

Inhaltsverzeichnis

Teil I	Grundlagen	1
1	Application Programming Interfaces – eine Einführung	3
1.1	Eine kurze Geschichte der APIs	3
1.2	Web-APIs ab dem Jahr 2000	5
1.3	API-Definition	7
1.4	Vorteile einer API	9
1.5	Nachteile einer API	10
1.6	API als Produkt	11
1.7	Welche Strategien verfolgen Unternehmen mit APIs?	11
1.8	Zusammenfassung	12
2	Qualitätsmerkmale	13
2.1	Allgemeine Qualitätsmerkmale	13
2.2	Benutzbarkeit	14
2.2.1	Konsistent	14
2.2.2	Intuitiv verständlich	15
2.2.3	Dokumentiert	17
2.2.4	Einprägsam und leicht zu lernen	17
2.2.5	Lesbaren Code fördernd	18
2.2.6	Schwer falsch zu benutzen	20
2.2.7	Minimal	21
2.2.8	Stabil	22
2.2.9	Einfach erweiterbar	23
2.3	Zusammenfassung	23

3	Allgemeines Vorgehen beim API-Design	25
3.1	Überblick	25
3.2	Heuristiken und Trade-offs	26
3.3	Anforderungen herausarbeiten	27
3.4	Wenn Use Cases nicht ausreichen	27
3.5	Entwurf mit Szenarien und Codebeispielen	28
3.6	Spezifikation erstellen	30
3.7	Reviews und Feedback	31
3.8	Wiederverwendung	32
3.9	Zusammenfassung	33
Teil II	Java-APIs	35
4	Ausprägungen	37
4.1	Implizite Objekt-API	37
4.2	Utility-Bibliothek	40
4.3	Service	40
4.4	Framework	41
4.5	Eine Frage der Priorität	42
4.6	Zusammenfassung	42
5	Grundlagen für Java-APIs	43
5.1	Auswahl passender Namen	43
5.1.1	Klassennamen	44
5.1.2	Methodennamen	44
5.1.3	Parameternamen	47
5.1.4	Ubiquitäre Sprache	48
5.1.5	Fazit	49
5.2	Effektiver Einsatz von Typen	49
5.2.1	Semantischen Vertrag minimieren	50
5.2.2	Semantische Verletzung der Datenkapselung vermeiden	50
5.2.3	Werden Namen überschätzt?	52
5.2.4	Fazit	54
5.3	Techniken für Objektkollaboration	54
5.3.1	Tell, Don't Ask	54
5.3.2	Command/Query Separation	55
5.3.3	Law of Demeter	58
5.3.4	Platzierung von Methoden	59
5.3.5	Fazit	60

5.4	Minimale Sichtbarkeit	60
5.4.1	Packages	60
5.4.2	Klassen	61
5.4.3	Methoden	61
5.4.4	Felder	61
5.4.5	Fazit	61
5.5	Optionale Hilfsmethoden	62
5.5.1	Komfort	62
5.5.2	Utility-Klassen	62
5.5.3	Fazit	63
5.6	Optionale Rückgabewerte	63
5.6.1	Ad-hoc-Fehlerbehandlung	64
5.6.2	Null-Objekte	65
5.6.3	Ergebnisobjekte	66
5.6.4	Fazit	66
5.7	Exceptions	67
5.7.1	Ausnahmesituationen	67
5.7.2	Checked Exception versus Unchecked Exception	68
5.7.3	Passende Abstraktionen	69
5.7.4	Dokumentation von Exceptions	70
5.7.5	Vermeidung von Exceptions	71
5.7.6	Fazit	72
5.8	Objekterzeugung	72
5.8.1	Erzeugungsmuster der GoF	73
5.8.2	Statische Factory-Methode	73
5.8.3	Builder mit Fluent Interface	75
5.8.4	Praktische Anwendung der Erzeugungsmuster	76
5.8.5	Fazit	78
5.9	Vererbung	78
5.9.1	Ansätze zum Einsatz von Vererbung	79
5.9.2	Stolperfallen bei Vererbung	80
5.9.3	Bedeutung für API-Design	82
5.9.4	Fazit	83
5.10	Interfaces	83
5.10.1	Typen nachrüsten	84
5.10.2	Unterstützung für nicht triviale Interfaces	84
5.10.3	Markierungsschnittstellen	85
5.10.4	Funktionale Interfaces	85
5.10.5	Fazit	86
5.11	Zusammenfassung	86

6	Fortgeschrittene Techniken für Java-APIs	87
6.1	Fluent Interface	87
6.1.1	DSL-Grammatik	88
6.1.2	Schachteln versus Verketteten	91
6.1.3	Fluent Interface von jOOQ	91
6.1.4	Ist der Aufwand gerechtfertigt?	92
6.1.5	Fazit	92
6.2	Template-Methoden	92
6.2.1	API versus SPI	93
6.2.2	Erweiterbare Parameter	95
6.2.3	Fazit	95
6.3	Callbacks	95
6.3.1	Synchrone Callbacks	96
6.3.2	Asynchrone Callbacks	97
6.3.3	Fazit	98
6.4	Annotationen	99
6.4.1	Auswertung zum Kompilierzeitpunkt	99
6.4.2	Auswertung zur Laufzeit	101
6.4.3	Fazit	102
6.5	Wrapper-Interfaces	103
6.5.1	Proxy	103
6.5.2	Adapter	105
6.5.3	Fassade	105
6.5.4	Fazit	107
6.6	Immutability	108
6.6.1	Wiederverwendung	108
6.6.2	Thread-Sicherheit	109
6.6.3	Einfachheit	110
6.6.4	Umsetzung	110
6.6.5	Automatische Überprüfung mit dem Mutability Detector	111
6.6.6	Codegenerierung mit Immutables	112
6.6.7	Fazit	113
6.7	Thread-sichere APIs	113
6.7.1	Vorteile	113
6.7.2	Nachteile	114
6.7.3	Was bedeutet Thread-Sicherheit?	114
6.7.4	Fazit	117
6.8	Zusammenfassung	117

7	Kompatibilität von Java-APIs	119
7.1	Kompatibilitätsstufen	119
7.1.1	Code-Kompatibilität	119
7.1.2	Binäre Kompatibilität	120
7.1.3	Funktionale Kompatibilität	121
7.2	Verwandtschaftsbeziehungen	123
7.3	Design by Contract	124
7.4	Codeänderungen	126
7.4.1	Package-Änderungen	127
7.4.2	Interface-Änderungen	128
7.4.3	Klassenänderungen	129
7.4.4	Spezialisierung von Rückgabetypen	130
7.4.5	Generalisierung von Parametertypen	131
7.4.6	Generics	131
7.4.7	Ausnahmen	132
7.4.8	Statische Methoden und Konstanten	132
7.5	Praktische Techniken für API-Änderungen	133
7.6	Test Compatibility Kit	137
7.7	Zusammenfassung	139

Teil III Remote-APIs **141**

8	Grundlagen RESTful HTTP	143
8.1	REST versus HTTP	143
8.2	REST-Grundprinzipien	144
8.3	Ressourcen – die zentralen Bausteine	149
8.4	HTTP-Methoden	151
8.5	HATEOAS	156
8.6	Zusammenfassung	160
9	Techniken für Web-APIs	161
9.1	Anwendungsbeispiel: Onlineshop	161
9.2	URI-Design	171
9.3	Medientypen	175
9.4	Fehlerbehandlung	189
9.5	Versionierung	194
9.5.1	Versionsidentifikation	197
9.6	Sicherheitsmechanismen	201

9.7	Partielle Rückgaben	204
9.8	Zusammenfassung	211
10	SOAP-Webservices	213
10.1	SOAP-Grundlagen	213
10.2	WSDL-Grundlagen	216
10.3	Entwurfsansätze und -muster	219
10.4	Versionierung	226
10.5	SOAP versus REST	230
10.6	Zusammenfassung	231
11	Messaging	233
11.1	Routenplanung für Lkw-Transporte (Teil 1)	234
11.2	Message Broker	235
11.3	Produkte	238
11.4	Standards und Protokolle	242
11.5	Routenplanung für Lkw-Transporte (Teil 2)	245
11.6	Transaktionen und garantierte Nachrichtenzustellung	247
11.7	Asynchrone Verarbeitung und REST	250
11.8	Push Notifications	252
11.9	Zusammenfassung	255
Teil IV	Übergreifende Themen	257
12	Dokumentation	259
12.1	Motivation	259
12.2	Zielgruppen unterscheiden	260
12.3	Allgemeiner Aufbau	260
12.4	Beispiele	262
12.5	Dokumentation von Java-APIs	264
12.6	Dokumentation von Web-APIs	272
12.7	Zusammenfassung	282
13	Caching	283
13.1	Anwendungsfälle	283
13.2	Performance-Vorteil	284

13.3	Verdrängungsstrategien	284
13.4	Cache-Strategien für Schreibzugriffe	285
13.5	Cache-Topologien für Webanwendungen	286
13.6	HTTP-Caching	287
13.7	Zusammenfassung	292
14	Skalierbarkeit	293
14.1	Anwendungsfall	293
14.2	Grundlagen	294
14.3	Load Balancing	297
14.4	Statuslose Kommunikation	301
14.5	Skalierung von Datenbanken	303
14.6	Skalierung von Messaging-Systemen	308
14.7	Architekturvarianten	310
14.8	Zusammenfassung	312
15	Erweiterte Architekturthemen	313
15.1	Consumer-Driven Contracts	313
15.2	Backends for Frontends	316
15.3	Vernachlässigte Frontend-Architektur	319
15.4	Netflix-APIs	320
15.5	Zusammenfassung	323
16	API-Management	325
16.1	Überblick	325
16.2	Funktionen einer API-Management-Plattform	326
16.3	API-Management-Architektur	328
16.4	Open-Source-Gateways	333
16.5	Zusammenfassung	336
Anhang		337
A	Literaturverzeichnis	339
	Index	345