

Inhaltsverzeichnis

Kapitel 1: Konzepte der modularen Programmierung	1
Allgemeine Aspekte der Modularität.....	2
Wartbarkeit	3
Wiederverwendbarkeit.....	3
Moduldefinition	4
Starke Kapselung.....	8
Explizite Schnittstellen.....	9
Hohe Modulkohäsion.....	9
Geringe Modulkopplung	10
Enge Kopplung vs. lose Kopplung	10
Modulare Programmierung.....	19
Prinzipien der modularen Programmierung	19
Vorteile der modularen Programmierung.....	20
Modulare Programmierung vs. objektorientierte Programmierung (OOP)	21
Monolithische Anwendung vs. modulare Anwendung	22
Zusammenfassung	24
Kapitel 2: Project Jigsaw.....	25
Schwächen in Java vor JDK 9	25
Schwache Kapselung.....	27
JAR-Hell-Problem	27
Was ist Project Jigsaw?.....	28
Herunterladen und Installieren.....	30
Dokumentation.....	31
Ziele von Project Jigsaw	32

INHALTSVERZEICHNIS

Neue Konzepte eingeführt in Jigsaw	34
Starke Kapselung	35
Zuverlässige Konfiguration	36
Verbesserungen durch Jigsaw	36
Sicherheit	36
Skalierbarkeit und Leistung	38
Andere Allgemeinheiten	38
Neue Schlüsselwörter in Java 9	38
Keine Versionierung in Jigsaw	39
Rückwärtskompatibilität	39
Plattformmodularisierung	40
Neue Struktur der JRE und JDK	41
Vorbereitung auf Jigsaw	43
Unterschiede zwischen OSGi und Jigsaw	44
Zusammenfassung	45
Kapitel 3: Modulares JDK und Quellcode	47
Modulares JDK	48
Plattformmodule	48
Standardmodule	51
Nicht-Standardmodule	51
Der JDK-Modulgraph	52
Mehr über Module	54
Lesen Sie die Beschreibung eines Moduls	54
Modul <code>java.base</code>	56
Modularer Quellcode	59
Neues Schema für den Quellcode	59
Vergleich Quellcode-Struktur	61
Anpassungen des Build-Prozesses	62
Zusammenfassung	64

Kapitel 4: Definition und Verwendung von Modulen.....	65
Das Konzept des Moduls	65
Moduldeklaration	67
Modulname	69
Fünf Arten von Klauseln	70
Die requires-Klausel	71
Die exports-Klausel.....	80
Die opens-Klausel.....	83
Andere Klauseln.....	85
Kompilieren und ausführen von Modulen.....	85
Kompilieren eines einzelnen Moduls	86
Eine Anwendung mit einem einzigen Modul ausführen.....	87
Mehrere Module kompilieren	89
Eine Anwendung mit mehreren Modulen ausführen.....	92
Private vs. öffentliche Methoden.....	94
Modulare JARs.....	96
Struktur eines modularen JAR	97
Verpackung.....	98
Verpacken als modulares JAR mit dem jar-Tool.....	98
Hinzufügen einer Modulversion	100
Ausgabe des Moduldeskriptors	100
Der Modulpfad	101
Anwendungsmodulpfad	102
Kompilierungsmodulpfad	104
Upgrade-Modulpfad	104
Modulauflösung	105
Root-Modul	106

INHALTSVERZEICHNIS

Zugänglichkeit	107
Lesbarkeit vs. implizite Lesbarkeit	109
Implizite Lesbarkeit.....	109
Qualifizierte Exporte.....	115
Arten von Modulen.....	117
Benannte Module	117
Normale Module.....	118
Automatische Module	118
Grundlegende Module	119
Offene Module.....	119
Aktivierung der Kern-Reflection mit offenen Modulen.....	121
Das unbenannte Modul	124
Beobachtbare Module	126
Zusammenfassung	126
Kapitel 5: Modulare Laufzeitbilder.....	129
Modulare Laufzeitbilder.....	129
Das Laufzeitbild vor Java 9	130
Das JRE-Bild vor Java 9.....	130
Das JDK-Bild vor Java 9	130
Warum ein neues Format für die Laufzeitbilder?.....	131
Das Laufzeitbild in Java 9	132
Identische Struktur des JDK und JRE	132
Die Struktur des neuen Laufzeitbildes.....	132
Die Freigabedatei.....	133
Entfernte Dateien	134
rt.jar entfernt	134
Tools.jar und dt.jar entfernt.....	135

INHALTSVERZEICHNIS

Neues URI-Schema	135
Kompatibilität.....	138
Zusammenfassung	139
Kapitel 6: Services.....	141
Starke Kopplung zwischen Modulen	143
Verwendung von Diensten in JDK 9.....	144
Bereitstellung und Verbrauch von Diensten.....	145
Bereitstellung eines Dienstes.....	146
Verbrauch eines Dienstes	147
Abfragen eines ServiceLoaders.....	148
Verwendung von einem Nutzer und einem Anbieter	149
Verwendung von einem Verbraucher und zwei Anbietern.....	153
Zusammenfassung	154
Kapitel 7: Jlink: Der Java Linker.....	157
Der Java Linker.....	157
Jlink-Bilder.....	159
Jlink Befehlssyntax	160
Jlink-Befehlsoptionen	161
Link-Phase	161
Das Modul jdk.jlink.....	163
Beispiel: Erstellen eines Laufzeitbildes mit Jlink.....	165
Ausführen des Laufzeitbildes.....	177
Modulare JAR-Dateien als Eingabe für das Jlink-Tool	177
Struktur des generierten Laufzeitbildes.....	178
Keine Unterstützung für die Verknüpfung automatischer Module	179
Jlink Plugins	180
Das compress Plugin	180

INHALTSVERZEICHNIS

Das release-info Plugin	182
Das excludes-files Plugin.....	183
Zusammenfassung	183
Kapitel 8: Migration	185
Automatische Module.....	188
Berechnung des Namens des automatischen Moduls	191
Beschreibung einer JAR-Datei	194
Keine Unterstützung für automatische Module zur Link-Zeit.....	195
Das JDeps-Tool.....	196
Finden Sie Abhängigkeiten von nicht unterstützten JDK-internen APIs.....	196
Generieren Sie Modulbeschreibungen mit JDeps	198
Kapselung in Java 9.....	200
Exportieren eines Pakets zur Kompilierungszeit und zur Laufzeit	203
Export zum unbenannten Modul.....	205
Pakete öffnen für tiefe Reflection	206
Lesbarkeit zwischen Modulen bereitstellen.....	208
Hinzufügen von Modulen zum Root-Set.....	209
Die Option --illegal-access	212
Migrationsprobleme	216
Eingekapselte JDK-interne APIs	217
Nicht aufgelöste Module	218
Geteilte Pakete.....	220
Zyklische Abhängigkeiten	224
Neues Versionierungsschema.....	226
Entfernte Methoden in JDK 9	226
Entfernung von rt.jar, tools.jar und dt.jar	227
Migration einer Anwendung zu Java 9	228
Top-down-Migration.....	228
Zusammenfassung	235

Kapitel 9: Die neue Modul-API	237
Die Klasse der Module	239
Attribute	239
Konstruktoren	240
Methoden	240
Änderungen in <code>java.lang.Class</code>	242
Die <code>ModuleDescriptor</code> -Klasse	243
<code>ModuleDescriptor</code> -Attribute	244
Methoden des <code>ModuleDescriptor</code>	245
Die <code>ModuleDescriptor.Requires</code> -Klasse	246
Die <code>ModuleDescriptor.Exports</code> -Klasse	247
Die Klasse <code>ModuleDescriptor.Opens</code>	247
Die Klasse <code>ModuleDescriptor.Provides</code>	248
Die <code>ModuleDescriptor.Version</code> -Klasse	249
Die <code>ModuleFinder</code> -Schnittstelle	250
Die <code>ModuleReader</code> -Schnittstelle	251
Durchführung von Operationen an Modulen	255
Erhalten des Moduls einer Klasse	255
Zugriff auf Ressourcen eines Moduls	255
Suche nach allen Modulen im Modulpfad	256
Erhalten von Modulinformationen	256
Zusammenfassung	261
Kapitel 10: Fortgeschrittene Themen	263
JMOD-Dateien	263
Das JMOD-Tool	264
Multi-Release-JAR-Dateien	265
Erstellen einer Multi-Release-JAR-Datei	269
Aktualisieren von Multi-Release-JAR-Dateien	270

INHALTSVERZEICHNIS

Klassenlademechanismus in JDK 9.....	270
Neue Methoden in der ClassLoader-Klasse	273
Schichten.....	274
Die Boot-Schicht	276
Konfiguration.....	277
Erstellen einer Konfiguration	278
Auflösen eines Moduls mit einer Konfiguration.....	278
Schichten erstellen	279
Die geladenen Module von einer Schicht abrufen	280
Beschreiben der Schichten zur Laufzeit	282
Aufrüstbare Module.....	283
Eigenschaften, die in den nächsten Versionen kommen	284
Zusammenfassung	285
Kapitel 11: Testen modularer Anwendungen	287
Szenarien für Unit-Tests in Java 9.....	288
Szenario 1: Junit-Testklassen und zu testende Typen befinden sich in verschiedenen Modulen	288
Szenario 2: Nur die zu testenden Typen befinden sich in einem Modul.....	289
Szenario 3: Sowohl Junit-Testklassen als auch zu testende Typen befinden sich im selben Modul	290
Die -Xmodule-Option	291
Die --patch-module-Option	291
Ein Modul patchen	293
Ausführen eines Junit-Tests, bei dem sich die Junit-Testklasse und die zu testenden Typen in getrennten Modulen.....	300
Ausführen eines Junit-Tests, bei dem die Junit-Testklasse nicht in einem ist Modul	304
Testen mit Maven	306
Zusammenfassung	309

Kapitel 12: Integration mit Werkzeugen	311
Integration mit IDEs	311
Integration mit IntelliJ IDEA	312
Integration mit Eclipse	315
Integration mit NetBeans	315
Integration mit Build-Tools	316
Integration mit Apache Maven	317
Apache Maven JDeps Plugin.....	318
Apache Maven Compiler Plugin.....	321
Rückwärtskompatibilität	323
Zusammenfassung	326