBJMSTopic method, 327
deleteSIBus method, 327
deleteSingleModuleFileToAnAppWithUpdateCommand() method, 405
deleteUser command, 273
deleteUserAndGroupEntries() method, 210
deleteWebServer() method, 210
deleting bindings, 365
buses, 308-310, 326
hosts, 259-260
J2C (JAAS) aliases, 246-249
servers, 207
deploying applications, 407-408
destinations, 303
dictionaries, 14
methods, 69-71
parameters, unpacking, 57
dir( [object] ) method, 78
dir() command, 395
directories
importing, 395
structures, 150
disabling security, 249-250
Distribution and Consistency Services (DCS), 315
divmod( a, b ) method, 78
docs
admin objects, 145-147
function strings, 61
manipulation methods, 164
domains
multiple security, 261-262
security, adding resources to, 262
SecurityDomainCommands group, 270-271
duplicateMembershipOfGroup command, 274
duplicateMembershipOfUser command, 274
errors, 92-94
OOM, 86
escape sequences, 9
escape( string ) method, 109
eval( expression[, globals[, locals]] ) method, 79
Everyone group, 311
exceptions, 92
built-in, 93-94
ClassNotFoundException, 282
IndexOutOfBoundsException, 282
NameError, 68
scripting libraries, 399
exchangeSigners command, 380
exec statements, 38
execfile( filename[, globals[, locals]] ) method, 79
expandtabs( [tabsize] ) method, 71
expressions
overview of, 18
regular *RegExp, 107-112
statements, 27-28
expressions
overview of, 18
regular *RegExp, 107-112
statements, 27-28
extend( item ) method, 67
extracting methods, names, 135
F
failonerror parameters, 399
failover, LDAP, 255-260
Federated registries, 260-261
filenames, 203, 206
files
  archive, 116
  help, 114
  modules, 87
  errors, 92-94
  import statements, 88-90
  nested_scopes, 90-92
  packages, 92
  profile script, 123-126
  script, 126-128
  script library, 394
  wsadmin.properties, 133
filter( function, sequence )
  method, 79
find( substring[, start[, end ]] )
  method, 71
findall( pattern, string )
  method, 109
findall(string [, startPos [, endPos ]] ) method, 110
finding constructors, 142-143
fixFileName() method, 207
float( x ) method, 79
flow, control, 39-41
for statements, 44-46
formatting. See also configuring
databases, 277-280
  advanced settings, 284-291
  references, 291-300
  troubleshooting, 280-284
indentation, 40
J2C (JAAS) aliases, 243-244
JVM properties, 218
messaging, 301-302
  creating buses, 303-308
  deleting buses, 308-310
  references, 317-327
  security, 310-317
terminology, 303
servers, 205-209
  commands, 209-212
  JVM methods, 216-217
  JVM properties, 218-219
  JVM system properties, 217-218
  references, 223-239
template-related commands, 214-216
z/OS methods, 220-223
strings, 74-77
functional programming, 94
functions, 49
  built-in, 77-86
  definitions, 49-50
  global statements, 51-52
MBattrAsDict() utility, 179
namespaces, 51
parameters, 52
  arbitrary, 55-58
  default, 53-62
  named, 54-55
return statements, 58-59
showAsDict, 154
getattr( object, name[, default ] ) method, 80
getAttribute() method, 175, 179
getAttributes() method, 175
getAttributes_jmx() method, 183
getBinding command, 386
defaultBindings command, 387
gDeployStatus() method, 204
gDefaultBindings method, 220
gEvent() method, 152
gJavaHome() method, 220
gKeyStoreInfo command, 380
gKeyStoreInfo() method, 223
gMBeanCount() method, 172
gObjectName() method, 165
gopt() library routine, 100-102
gPolicySet command, 385
gPolicySetAttachments command, 386
gPolicyType command, 385
getRequiredBindingVersion command, 387
gServerType command, 233
gServerType() method, 202
gSingleSignon command, 269
getters, 140
getUserRegistryInfo command, 269
getWebService method, 392
global security, 249, 311. See also security
global statements, 51-52
globals() method, 80
group( [group] ) method, 110
groupdict() method, 110
groups
  AdminNodeGroup
    Management library module, 445-446
  IdMgrRepositoryConfig, 275-276
  JACCUtilityCommands, 275
  JDBCProviderManagement, 291-293
  mapping, 251-253
  PolicySetManagement, 382-390
  SecurityConfiguration-Commands, 264-270
  SecurityDomain-Commands, 270-271
  SecurityRealmInfo-Commands, 272-273
  ServerManagement, 209
  SIBWebServices, 374-377
  VariableConfiguration, 294-296
  WebServicesAdmin, 390-392
  WIMManagement-Commands, 273-274
  WizardCommands, 274-275
  Help.descriptions() method, 143
  hex(x) method, 80
  hierarchies, classes, 93
  hosts
    adding, 258
    configuring, 258-259
    deleting, 259-260
    viewing, 257
  id(object) method, 80
  identifiers, 15-16
  identity operators, 24
  IdMgrRepositoryConfig group, 275-276
  IEEE Standard for Binary Floating-Point Arithmetic (ANSI/IEEE Std 754-1985), 7
  if statements, 42
  import statements, 39, 88-90
  importBinding command, 388
  importing directories, 395
  importPolicySet command, 387
  importServer command, 231
  *_imx methods, 183-184
  indentation, 40
  index(item) method, 68
  index(substring[, start [, end ]]) method, 71
  IndexError exceptions, 68
  indexing sequence elements, 13
  infinite loops, 42
  information types, 15
  expressions, 18
  identifiers, 15-16
  literals, 15
  statement separators, 25
  string operators, 19-24
  variables, 16-18
  initializations
    environments, 116-120
    JVM, 116
  input([prompt]) method, 80
  insert(index, item) method, 68
  installing applications, 400-402
  installInteractive() method, 207
  installResourceAdapter() method, 165
  instances, AdminControl objects, 173-174
  instantiation, 66
  int(x[, radix]) method, 81
  Integrated Solutions Console, 251
  integrating SIBs, 307
  interactive wsadmin sessions, 43
  interfaces, JNDI, 211
  invoke() method, 181
  isalnum() method, 71
  isalpha() method, 72
  isAppReady() method, 201
  isAppSecurityEnabled command, 275
  isdigit() method, 72
  isFederated() method, 220
  issubclass(object, classinfo) method, 81
  isJACCEnabled command, 275
  islower() method, 72
  isspace() method, 72
  isSingleSecurityDomain command, 275
  issubclass(class, classinfo) method, 81
  istitle() method, 72
  isupper() method, 72
  items() method, 70
J

J2C (J2EE Connector Architecture), 242-249
authentication aliases, 278
library modules, 412-414
JAASAuthData objects, 243, 313
JACCUtilityCommands group, 275
Java, 95-103
roles, mapping, 250
security, enabling, 253-255
Java 2 Connector Architecture. See J2C
Java Messaging Service (JMS), 307
Java Naming and Directory Interface (JNDI), 211
Java Virtual Machine. See JVM

JDBC (Java Database Connector)
 AdminJDBC library modules, 414-416
providers, 278
JDBCProvider, troubleshooting, 281
JDBCProviderManagement Group methods, 291-293
JMS (Java Messaging Service), 307
 AdminJMS library modules, 416-423
JNDI (Java Naming and Directory Interface), 211
join( sequence ) method, 72
JVM (Java Virtual Machine), 6
 initialization, 116
methods, 216-217
properties, 218-219
system properties, 217-218

Jython
coding conventions, 26
comments, 26
data types, 6-7
accessing sequence elements, 12-13
dictionaries, 14
lists, 12
numbers, 7
strings, 8-11
tuples, 11
information types, 15
expressions, 18
identifiers, 15-16
literals, 15
statement separators, 25
string operators, 19-24
variables, 16-18
overview of, 6

len( s ) method, 81
libraries
Java, 95-105
scripting
AdminApplication objects, 399-409
AdminAuthorizations modules, 428-430
AdminClusterManagement modules, 430-432
AdminJ2C modules, 412-414
AdminJDBC modules, 414-416
AdminJMS modules, 416-423
AdminNodeGroupManagement modules, 445-446
AdminNodeManagement modules, 446-447
AdminResources modules, 423-427
AdminServerManagement modules, 432-445
business-level applications, 409-412
navigating, 393-397
troubleshooting, 397-399

K

KeyManager commands, 377
keys
policy sets, 362-363
properties, 117
keys() method, 70
KeySet commands, Web services, 381-382
KeyStore commands, Web services, 378-381
keytools, 363
keywords, 16, 58

L

languages
 Jython. See Jython
scripting, defining, 122
lastgroup, 111
lastindex, 111
LDAP failover, 255-260
LDAPUserRegistry, 256
leaks, tracing connection, 284

list( sequence ) method, 81
listAllDestinationsWithRoles method, 341
listAllForeignBusesWithRoles method, 341
listAllRolesForGroup method, 341
listAllRolesForUser method, 341
listAllTopicsWithRoles method, 341
listApplicationPorts
command, 232
listAssetsAttachedToPolicy-
Set command, 385
listAttachmentsForPolicySet
command, 384
listAuthDataEntries
command, 265-266
listDataSources, 294
listGroupsInBusConnector-
Role method, 341
listGroupsInDefaultRole
method, 342
listGroupsInDestinationRole
method, 342
listGroupsInTopicRole
method, 342
listGroupsInTopicSpaceRoot
Role method, 342, 344
listInterceptors command, 270
listJDBCProviders, 293
listKeyFileAliases
command, 380
listKeyManagers
command, 377
listKeyStores command, 378
listKeyStoreTypes
command, 378
listModules() method, 201
listPolicySets command, 384
listPolicyTypes
command, 384
ListPorts.py, 159
listRegistryGroups
command, 272
lists, 12
   methods, 67-68
      Web services, 352-354
listSecurityDomains
command, 271
listSecurityRealms
command, 273
listServer command, 233
listServerPorts
command, 232
listServers() method, 210
listServerTemplates()method, 214
listServerTypes
command, 233
listServerTypes() method, 202
listServices method, 392
listSIBEngines method, 335
listSIBJMSQueues
method, 336
listSIBJMSTopics
method, 336
listSIBMeditations
method, 337
listSIBuses method, 334
listSIBusMembers
method, 334
listUsersInBusConnectorRole
method, 343
listUsersInDefaultRole
method, 343
listUsersInDestinationRole
method, 343
listUsersInTopicRole
method, 343
listWebServiceEndpoints
method, 391
listWebServiceOperations
method, 391
listWebServices method, 391
literals, 8-15
ljust( width ) method, 73
local mode, 120
local precedence, 51
locals() method, 81
long strings, 8
long( x[, radix ] ) method, 81
loop statements, 42
lower() method, 73
lstrip() method, 73

M
management. See also
administration
AdminClusterManagement
library module, 430-432
AdminNodeGroupManage-
ment library module, 445-446
AdminNodeManagement
library module, 446-447
AdminServerManagement
library module, 432-445
databases, 277-280
   advanced settings, 284-291
   references, 291-300
   troubleshooting, 280-284
servers, 199-200
   commands, 209-212
   creating clusters, 205-209
   examples, 201-202
   JVM methods, 216-217
   JVM properties, 218-219
   JVM system properties, 217-218
   methods, 202-205
   references, 223-239
template-related
commands, 214-216
z/OS methods, 220-223
Web services, 352
   exporting WSDL, 355
KeyManager
   commands, 377
   KeySet commands, 381-382
   KeyStore commands, 378-381
listing, 353-354
navigating, 353-355
overview, 346-350
policy sets, 355-371
PolicySetManagement
group, 382-390
references, 373-376
runtimes, 351-352
SIBWebServices group,
374-377
SOAP, 347
topics, 371-372
WebServicesAdmin
group, 390-392
WS-I profiles, 346-347
WSDL, 347
wsadmin program,
113-115
commands, 126-128
connecting, 120
defining scripting
languages, 122
environment
initialization, 116-120
JVM initialization, 116
options, 115
profile script files,
123-126
tracing, 122
usage information, 116
manageprofiles command, 203
map( function, sequence, ... )
method, 82
mapping
Java EE roles, 250
reviewing, 251
users, 251-253
match( pattern, string
[, flags ] ) method, 109
match( string [, startPos [, endPos ] ] method, 110
max( sequence ) method, 82
MBattrAsDict() utility
function, 179
MBBeans, 296
AdminController object
support methods,
172-173
DataSource, 296-298
Help object, 136-144
TraceService, 298-300
MDBs (Message-Driven
Beans), 352
members, bus, 303
membership operators, 24
memory, OOM errors, 86
Message-Driven Beans
(MDBs), 352
messaging
buses
creating, 303-308
deleting, 308-310
connections, 121
Help object, 145
overview, 301-302
references, 317-344
security, 310-317
terminology, 303
methods
addGroupToBusConnectorRole, 338
addGroupToDefaultRole, 338
addGroupToDestinationRole, 338
addGroupToTopicRole, 339
addGroupToTopicSpaceRootRole, 340
addUserToBusConnectorRole, 338
addUserToDefaultRole, 338
addUserToDestinationRole, 339
addUserToTopicRole, 340
addUserToTopicSpaceRootRole, 340
AdminController object,
200-212
application installation
methods, 204-208
editing, 208-210
updating, 208-210
AdminController.getDeployStatus(), 204
AdminController.isAppReady(), 200
AdminController.updateInteractive(), 208
AdminApplication.getAppDeploymentTarget(), 404
AdminConfig object,
164-165
AdminConfig.create(), 211
AdminConfig.getid(), 151
AdminConfig.list(), 152
AdminConfig.parents(), 160
AdminConfig.remove(), 162
AdminConfig.reset(), 352
AdminConfig.save(), 352
AdminConfig.show, 153
AdminConfig.showall(), 156
AdminConfig.types(), 160
AdminConfig.validate(), 163
AdminController objects,
181-182
*_inx, 183-184
MBean, 172-173
AdminController.startServer(), 168
AdminTask, 264
AdminTask.listPolicySets(), 356
create(), 160-162
createSIBDestination,
320-322
createSIBEngine, 322-323
createSIBJMSActivationSpec, 325-326
createSIBJMSCConnectionFactory, 323-324
createSIBJMSSQueue, 324-325
createSIBJMSTopic, 325
createSIBus, 317-318
deleteSIBDestination, 328-329
deleteSIBJMSActivationSpec, 327
deleteSIBJMSCConnectionFactory, 327
deleteSIBJMSSQueue, 327
deleteSIBJMSTopic, 327
deleteSIBus, 327
deleteUserAndGroupEntries(), 210
dictionary, 69-71
export(), 210
exportDDL(), 210
fixFileName(), 207
getAttribute(), 175-179
getAttributes(), 175
getAttributes_jmx(), 183
getDeployStatus(), 204
getId(), 152
getMBeanCount(), 172
getObjectName(), 165
Help object, 136
help(), scripting libraries, 397-399
Help.constructors(), 142
Help.descriptions(), 143
installInteractive(), 207
installResourceAdapter(), 165
invoke(), 181
isAppReady(), 201
JDBCProviderManagement Group, 291-293
JVM, 216-217
listAllDestinationsWithRoles, 341
listAllForeignBusesWithRoles, 341
listAllRolesForGroup, 341
listAllRolesForUser, 341
listAllTopicsWithRoles, 341
listGroupsInBusConnectorRole, 341
listGroupsInDefaultRole, 342
listGroupsInDestinationRole, 342
listGroupsInTopicRole, 342
listGroupsInTopicSpaceRootRole, 342-344
listModules(), 201
lists, 67-68
listSIBEngines, 335
listSIBJMSSQueues, 336
listSIBJMSTopics, 336
listSIBMediations, 337
listSIBuses, 334
listSIBusMembers, 334
listUsersInBusConnectorRole, 343
listUsersInDefaultRole, 343
listUsersInDestinationRole, 343
listUsersInTopicRole, 343
modify(), 160-162
modifySIBDestination, 330-331
modifySIBEngine, 330
modifySIBJMSSQueue, 332
modifySIBJMSTopic, 333
modifySIBus, 329
names, extracting, 135
options(), 203
publishWSDL(), 211
queryNames(), 174
removeDefaultRoles, 337
removeGroupFromAllRoles, 337
removeSIBusMember, 328
removeUserFromAllRoles, 337
removeVariable, 296
restart(), 182
searchJNDIReferences(), 211
servers, 202-205
setAttributes(), 179-180
setAttributes_jmx(), 184
setVariable, 295
show(), 155-159
showAsDict(), 157
showSIBEngine, 336
showSIBJMSCConnectionFactory, 336
showSIBJMSSQueue, 336
showSIBJMSTopic, 336
showSIBMediation, 337
showSIBus, 334
showSIBusMember, 335
showVariables, 294
startServer(), 169
stopServer(), 169
strings, 71-77
tell(), 155-159
uninstall(), 208
uninstallResourceAdapter(), 165
updateAccessIDs(), 212
updateInteractive(), 209
VariableConfiguration Group, 294-296
view(), 203
z/OS, 220-223
min( sequence ) method, 82
missing attributes, 155
modes, local, 120
modify() method, 160-162
modifying
  AdminControl objects, 167-172
  buses, 329
  configuration, 150
  J2C (JAAS) aliases, 245
  runtime behavior, 283
  security domains, 261
modifyKeySet command, 382
modifySIBDestination method, 330-331
modifySIBEngine method, 330
modifySIBJMSQueue method, 332
modifySIBJMSTopic method, 333
modifySIBus method, 329
modules
  AdminApplication script library, 399-409
  AdminAuthorizations, 428-430
  AdminClusterManagement, 430-432
  AdminJ2C, 412-414
  AdminJDBC, 414-416
  AdminJMS, 416-423
  AdminLibHelp, 447
  AdminNodeGroupManagement, 445-446
  AdminNodeManagement, 446-447
  AdminResources, 423-427
  AdminServerManagement, 432-445
  AdminUtilities, 447
  errors, 92-94
  import statements, 88-90
  names, 203
  namespaces, 124
  nested_scopes, 90-92
  overview, 87
  packages, 92
  simple, 129
  sys, 98-99
  moduleTest.py, 131
  moveClusterToCoreGroup command, 225
  moveServerToCoreGroup command, 225
  multiple security domains, 261-262

N
  named function parameters, 54-55
  NameError exception, 132
  names
    AdminControl object, 173-174
    applications, 203
    filenames, 203
    JNDI, 211
    methods, extracting, 135
    modules, 203
    variables, 15-16
  namespaces, 51
  modules, 124
  wsadmin, 123
  navigating
    J2C (JAAS) aliases, 244-245
    LDAPUserRegistry, 256
    scripting libraries, 393-397
    Web services, 353-355
  negative indexes, 13
  nested_scopes, 90-92
  nodes
    AdminNodeGroupManagement library module, 445-446
    AdminNodeManagement library module, 446-447

O
  object-oriented programming.
  See OOP
  objects
    AdminApp, 199
    application installation methods, 204-208
    editing, 208-210
    methods, 199-212
    updating, 208-210
    AdminApplication, 399-409
    AdminConfig, 149
    config ID, 150-155
    configuration types, 152-153
    containment paths, 151
    create/modify methods, 160-162
    document manipulation methods, 164
    methods, 164-165
    overview of, 149-150
    show/tell methods, 155-159
    verification, 162-163
    AdminControl, 167
    attributes, 174-181
    environment information, 167-172
    *_imx methods, 183-184
    MBean support methods, 172-173
    methods, 181-182
    names, 173-174
    administration, 129-134
    documentation, 145-147
    Help, 134-145
    availability, 133
    dictionary, 69-71
    instantiation, 66
Java, 95-105
lists, 67-68
modules, 87
errors, 92-94
import statements, 88-90
nested_scopes, 90-92
packages, 92
Security, 278
strings, 71-77
target, 210
OOM (out of memory) errors, 86
OOP (object-oriented programming), 63
   built-in
      constants, 66
data types, 67
functions, 77-86
class statements, 63-66
dictionary methods, 69-71
instantiation, 66
list methods, 67-68
string methods, 71-77
open( filename[, mode[, bufsize ] ] ) method, 82
operators
   arithmetic, 18
   augmented assignment statements, 32
   bitwise, 21
   Boolean, 21
   comparison, 22
   identity, 24
   membership, 24
   relationships, 22
   strings, 19-24
   unary, 21
options
   command line, 117, 284-294
   wsadmin program, 115
   environment
      initialization, 116-120
   JVM initialization, 116
   usage information, 116
   options() method, 203
   ord( c ) method, 82
out of memory (OOM) errors, 86

packages, 92
packing assignment statements, 30-31
parameters
   failon error, 399
   functions, 52
      arbitrary, 55-58
      default, 53-62
      named, 54-55
scripts, 127
sequences, unpacking, 56
ParentTypes.py, 161
parmTest() method, 103-104
parseOpt() method, 104-106
pass statements, 39
passing filenames to scripts, 206
paths, containment, 151
period (.), 63
permissions, configuring, 314
ping, 348-350
PingServiceJAXRPCApplication, 353
POJO (Plain Old Java Object), 351
policy sets, Web services, 355-365, 367-371
PolicySetManagement group, 382-390
pop( [ index ] ) method, 68
pos, 111
pow( x, y[, z ] ) method, 82
precedence
   local, 51
   operators, 20, 24
print statement, 27-28
processing command line options, 100-102
profiles
   script files, 123-126
   WS-I, 346-347
programming
   functional, 94
   Jython
      accessing sequence elements, 12-13
coding conventions, 26
comments, 26
data types, 6-7
dictionaries, 14
expressions, 18
identifiers, 15-16
information types, 15  
lists, 12  
literals, 15  
numbers, 7  
operators, 19-24  
overview of, 6  
statement separators, 25  
strings, 8-11  
tuples, 11  
variables, 16-18  
OOP, 63  
built-in constants, 66  
built-in data types, 67  
built-in functions, 77-86  
class statements, 63-66  
dictionary methods, 69-71  
instantiation, 66  
list methods, 67-68  
string methods, 71-77  
programs  
wsadmin, 113-115  
commands, 126-128  
connecting, 120  
defining scripting languages, 122  
environment initialization, 116-120  
JVM initialization, 116  
options, 115  
profile script files, 123-126  
tracing, 122  
usage information, 116  
properties, 121  
JVM, 217-223  
keys, 117  
profileToDict() method, 95  
publishSIBWSInboundService method, 377  
publishWSDL() method, 211  
Q  
QOS (Qualities of Service), 355  
query application configurations, 403-404  
queryNames() method, 174  
queues, messaging  
creating buses, 303-308  
deleting buses, 308-310  
security, 310-317  
R  
RAD (Rational Application Developer) tool, 361  
raise statements, 39  
range( [start[,] stop[, step ] ) method, 82  
Rational Application Developer (RAD) tool, 361  
raw strings, 11  
raw_input( [prompt] ) method, 83  
re (regular repression) function, 111  
Receiver security role, 311  
reduce( function, sequence[, initialValue ] ) method, 83  
references  
databases, 291-300  
messaging, 317-344  
servers, 223-239  
Web services, 373-376  
registries, Federated, 260-261  
regular expressions (RegExp), 107-112, 135, 157  
relationship operators, 22  
reload( module ) method, 84  
remove( index ) method, 68  
removeDefaultRoles method, 337  
removeGroupFromAllRoles method, 337  
removeNodeGroup command, 225  
removeNodeGroupMember command, 224  
removeSIBusMember method, 328  
removeUserFromAllRoles method, 337  
removeVariable method, 296  
replace( old, new[, count ] ) method, 73  
reportConfigInconsistencies command, 226  
ReportConfiguredPorts command, 226  
repr( object ) method, 84  
resources, AdminResources library module, 423-427  
restart() method, 182  
return statement, 58-59  
reverse() method, 68  
reviewing mappings, 251  
replace( old, new[, count ] ) method, 73  
replace( substring[, start [,end ] ] ) method, 73  
rindex( substring[, start[, end ] ] ) method, 73  
rjust( width ) method, 73  
RMI/IIOP connections, 121  
roles  
Java, 250  
mapping, 251-253  
security, 311  
round( x[, digits ] ) method, 84  
rstrip() method, 73  
runtimes  
behavior, modifying, 283  
Web services, 351-352  
RW attributes, 177
S

scope, nested_scopes, 90-92

scripting
administration objects, 129-134
documentation, 145-147
Help, 134-145
functional programming, 94
languages, defining, 122
libraries
AdminApplication objects, 399-409
AdminAuthorizations modules, 428-430
AdminClusterManagement modules, 430-432
AdminJ2C modules, 412-414
AdminJDBC modules, 414-416
AdminJMS modules, 416-423
AdminNodeGroupManagement modules, 445-446
AdminNodeManagement modules, 446-447
AdminResources modules, 423-427
AdminServerManagement modules, 432-445
business-level applications, 409-412
navigating, 393-397
troubleshooting, 397-399
profile script files, 123-126

S

support for, 1
troubleshooting, 92-94
wsadmin program commands, 126-128
scripts
filenames, passing, 206
Jython, 6. See also Jython
search( pattern, string [, flags ] ) method, 109
search( string [, startPos [, endPos ] ] ) method, 110
searching chains, 315
searchJNDIReferences() method, 211
SecureConversation policy set, 357
security
AdminTasks methods, 264
applications, configuring, 249-253
bus, 263
buses, 337
datasources, 263
Federated registries, 260-261
J2C, 242-249
Java, enabling, 253-255
LDAP failover, 255-260
message, 310-317
multiple domains, 261-262
overview of, 241-242
Web services, 264
Security object, 278
SecurityConfigurationCommands group, 264-270
SecurityDomainCommands group, 270-271
SecurityRealmInfoCommands group, 272-273
Sender security role, 311
separator statements, 25
sequences
elements, accessing, 12-13
escape, 9
parameters, unpacking, 56
slices, 32-33
tuples, 11
unpacking, 31
server-side policy
attachments, 368
ServerManagement group, 209
servers
AdminServerManagement library module, 432-445
applications, 2
deleting, 207
LDAP failover, 255-260
management
commands, 209-212
creating clusters, 205-209
elements, 201-202
JVM methods, 216-217
JVM properties, 218-219
JVM system properties, 217-218
methods, 202-205
references, 223-239
template-related commands, 214-216
z/OS methods, 220-223
management, 199-200
security, 241-242
applications, 249-253
J2C, 242-249
Java, 253-255
serverStatus command, 172
ServerTypes methods, 202-205
service integration buses (SIBs), 307
services, Web. See Web services
sessions, wsadmin programs, 128
setAdminActiveSecuritySettings command, 266-267
setAppActiveSecuritySettings command, 267-268
setattr( object, name, value ) method, 85
setAttributes() method, 179-180
setAttributes_jmx() method, 184
setBinding command, 390
setdefault( key [, defaultValue ] ) method, 70
setGlobalSecurity command, 274
setJVMProperties command, 236
setJVMProperties() method, 219
setJVMSystemProperties command, 237
setProcessDefinition command, 237
setProcessDefinition() method, 220-222
setServerInstance() method, 220
setters, 140
setTraceSpecification() method, 220
setVariable method, 295
show() method, 155-159
showAsDict function, 154
showAsDict() method, 157
showJVMProperties() method, 219
showProcessDefinition() method, 220-222
showServerInfo command, 234
showServerInfo() method, 210
showServerInstance() method, 210
showServerTypeInfo() command, 234
showServerTypeInfo() method, 202
showSIBEngine method, 336
showSIBJMSCollection-Factory method, 336
showSIBMSQueue method, 336
showSIBJMSConnectionFactory method, 336
showSIBJMSMediation method, 337
showSIBus method, 334
showSIBusMember method, 335
showTemplateInfo() method, 214-215
showVariables method, 294
SIBs (service integration buses), 307
SIBWebServices group, 374-377
simple modules, 129
simple statements, 35
assert, 35-36
break, 36
continue, 36
class, 32
control flow, 41
del, 36
for, 44-46
exec, 38
for, 44-46
import, 39
import, 37
pass, 39
raise, 39
range, 39
return, 58-59
return, 58-59
SmartCellStop.py, 170
SOAP, 121, 347
span( [group] ) method, 111
split( string [, maxsplit ] ) method, 109
split( [ separator [,maxsplit ] ] ) method, 73
splitlines( [keepends] ) method, 73
square brackets ([ ]), 153, 157
start( [group] ) method, 110
startApplicationOnSingleServer() method, 408
starting applications, 408-409
startServer() method, 169
startswith( prefix[, start[, end ] ] ) method, 73
Stateless Session Beans, 352
statements
assignment, 16, 28-30
augmented, 32
packing/unpacking, 30-31
slices, 34
class, 63-66
comments, 116
compound, 41
for, 44-46
if, 42
loop, 42
try, 47-48
while, 42-43
expressions, 27-28
global, 51-52
import, 88-90
return, 58-59
separators, 25
simple, 35
assert, 35-36
break, 36
continue, 36
class, 63-66
control flow, 39-41
del, 36
deleted, 36
exec, 38
for, 44-46
import, 39
import, 37
pass, 39
raise, 39
StopCell.py, 170
stopping applications, 408-409
stopServer() method, 169
__str__() method, 93
str( object ) method, 85
strings, 8-11
functions, 111
long, 8
methods, 71-77
operators, 19-24
raw, 11
triple-quoted, 8
strip() method, 74
sub( pattern, replacement, string [, count ] ) method, 110
sub( replacement, string [, count ] ) method, 110
subn( pattern, replacement, string [, count ] ) method, 110
subn( replacement, string [, count ] ) method, 110
swapcase() method, 74
syntax errors, 92-94
sys module, 98-99
dictionary, 131
sys.path variable, 394

T
target objects, 210
tell() method, 155-159
TemplateInfo.py, 215
template servers, 214-216
testing, 169
title() method, 74
tools
    keytools, 363
    RAD, 361
topics, policy, 371-372
TraceService, MBeans, 298-300

tracing
    connection leaks, 284
    wsadmin programs, 122
    transport security, enabling, 314-317
triple-quoted strings, 8
troubleshooting, 92-94
databases, 280-284
    OOM errors, 86
    scripting libraries, 397-399
try statements, 47-48
tuples, 11, 31
type( object ) method, 85
types
    configuration, 152-153
data, 6-7
    accessing sequence elements, 12-13
dictionaries, 14
    lists, 12
    numbers, 7
    strings, 8-11
tuples, 11
    information, 15
    expressions, 18
    identifiers, 15-16
    literals, 15
    statement separators, 25
    string operators, 19-24
    variables, 16-18
    policy, 359

unpacking
    assignment statements, 30-31
    sequence parameters, 56
    unsetAppActiveSecuritySettings command, 268-269
    update( dict ) method, 71
    updateAccessIDs() method, 212
    updateEntireAppToAnAppWithUpdateCommand() method, 406
    updateInteractive() method, 209
    updating
        AdminApp methods, 208-210
        applications, 404-406
    upper() method, 74
    usage information, wsadmin programs, 116
    usage() method, 102-103
    users, mappings, 251-253

V
validatePolicySet command, 388
validation, configuring, 162-163
values, 59
values() method, 71
van Rossum, Guido, 61
VariableConfiguration command group, 282
VariableConfiguration Group methods, 294-296
variables
    names, 15-16
    overview of, 16-18
    slices, 32-33
    sys.path, 394
    VariableSubstitutionEntry, 281
vars([object]) method, 86
verifying configurations, 162-163
view() method, 203
viewing
  buses, 334
  hosts, 257
  J2C (JAAS) aliases, 244-245

W
WAuJ.py, 133
Web services
  KeyManager commands, 377
  KeySet commands, 381-382
  KeyStore commands, 378-381
  listing, 353-354
  managing, 352
  navigating, 353-355
  overview, 346-350
  policy sets, 355-371
  PolicySetManagement group, 382-390
  references, 373-376
  runtimes, 351-352
  security, 264
  SIBWebServices group, 374-377
  SOAP, 347
  topics, 371-372
  WebServicesAdmin group, 390-392
  WS-I profiles, 346-347
  WSDL, 347, 355
Web Services Definition Language. See WSDL
  WebServicesAdmin group, 390-392
  Websphere Application Server, overview of, 2
  WebSphere Control Program (WSCP), 6
  while statements, 42-43
  WIMManagementCommands group, 273-274
  WizardCommands group, 274-275
  WS-I profiles, 346-347
  wsadmin program, 113-115.
  See also administration commands, 126-128
  connecting, 120
  options, 115
    environment initialization, 116-120
    JVM initialization, 116
    usage information, 116
  profile script files, 123-126
  scripting language, 122
  sessions, 43
  tracing, 122
  wsadmin.properties file, 133
  WSASObjects.py, 131-132
  WSCP (WebSphere Control Program), 6
  WSDL (Web Services Definition Language), 211, 347
    exporting, 355
    WSSecurity policy set, 358

X–Z
xrange([start[, stop[, step]])
  method, 86
z/OS methods, 220-223
zfill(width) method, 74