Vorwort

Zunächst einmal bedanke ich mich bei Ihnen, dass Sie sich für dieses Buch entschieden haben. Hierin finden Sie Informationen zu den Datenaustauschformaten XML und JSON sowie zum Zugriff auf Datenbanken mit JDBC und JPA als auch auf MongoDB. Darüber hinaus werden RESTful Webservices mit JAX-RS und Jersey behandelt. Diese für Unternehmensanwendungen wichtigen Themen möchte ich Ihnen anhand von praxisnahen Beispielen näherbringen. Dabei kommen die vielfältigen Neuerungen aus JDK 8 zum Einsatz, um die Beispiele prägnanter zu machen. Für einen fundierten Einstieg in Java 8 möchte ich Sie auf meine Bücher »Java 8 – Die Neuerungen« [9] oder alternativ »Der Weg zum Java-Profi« [8] verweisen. Beide können ergänzend, aber auch unabhängig von diesem Buch gelesen werden.

Motivation

Wenn Sie bereits komplexe Java-Applikationen für den Desktop-Bereich schreiben und sich vertraut mit der Sprache Java fühlen, dann sind Sie schon recht gut für das Berufsleben gerüstet. Allerdings kommen Sie dort früher oder später mit Datenbanken, dem Informationsaustausch basierend auf XML oder JSON und vermutlich auch verteilten Applikationen in Berührung. Darunter versteht man Programme, die auf mehreren JVMs (und gewöhnlich somit auf mehreren Rechnern) ausgeführt werden. Um zusammenzuarbeiten, müssen diese miteinander kommunizieren, wodurch ganz neue Herausforderungen, aber auch Möglichkeiten entstehen.

Vielleicht haben Sie sich bisher auf den Desktop-Bereich konzentriert und wollen nun per JDBC oder JPA mit einer Datenbank kommunizieren. Dann erhalten Sie in diesem Buch eine fundierte Einführung in die Persistenz mit Java, SQL, JDBC und JPA. Oftmals benötigen Sie aber weiteres Know-how, da die Programmanwender zunehmend anspruchsvoller werden: Neben einer gut bedienbaren Benutzeroberfläche kommt für viele Applikationen der Wunsch auf, deren Funktionalität – zumindest teilweise – auch im Netzwerk bereitzustellen. Dazu existieren vielfältige Technologien. In diesem Buch wollen wir uns auf die populären RESTful Webservices konzentrieren und mit der Programmierung einer sogenannten Client-Server-Applikation beschäftigen.

Wie Sie sehen, sind Unternehmensanwendungen ein spannendes, aber auch weitreichendes Feld, was deutlich mehr Anforderungen als reine Java-SE-Anwendungen an den Entwickler stellt. Dieses Buch gibt Ihnen einen fundierten Einstieg. Wie schon

bei meinem Buch »Der Weg zum Java-Profi« war es auch diesmal mein Ziel, ein Buch zu schreiben, wie ich es mir selbst immer als Hilfe gewünscht habe, um mich auf die Herausforderungen und Aufgaben im Berufsleben vorzubereiten.

Wer sollte dieses Buch lesen?

Dieses Buch ist kein Buch für Programmierneulinge, sondern richtet sich an all diejenigen Leser, die solides Java-Know-how besitzen und ihren Horizont auf die interessante Welt der Unternehmensanwendungen erweitern wollen. Dazu werden die dafür benötigten Themen Datenaustauschformate (XML, JSON) sowie Datenbankentwicklung (RDBMS, SQL, JDBC, JPA und auch NoSQL-DBs mit MongoDB) sowie die Kommunikation in verteilten Applikationen mit REST-Webservices (JAX-RS) vorgestellt.

Dieses Buch richtet sich im Speziellen an zwei Zielgruppen:

1. Zum einen sind dies engagierte Hobbyprogrammierer, Informatikstudenten und Berufseinsteiger, die Java als Sprache beherrschen und nun neugierig auf die zuvor genannten Themen sind.
2. Zum anderen ist das Buch für erfahrene Softwareentwickler und -architekten gedacht, die ihr Wissen ergänzen oder auffrischen wollen.

Was soll mithilfe dieses Buchs gelernt werden?

Dieses Buch zeigt und erklärt einige wesentliche Themen, die bei der Realisierung von Unternehmensapplikationen von Bedeutung sind. Sollte ein Thema bei Ihnen besonderes Interesse wecken und Sie weitere Informationen wünschen, so finden sich in den meisten Kapiteln Hinweise auf weiterführende Literatur.

Zwar ist Literaturstudium hilfreich, aber nur durch Übung und Einsatz in der Praxis können wir unsere Fähigkeiten signifikant verbessern. Deshalb ermuntere ich Sie, die gezeigten Beispiele (zumindest teilweise) durchzuarbeiten. Manchmal werde ich bei der Lösung eines Problems bewusst zunächst einen Irrweg einschlagen, um anhand der anschließend vorgestellten Korrektur die Vorteile deutlicher herauszustellen. Mit dieser Darstellungsweise hoffe ich, Ihnen mögliche Fallstricke und Lösungen aufzeigen zu können.

Des Weiteren lege ich Wert darauf, auch den kleinen, scheinbar nicht ganz so wichtigen Dingen ausreichend Beachtung zu schenken. Zum Beispiel ist es von großem Nutzen, wenn Klassen, Methoden, Attribute usw. einen sinnvollen Namen tragen.

Auch auf der Ebene des Designs lässt sich einiges falsch machen. Die Komplexität in der zu modellierenden Fachlichkeit dient häufig als Ausrede für konfuse und verwirrende Lösungen. Beim professionellen Entwickeln sollte man aber viel Wert auf klares Design legen. Grundsätzlich sollte alles möglichst einfach und vor allem gut verständlich gehalten werden, sodass eine ausgereifte und wartbare Lösung entsteht.

Sourcecode und ausführbare Programme

Da der Fokus des Buchs auf dem praktischen Nutzen und der Vorbereitung auf das Berufsleben bzw. dessen besserer Meisterung liegt, werden praxisnahe Beispiele vorgestellt. Um den Rahmen des Buchs nicht zu sprengen, stellen die Listings häufig nur Ausschnitte aus lauffähigen Programmen dar – zum besseren Verständnis sind wichtige Passagen dort mitunter fett hervorgehoben. Die in den Listings abgebildeten Sourcecode-Fragmente stehen als kompilierbare und lauffähige Programme (Gradle-Tasks) auf der Webseite zu diesem Buch `www.dpunkt.de/java-persistenz` zum Download bereit. Der Programmname bzw. der Name des ausführbaren Gradle-Tasks wird in Kapitälchenschrift, etwa FIRSTSAXEXAMPLE, angegeben.

Neben dem Sourcecode befindet sich auf der Webseite ein Eclipse-Projekt, über das sich alle Programme ausführen lassen. Idealerweise nutzen Sie dazu Eclipse 4.5 oder neuer, weil diese Version der IDE bereits Java 8 unterstützt und die Beispiele dieses Buchs immer wieder auch Funktionalitäten aus JDK 8 nutzen.

Neben dem Eclipse-Projekt wird eine Datei `build.gradle` mitgeliefert, die den Ablauf des Builds für Gradle beschreibt. Dieses Build-Tool besitzt viele Vorzüge wie die kompakte und gut lesbare Notation und vereinfacht die Verwaltung von Abhängigkeiten enorm. Gradle wird im Anhang A einführend beschrieben. Als Grundlage für spätere Ergänzungen dient folgende Datei `build.gradle`, die JUnit als Abhängigkeit definiert und trotz der Kürze schon ein vollständiges Release als `jar`-Datei namens `java-profi-db-rest.jar` erzeugt:

```
apply plugin: 'java'
apply plugin: 'eclipse'

sourceCompatibility=1.8

// create special jar containing the starter app
jar
{
    baseName = "java-profi-db-rest"

    manifest
    {
        attributes ( "Main-Class" : "de.inden.starter.ApplicationStarter" )
    }
}

repositories
{
    mavenCentral()
}

dependencies
{
    testCompile 'junit:junit:4.11'

    // Weitere Abhängigkeiten hier eintragen
}
```

Aufbau dieses Buchs

Nachdem Sie nun einen groben Überblick über den Inhalt dieses Buchs haben, möchte ich die Themen der einzelnen Kapitel kurz vorstellen.

Kapitel 1 – Einstieg in XML und JSON Weil proprietäre Formate beim Datenaustausch oftmals Probleme bereiten, spielt in heutigen Applikationen die standardisierte Darstellung von Daten eine immer größere Rolle. Kapitel 1 stellt die Datenaustauschformate XML und JSON vor, die die Interoperabilität zwischen verschiedenen Programmen erleichtern und sogar einen Austausch erlauben, wenn diese Programme in unterschiedlichen Programmiersprachen erstellt wurden.

Kapitel 2 – Einführung in Persistenz und relationale Datenbanken Dieses Kapitel stellt wichtige Grundlagen zu Datenbanken und zu SQL vor. Insbesondere wird auch auf Möglichkeiten der Transformation von Objekten in entsprechende Repräsentationen in Datenbanken eingegangen. Dabei wird vor allem auch der sogenannte Impedance Mismatch, die Schwierigkeiten bei der Abbildung von Objekten auf Tabellen einer Datenbank, thematisiert.

Kapitel 3 – Persistenz mit JDBC Wie man mit Java-Bordmitteln auf Datenbanken zugreifen kann, ist Thema von Kapitel 3. Zunächst betrachten wir JDBC als Basistechnologie und erstellen verschiedene Beispielapplikationen bis hin zum Mapping von Objekten in die Datenbank, dem sogenannten ORM (Object-Relational Mapping).

Kapitel 4 – Persistenz mit JPA Das JPA (Java Persistence API) stellt eine Alternative zu JDBC dar und erleichtert die Realisierung von Persistenzlösungen mit Java, insbesondere das ORM. In Kapitel 4 werden zunächst Grundlagen besprochen und dann gezeigt, wie sich selbst komplexere Objektgraphen mithilfe von JPA persistieren lassen.

Kapitel 5 – NoSQL-Datenbanken am Beispiel von MongoDB Neben relationalen Datenbanken gewinnen NoSQL-Datenbanken immer mehr an Bedeutung. Ein Vertreter ist MongoDB, das in Kapitel 5 behandelt wird. Neben einer Einführung in die Theorie und ersten Experimenten mit der Mongo Console schauen wir uns die Verarbeitung mit Java, im Speziellen unter Zuhilfenahme von Spring Data MongoDB, an.

Kapitel 6 – REST-Services mit JAX-RS und Jersey Kapitel 6 stellt RESTful Webservices vor, die seit geraumer Zeit im praktischen Alltag immer wichtiger werden: Viele Systeme binden die Funktionalität anderer Systeme basierend auf REST ein. Nach einer Einführung in die Thematik zeige ich einige Varianten, wie sich die Funktionalität eigener Applikationen als RESTful Webservice bereitstellen lassen.

Kapitel 7 – Entwurf einer Beispielapplikation Den Abschluss des Hauptteils dieses Buchs bildet ein Kapitel, in dem eine Beispielapplikation entwickelt wird, die eine Vielzahl der zuvor im Buch vorgestellten Technologien einsetzt. Wir folgen einer iterativ inkrementellen Vorgehensweise und sehen dabei, wie man Erweiterungen schrittweise geschickt in bestehende Applikationen integrieren kann.

Anhang A – Einführung Gradle Anhang A liefert eine Einführung in das Build-Tool Gradle, mit dem die Beispiele dieses Buchs übersetzt wurden. Mit dem vermittelten Wissen können Sie dann auch kleinere eigene Projekte mit einem Build-System ausstatten.

Anhang B – Einführung Client-Server und HTTP Im Anhang B erhalten Sie einen Einstieg in die Client-Server-Kommunikation und HTTP, weil beides für verteilte Applikationen und REST-Services von zentraler Bedeutung ist.

Anhang C – Grundlagenwissen HTML HTML und XML sind wichtige Standards, deren Kenntnis einem Java-Entwickler immer mal wieder nützlich sein kann. In diesem Anhang werden verschiedene Grundlagen zu HTML so weit vorgestellt, wie diese für das Verständnis einiger Beispiele aus diesem Buch notwendig sind.

Anhang D – Grundlagenwissen JavaScript Das in Kapitel 7 entwickelte Abschlussbeispiel verwendet mitunter etwas JavaScript, um ein interaktives Web-GUI mit Zugriffen auf REST-Services zu erstellen. Die dazu benötigten Grundlagen zur Sprache JavaScript werden in diesem Anhang behandelt.

Konventionen

Verwendete Zeichensätze

In diesem Buch gelten folgende Konventionen bezüglich der Schriftart: Neben der vorliegenden Schriftart werden wichtige Textpassagen *kursiv* oder ***kursiv und fett*** markiert. Englische Fachbegriffe werden eingedeutscht großgeschrieben, etwa Event Handling. Zusammensetzungen aus englischen und deutschen (oder eingedeutschten) Begriffen werden mit Bindestrich verbunden, z. B. Plugin-Manager. Namen von Programmen und Entwurfsmustern werden in KAPITÄLCHEN geschrieben. Listings mit Sourcecode sind in der Schrift `courier` gesetzt, um zu verdeutlichen, dass dies einen Ausschnitt aus einem Java-Programm darstellt. Auch im normalen Text wird für Klassen, Methoden, Konstanten und Parameter diese Schriftart genutzt.

Verwendete Klassen aus dem JDK

Werden Klassen des JDKs erstmalig im Text erwähnt, so wird deren voll qualifizierter Name, d. h. inklusive der Package-Struktur, angegeben: Die Klasse `String` würde dann einmal als `java.lang.String` notiert – alle weiteren Nennungen erfolgen dann ohne Angabe des Package-Namens. Diese Regelung erleichtert initial die Orientierung und ein Auffinden im JDK und zudem wird der nachfolgende Text nicht zu sehr aufgebläht. Die voll qualifizierte Angabe hilft insbesondere, da in den Listings eher selten `import`-Anweisungen abgebildet werden.

Im Text beschriebene Methodenaufrufe enthalten in der Regel die Typen der Übergabeparameter, etwa `substring(int, int)`. Sind die Parameter in einem Kontext nicht entscheidend, wird mitunter auf deren Angabe aus Gründen der besseren Lesbarkeit verzichtet – das gilt ganz besonders für Methoden mit generischen Parametern.

Klassen- und Tabellennamen

Zur besseren Unterscheidbarkeit von Objekt- und Datenbankwelt werde ich für dieses Buch als Konvention deutsche Tabellen- und Spaltennamen verwenden. Zudem werden Tabellen im Plural benannt, z. B. `Personen`. Beides hilft, die in Englisch gehaltene Objektwelt leichter von der in Deutsch repräsentierten Datenbankwelt abzugrenzen.

Verwendete Abkürzungen

Im Buch verwende ich die in der nachfolgenden Tabelle aufgelisteten Abkürzungen. Weitere Abkürzungen werden im laufenden Text in Klammern nach ihrer ersten Definition aufgeführt und anschließend bei Bedarf genutzt.

Abkürzung	Bedeutung
API	Application Programming Interface
ASCII	American Standard Code for Information Interchange
DB	Datenbank
(G)UI	(Graphical) User Interface
IDE	Integrated Development Environment
JDBC	Java Database Connectivity
JDK	Java Development Kit
JLS	Java Language Specification
JPA	Java Persistence API
JRE	Java Runtime Environment
JSON	JavaScript Object Notation
JVM	Java Virtual Machine
XML	eXtensible Markup Language

Tipps und Hinweise aus der Praxis

Dieses Buch ist mit diversen Praxistipps gespickt. In diesen werden interessante Hintergrundinformationen präsentiert oder es wird auf Fallstricke hingewiesen.

Tipp: Praxistipp

In derart formatierten Kästen finden sich im späteren Verlauf des Buchs immer wieder einige wissenswerte Tipps und ergänzende Hinweise zum eigentlichen Text.

Danksagung

Ein Fachbuch zu schreiben ist eine schöne, aber arbeitsreiche und langwierige Aufgabe. Alleine kann man eine solche Aufgabe kaum bewältigen. Daher möchte ich mich an dieser Stelle bei allen bedanken, die direkt oder indirekt zum Gelingen des Buchs beigetragen haben. Insbesondere konnte ich bei der Erstellung des Manuskripts auf ein starkes Team an Korrekturlesern zurückgreifen. Es ist hilfreich, von den unterschiedlichen Sichtweisen und Erfahrungen profitieren zu dürfen.

Zunächst einmal möchte ich mich bei Michael Kulla, der als Trainer für Java SE und Java EE bekannt ist, für sein Review vieler Kapitel und die fundierten Anmerkungen bedanken. Auch Tobias Trelle als MongoDB-Experte hat sein Know-how in das Kapitel zu NoSQL-Datenbanken eingebracht. Vielen Dank!

Merten Driemeyer, Dr. Clemens Gugenberger und Prof. Dr. Carsten Kern haben mit verschiedenen hilfreichen Anmerkungen zu einer Verbesserung beigetragen. Zudem hat Ralph Willenborg mal wieder ganz genau gelesen und so diverse Tippfehler gefunden. Vielen Dank dafür! Ein ganz besonderer Dank geht an Andreas Schöneck für die schnellen Rückmeldungen auch zu später Stunde mit wertvollen Hinweisen und Anregungen.

Schließlich möchte ich verschiedenen Kollegen meines Arbeitgebers Zühlke Engineering AG danken: Jeton Memeti, Joachim Prinzbach, Marius Reusch, Dr. Christoph Schmitz, Dr. Hendrik Schöneberg und Dr. Michael Springmann. Sie trugen durch ihre Kommentare zur Klarheit und Präzisierung bei.

Ebenso geht ein Dankeschön an das Team des dpunkt.verlags (Dr. Michael Barabas, Martin Wohlrab, Miriam Metsch und Birgit Bäuerlein) für die tolle Zusammenarbeit. Außerdem möchte ich mich bei Torsten Horn für die fundierte fachliche Durchsicht sowie bei Ursula Zimpfer für ihre Adleraugen beim Copy-Editing bedanken.

Abschließend geht ein lieber Dank an meine Frau Lilija für ihr Verständnis und die Unterstützung. Glücklicherweise musste sie beim Entstehen dieses Erweiterungsbandes einen weit weniger gestressten Autor ertragen, als dies früher bei der Erstellung meines Buchs »Der Weg zum Java-Profi« der Fall war.

Anregungen und Kritik

Trotz großer Sorgfalt und mehrfachen Korrekturlesens lassen sich missverständliche Formulierungen oder sogar Fehler leider nicht vollständig ausschließen. Falls Ihnen etwas Derartiges auffällt, so zögern Sie bitte nicht, mir dies mitzuteilen. Gerne nehme ich auch sonstige Anregungen oder Verbesserungsvorschläge entgegen. Kontaktieren Sie mich bitte per Mail unter:

michael_inden@hotmail.com

Zürich und Aachen, im April 2016
Michael Inden