

Inhaltsverzeichnis

Vorwort	v
I Einführung und Grundlagen	1
1 Das Prinzip der Modularisierung	3
1.1 Was ist Modularisierung?	3
1.2 Was ist ein Modul?	6
1.2.1 Geheimnisprinzip und Datenkapselung	9
1.3 Modularisierung eines Systems	11
1.3.1 Entwurfsprozess für Module	11
1.3.2 Entwurfstechniken	12
1.3.3 Entwurfskriterien zur Modularisierung	15
1.3.4 Probleme bei der Modularisierung	21
1.4 Warum modularisieren?	23
1.5 Zusammenfassung	26
2 Der Weg zum Java-Modulsystem	29
2.1 Modularisierung vor Java 9	32
2.1.1 Methoden, Klassen und Komponenten	32
2.1.2 Pakete	32
2.1.3 JARs und Build-Tools	33
2.1.4 Open Services Gateway initiative (OSGi)	33
2.2 Ziele des Java-Modulsystems	33
2.2.1 Abhängigkeiten	34
2.2.2 Startup Performance	36
2.2.3 Mangelnde Sicherheit	36
2.2.4 Skalierbarkeit der Plattform	36
2.3 Zusammenfassung	37

II	Module in der Praxis	39
3	Das Java-Modulsystem	41
3.1	Das Modul	41
3.2	Abhängigkeiten und Sichtbarkeiten	46
3.2.1	Verteilter Modul-Quellcode	53
3.2.2	Transitive Abhängigkeiten	54
3.3	Services	57
3.3.1	Services vor Java 9	59
3.3.2	Services mit Modulen	61
3.3.3	ServiceLoader und Module	66
3.4	Ressourcen	73
3.4.1	Modulübergreifender und -interner Zugriff	75
3.5	Arten von Modulen	78
3.5.1	Platform Explicit Modules	79
3.5.2	Application Explicit Modules	80
3.5.3	Automatic Modules	80
3.5.4	Namensfestlegung für Automatic Modules	81
3.5.5	Open Modules	83
3.5.6	Unnamed Module	84
3.6	Reflection	86
3.7	Schichten und Klassenloader	92
3.7.1	Anlegen neuer Schichten	94
3.8	Analyse von Modulen	100
3.8.1	Visualisierung des Modulgraphen	101
3.9	Ein Blick unter die Motorhaube	103
3.9.1	Die Entstehung eines Modulgraphen	103
3.9.2	Configuration	105
3.9.3	ModuleLayer	109
3.10	Zusammenfassung	112
4	Das modularisierte JDK	115
4.1	Das JDK war ein Monolith	115
4.2	Compact Profiles	117
4.3	Die Modularisierung der Plattform	118
4.3.1	JDK-Struktur	119
4.4	Eigene modulare Laufzeit-Images erstellen	121
4.5	Zusammenfassung	124
5	Testen und Patches von Modulen	125
5.1	Testen – kurz und knapp	125
5.1.1	Validierung und Verifizierung	126
5.1.2	Testplanung und -spezifikation	127

5.1.3	Testarten	127
5.2	Black-Box-Test	128
5.3	White-Box-Test	133
5.4	Patchen	135
5.5	Zusammenfassung	139
6	Migration von Anwendungen	141
6.1	Was bedeutet Migration?	141
6.2	Fallstricke	142
6.3	Migrationsstrategien	143
6.3.1	Reine Plattform-Migration	144
6.3.2	Big-Bang-Migration	145
6.3.3	Top-down-Migration	146
6.3.4	Bottom-up-Migration	147
6.4	Beispiel für die Vorgehensweise einer Migration	149
6.5	Big Kill Switch	153
6.6	Praktisches Beispiel	154
6.6.1	Die Anwendung	154
6.6.2	Untersuchung auf Abhängigkeiten	159
6.6.3	Probleme bei der Migration vom Klassenpfad	160
6.6.4	Integration nichtmodularer Abhängigkeiten	160
6.6.5	Die Migration der Anwendung	161
6.7	Tipps für die Migration	163
6.8	Zusammenfassung	164
7	Kritik am Modulsystem	167
8	OSGi vs. Java-Modulsystem	171
8.1	Was ist OSGi?	171
8.2	OSGi in Kürze	172
8.3	Unterschiede zum Java-Modulsystem	176
8.4	Zusammenfassung	178
9	Entwicklungswerkzeuge	181
9.1	IDEs	181
9.1.1	Eclipse	181
9.1.2	NetBeans IDE	188
9.1.3	IntelliJ IDEA	194
9.2	Build-Tools	196
9.2.1	Ant	197
9.2.2	Maven	201
9.2.3	Maven und Eclipse	208
9.2.4	Gradle	214
9.3	Zusammenfassung	220

10	Ein »Real World«-Projekt	221
10.1	Eine modularisierte Anwendung.....	222
10.2	Klassischer Ansatz	222
10.2.1	Anwendungsarchitektur	223
10.2.2	Modulentwurf und Implementierung	226
10.2.3	Starten der Anwendung.....	253
10.3	Alternativer Ansatz	255
10.3.1	Anwendungsarchitektur	256
10.3.2	Modulentwurf und Implementierung	256
10.3.3	Starten der Anwendung.....	273
10.4	Vergleich beider Ansätze	275
10.5	Zusammenfassung	276
11	Weitere Modularisierungsansätze	277
11.1	Microservices	277
11.1.1	Was ist ein Microservice?	277
11.1.2	Eigenschaften von Microservices	278
11.1.3	Größe eines Microservice.....	279
11.1.4	Kommunikation	279
11.1.5	Vorteile	280
11.1.6	Nachteile	281
11.1.7	Microservices vs. Java-Module	282
11.1.8	Microservices und Java EE	283
11.1.9	Ein auf Java-Modulen basierender Microservice	286
11.1.10	Zusammenfassung	298
11.2	Container	298
11.2.1	Virtualisierung	299
11.2.2	Was ist Docker?	300
11.2.3	Docker, das modularisierte JDK und Java-Module	301
11.2.4	Ein Docker-Container mit Java-Modulen	301
11.2.5	Zusammenfassung	307
	Literaturverzeichnis	309
	Index	313