Jakarta EE Cookbook

Second Edition

Practical recipes for enterprise Java developers to deliver large scale applications with Jakarta EE

Elder Moraes

Packt
www.packt.com

Jakarta EE Cookbook
Second Edition

Practical recipes for enterprise Java developers to deliver large scale applications with Jakarta EE

Elder Moraes

Packt

BIRMINGHAM - MUMBAI

Jakarta EE Cookbook
Second Edition

Commissioning Editor: Kunal Chaudhari
Acquisition Editor: Denim Pinto
Content Development Editor: Tiksha Lad
Senior Editor: Afshaan Khan
Technical Editor: Sonam Pandey
Copy Editor: Safis Editing
Project Coordinator: Francy Puthiry
Proofreader: Safis Editing
Indexer: Pratik Shirodkar
Production Designer: Shankar Kalbhor

First published: June 2019
Second edition: May 2020

Production reference: 1280520

Published by Packt Publishing Ltd.
Livery Place
35 Livery Street
Birmingham
B3 2PB, UK.

ISBN 978-1-83864-288-4

www.packt.com

To Jesus Christ, my only source of eternal life and purpose.

To my beloved wife, Erica—thanks for your love and for sharing your life with me.

To my adorable daughter, Rebeca—if this book helps a single person, maybe it could help to turn the world into a better place for you.

To the memory of my mother, Matilde, who I miss every day.

To my brother, Marco, who introduced me to this incredible world of computers and software.

To my friend and guru, Bruno "Javaman" Souza—I would probably never have written this book if I hadn't met you.

To the amazing team at SouJava—you folks really live the community thing.

To my peers at TCDB for all encouragement, tips, sharing, and feedback. Thank you!

Packt.com

Subscribe to our online digital library for full access to over 7,000 books and videos, as well as industry leading tools to help you plan your personal development and advance your career. For more information, please visit our website.

Why subscribe?

- Spend less time learning and more time coding with practical eBooks and Videos from over 4,000 industry professionals
- Improve your learning with Skill Plans built especially for you
- Get a free eBook or video every month
- Fully searchable for easy access to vital information
- Copy and paste, print, and bookmark content

Did you know that Packt offers eBook versions of every book published, with PDF and ePub files available? You can upgrade to the eBook version at www.packt.com and as a print book customer, you are entitled to a discount on the eBook copy. Get in touch with us at customercare@packtpub.com for more details.

At www.packt.com, you can also read a collection of free technical articles, sign up for a range of free newsletters, and receive exclusive discounts and offers on Packt books and eBooks.

Foreword

(Foreword from the previous edition)

It is a measure of the penetration, longevity, and quality of Java EE technology that, in 2018, my friend Elder Moraes asked me to write the foreword for his book about Java EE 8. My personal involvement with Java EE goes back to the days preceding J2EE 1.4 in 2001. Since then, I have had the great honor of leading or co-leading the community teams that have developed JavaServer Faces and, later, Servlet, two of the technologies Elder covers in this book. During that time, I tried to follow the model of servant-leader, and I think the result has been a very engaged community that has a real stake in the continued success of Java EE.

When writing this foreword, I want to focus on four Cs: **curation**, **cohesion**, **current**, and **completeness**. So much has been written about Java EE over the years, and continues to be written, that the task of writing a book, particularly one in the useful cookbook format, involves a lot of **curation**. From the range of all possible things that people are doing with Java EE, which is vast, Elder has presented a curation of what he thinks are the most useful and essential things. Elder is well positioned to decide what goes in and what stays out. Elder has been consulting and working with Java EE for nearly as long as I have, but from the more practical perspective of the user.

Technical books that follow the cookbook pattern frequently suffer from a feeling of disjointedness. Not this book. Elder has put a great deal of effort into ensuring **cohesion**. Over the years, the technologies of Java EE have sometimes been criticized for not being cohesive enough with each other. This is something Sun made a conscious effort to address, starting with Java EE 6, and which Oracle continued working on in Java EE 8. Elder has leveraged this effort to seek out and present the best way to leverage the synergy of all the technologies of Java EE 8 to maximum effect.

The world outside Java EE has continued to evolve, and this has changed the way people use Java EE dramatically. The challenge for any architect on a multiyear software effort, with a service lifetime of at least a decade, is how to keep it maintainable even while the surrounding technology landscape changes. Elder has accounted for this with two excellent chapters about microservices and Docker. These two technologies provide a great complement to the power of Java EE, but also have numerous pitfalls. Elder helps you avoid the pitfalls while getting the most out of these **current** trends.

Finally, **completeness**. Many technology cookbooks stop short of providing complete reference sort of material, but Elder goes much deeper. It's almost to the point that the term cookbook does not do this book justice. Perhaps a more correct label would be complete restaurant management with supply chain logistics and a cookbook on top. Elder covers the current popular app servers on which people are running Java EE, continuous integration and pipelines, reactive programming, and more. Coming back to the curation point, it's all there, and in depth.

I hope you have success with Java EE and with its successor, Jakarta EE from the Eclipse Foundation.

Ed Burns

Consulting Member of Technical Staff at Oracle

Specification Lead of JSF and Servlet

Contributors

About the author

Elder Moraes helps Jakarta EE developers build and deliver secure, fast, and available applications so that they are able to work on great projects. He is passionate about content sharing; he does it by speaking at international events, blogging, and writing articles. He has been working with Java since 2002 and has developed applications for different industries. As a board member at SouJava, he led the *Java EE 8 - The Next Frontier* initiative, interviewing some world-class Java EE experts.

First, I have to thank my wife and daughter, Erica and Rebeca, respectively, for all the time they allowed me to put into writing this book. It was not easy for any of us. Also, thank you to my friends, Lucas and Mari, for all the support and encouragement since day one. Last but not least, thank you to all the Packt team (Isha, Sreeja, Jason, Prajakta, and others that I haven't talked to personally). You folks rock!

About the reviewer

Deepak Vohra is a consultant and a principal member of the NuBean.com software company. He is a Sun Certified Java Programmer and Web Component Developer and has worked in the fields of XML, Java programming and Java EE for ten years.
Deepak is the co-author of the Apress book, Pro XML Development with Java Technology and is also the author of the Packt Publishing books JDBC 4.0 and Oracle JDeveloper for J2EE Development, Processing XML Documents with Oracle JDeveloper 11g, EJB 3.0 Database Persistence with Oracle Fusion Middleware 11g, and Java EE Development in Eclipse IDE, and Advanced Java EE Development with WildFly.

Packt is searching for authors like you

If you're interested in becoming an author for Packt, please visit `authors.packtpub.com` and apply today. We have worked with thousands of developers and tech professionals, just like you, to help them share their insight with the global tech community. You can make a general application, apply for a specific hot topic that we are recruiting an author for, or submit your own idea.

Table of Contents

Preface

Jakarta EE is a mature platform that's widely used around the world. It is also a standard that has evolved through the hard work of individuals, vendors, groups leaders, and communities. It has a whole market and ecosystem around it, with millions of users, which also means a big and active community that is always willing to help it move forward.

For those reasons, the purpose of this book is to meet the needs of those professionals who depend on Jakarta EE to deliver really awesome enterprise solutions, not only talking about real solutions to real problems but also showing how to implement those solutions in a practical way.

The book starts with a quick overview of what Jakarta EE is and the improvements in version 8. Then, it takes you on a hands-on journey through the most important APIs.

You will learn how to use Jakarta EE for server-side development, web services, and web applications. You will also take a look at how you can properly improve the security of your enterprise solutions.

No Jakarta EE application is good enough if it doesn't follow the standards, and for that, you can count on the Jakarta EE application servers. This book will teach you how to use the most important servers on the market and get the best that they have to offer for your project.

From an architectural point of view, the book will cover microservices, cloud computing, and containers. Also, it will not forget to give you all the tools you need to build a reactive Jakarta EE application using not only Jakarta EE features, but also Java core features such as lambdas and completable futures.

The whole Java world is all about the community, so we will also show you how community-driven professionals can improve the results of their projects and even go to higher levels in their careers.

This book was based on a concept that I call *The Five Mistakes That Keep Jakarta EE Professionals Away From Great Projects.* I am ruining my career when I don't do the following things:

- Keep myself up to date
- Know the APIs (having an overview of all of them and a mastery of the most important ones)
- Know the most commonly used Jakarta EE application servers
- Know advanced architectures
- Share what I know

So, the book is a straightforward, practical, and helpful solution to each one of these mistakes. I can say with confidence that dealing with them properly can change the careers and lives of many developers around the world. I know because they've changed mine, for good.

Who this book is for

This book is made for developers who would like to learn how to meet real enterprise application needs using Jakarta EE 8. They should be familiar with application development and need to have knowledge of at least basic Java, some minimal knowledge of Jakarta EE, the basic concepts of cloud computing, and web services.

You should want to learn how to combine a bunch of APIs in a secure and fast solution, and for this, you need to know how the APIs work and when to use each one.

If when you got this book in your hands we already have Jakarta EE 9 released, please read the *To get the most out of this book* section ahead, for directions on how to make 100% of the code of this book work without any issues.

What this book covers

`Chapter 1`, *New Features and Improvements*, explains the main changes to the Jakarta EE 8 specification and what you can do with them. It also shows the new features and briefly explores the benefits of them. All these topics are supported by code examples.

`Chapter 2`, *Server-Side Development*, deep-dives into the most important APIs and the most commonly used features for server-side development. You will go through real recipes for solving real problems.

`Chapter 3`, *Building Powerful Services with JSON and RESTful Features*, creates web services for different enterprise scenarios. You will go deep into the JAX-RS, JSON-P, and JSON-B APIs.

`Chapter 4`, *Web- and Client-Server Communication*, deals with the communication generated by web applications in a fast and reliable way using the latest Jakarta EE 8 features, such as HTTP2 and Server Push.

`Chapter 5`, *Security of Enterprise Architecture*, gives you information on various tools using the best Jakarta EE features to create secure architectures.

`Chapter 6`, *Reducing the Coding Effort by Relying on Standards*, describes the services and features that Jakarta EE application servers give to the applications they host. Those features not only enable you to rely on a standard and build your application based on it, but also allow you to write less code, as you don't need to implement features that have already been implemented by the server.

`Chapter 7`, *Deploying and Managing Applications on Major Jakarta EE Servers*, describes the use of each of the most commonly used Jakarta EE application servers on the market, giving special attention to the way you deploy and manage them.

`Chapter 8`, *Building Lightweight Solutions Using Microservices*, helps you to understand how microservice architectures work and how you can easily use Jakarta EE 8 to build microservices and/or break down your monoliths in order to implement this paradigm.Continuous delivery and continuous deployment are also described, as no successful microservice project is complete without a mature building and deployment process.

`Chapter 9`, *Using Multithreading on Enterprise Context*, describes the use of multithreading and concurrency when building enterprise applications.

`Chapter 10`, *Using Event-Driven Programming to Build Reactive Applications*, describes the use of Jakarta EE 8 and core Java to create low-latency, efficient, and high-throughput applications.

`Chapter 11`, *Rising to the Cloud – Jakarta EE, Containers, and Cloud Computing*, describes how to combine Jakarta EE and containers to run applications on the cloud.

`Appendix`, *The Power of Sharing Knowledge*, describes how the community is vital for the whole Jakarta EE ecosystem (even if you don't know about it), and looks at how you can improve your own daily work by joining the Adopt a JSR initiative.

The chapter also describes how sharing knowledge is a powerful tool for improving your career and explains what it has to do with Jakarta EE (it has everything to do with Jakarta EE!).

To get the most out of this book

You should be familiar with application development and you need to have at least basic knowledge of Java and Jakarta EE. Basic knowledge of cloud computing and web services is also assumed.

Software/hardware covered in the book	**OS requirements**
Open JDK 8 or superior	Linux/macOS/Windows
Maven 3.5	Linux/macOS/Windows
Eclipse GlassFish 5.1	Linux/macOS/Windows
Docker CE 18	Linux/macOS/Windows
Git SCM 2.16	Linux/macOS/Windows

If when trying to run any of the code in this book you got a message saying that the "address is already in use" or something like it, you probably have an other service/application using the same port that the example is trying to use. Closing it will fix this issue.

Jakarta EE 9 is on the way – what now?

By the time this second edition is released, we will just be a few months away from the release of Jakarta EE 9! Because of this, you may be wondering, "*why not publish the book when 9 is alive and kicking?*".

Good question! And the answer is equally clear: because no APIs/features will be changing in this release. So, what's changed that would justify a new release?

Because of one thing in the Jakarta EE field known as *big bang*. If you've been following Jakarta EE for a while now, you should be aware of it. If not, let me quickly explain it to you.

Due to Java EE being transferred from Oracle to Eclipse Foundation, it was rebranded to Jakarta EE, and due to this, we had the Jakarta EE 8 release. That's why we have the second edition of this book!

As part of the branding and intellectual property process, another thing needed to be changed: the `javax` namespaces.

The following examples have been taken from this very book, and all of them use the `javax` namespace:

```
import javax.ws.rs.core.Application;
import javax.ws.rs.container.AsyncResponse;
import javax.ejb.EJB;
import javax.validation.constraints.Size;
```

So, all the APIs under Jakarta EE have to change their namespaces from `javax` to `jakarta`. This same list in Jakarta EE 9 will look like this:

```
import jakarta.ws.rs.core.Application;
import jakarta.ws.rs.container.AsyncResponse;
import jakarta.ejb.EJB;
import jakarta.validation.constraints.Size;
```

Since this book was written to help you deliver the most incredible applications using Jakarta EE 8, I need to answer the following, and most important, question: *how does this change affect you and/or your code if you migrate from 8 to 9?*

The simple and quick answer is that your code/project will break. Period.

This isn't as bad as it sounds. At the end of the day, Jakarta EE has a huge community of developers around the world (probably one of the biggest communities), and most of them are developers like you that will face these same problems if there's no way out.

So, yes, you'll need to change the *imports* for all the classes in your project that use the `javax` namespace. And no, you won't need to do this manually (unless you want to).

While I'm writing this, there is already one tool that is being developed by the Apache Tomcat folks. You can try it out and follow the project for more information at `https://github.com/apache/tomcat-jakartaee-migration`.

I'm pretty sure that even more tools will become available soon. Stay tuned!

For more information about the *big bang*, you can read this amazing blog post by Mike Milinkovich: `https://eclipse-foundation.blog/2020/01/16/moving-forward-with-jakarta-ee-9/`.

Download the example code files

You can download the example code files for this book from your account at `www.packtpub.com`. If you purchased this book elsewhere, you can visit `www.packtpub.com/support` and register to have the files emailed directly to you.

You can download the code files by following these steps:

1. Log in or register at `www.packtpub.com`.
2. Select the **SUPPORT** tab.
3. Click on **Code Downloads & Errata**.
4. Enter the name of the book in the **Search** box and follow the onscreen instructions.

Once the file is downloaded, please make sure that you unzip or extract the folder using the latest version of:

- WinRAR/7-Zip for Windows
- Zipeg/iZip/UnRarX for Mac
- 7-Zip/PeaZip for Linux

The code bundle for the book is also hosted on GitHub at `https://github.com/eldermoraes/javaee8-cookbook`. In case there's an update to the code, it will be updated on the existing GitHub repository.

We also have other code bundles from our rich catalog of books and videos available at `https://github.com/PacktPublishing/`. Check them out!

Download the color images

We also provide a PDF file that has color images of the screenshots/diagrams used in this book. You can download it here: `https://static.packt-cdn.com/downloads/9781838642884_ColorImages.pdf`.

Conventions used

There are a number of text conventions used throughout this book.

`CodeInText`: Indicates code words in text, database table names, folder names, filenames, file extensions, pathnames, dummy URLs, user input, and Twitter handles. Here is an example: "Then, two key methods from `SseResource` do their work."

A block of code is set as follows:

```
<dependency>
    <groupId>javax</groupId>
    <artifactId>javaee-api</artifactId>
    <version>8.0</version>
```

```
        <scope>provided</scope>
    </dependency>
```

Any command-line input or output is written as follows:

```
Info:   destroy
```

Bold: Indicates a new term, an important word, or words that you see onscreen. For example, words in menus or dialog boxes appear in the text like this. Here is an example: "We defined the security rules and roles through our code (the program). There's another approach called **declarative security**."

Warnings or important notes appear like this.

Tips and tricks appear like this.

Sections

In this book, you will find several headings that appear frequently (*Getting ready*, *How to do it...*, *How it works...*, *There's more...*, and *See also*).

To give clear instructions on how to complete a recipe, use these sections as follows.

Getting ready

This section tells you what to expect in the recipe and describes how to set up any software or any preliminary settings required for the recipe.

How to do it...

This section contains the steps required to follow the recipe.

How it works...

This section usually consists of a detailed explanation of what happened in the previous section.

There's more…

This section consists of additional information about the recipe in order to make you more knowledgeable about the recipe.

See also

This section provides helpful links to other useful information for the recipe.

Get in touch

Feedback from our readers is always welcome.

General feedback: Email `feedback@packtpub.com` and mention the book title in the subject of your message. If you have questions about any aspect of this book, please email us at `questions@packtpub.com`.

Errata: Although we have taken every care to ensure the accuracy of our content, mistakes do happen. If you have found a mistake in this book, we would be grateful if you would report this to us. Please visit `www.packtpub.com/submit-errata`, selecting your book, clicking on the Errata Submission Form link, and entering the details.

Piracy: If you come across any illegal copies of our works in any form on the internet, we would be grateful if you would provide us with the location address or website name. Please contact us at `copyright@packtpub.com` with a link to the material.

If you are interested in becoming an author: If there is a topic that you have expertise in and you are interested in either writing or contributing to a book, please visit `authors.packtpub.com`.

Reviews

Please leave a review. Once you have read and used this book, why not leave a review on the site that you purchased it from? Potential readers can then see and use your unbiased opinion to make purchase decisions, we at Packt can understand what you think about our products, and our authors can see your feedback on their book. Thank you!

For more information about Packt, please visit `packtpub.com`.

1
New Features and Improvements

Jakarta EE 8 is a big release, desired and anticipated by the global community for many years. More than ever before, the whole platform is now even more robust, mature, and stable.

This chapter will cover the main APIs that we can highlight for Jakarta EE 8. Not that they are the only topics covered by this release—far from it—but they have a big role in the enterprise context and are worthy of a careful look inside.

Also, this chapter (as the whole book) will work perfectly with Jakarta EE 9.

In this chapter, we will cover the following recipes:

- Running your first Jakarta Bean Validation 2.0 code
- Running your first Jakarta CDI 2.0 code
- Running your first JAX-RS 2.1 code
- Running your first JSF 2.3 code
- Running your first JSON-P 1.1 code
- Running your first JSON-B 1.0
- Running your first Jakarta Servlet 4.0 code
- Running your first Jakarta Security 1.0
- Running your first MVC 1.0 code

Running your first Jakarta Bean Validation 2.0 code

Jakarta Bean Validation is a Java specification that basically helps you to protect your data. Through its API, you can validate fields and parameters, express constraints using annotations, and extend your customs validation rules.

It can be used both with Java SE and Jakarta EE.

In this recipe, you will have a glimpse of Jakarta Bean Validation 2.0. It doesn't matter whether you are new to it or are already using version 1.1; this content will help to familiarize you with some of its new features.

Getting ready

First, you need to add the right Jakarta Bean Validation dependency to your project, as follows:

```
<dependencies>
    <dependency>
        <groupId>junit</groupId>
        <artifactId>junit</artifactId>
        <version>4.12</version>
        <scope>test</scope>
    </dependency>
    <dependency>
        <groupId>org.hibernate.validator</groupId>
        <artifactId>hibernate-validator</artifactId>
        <version>6.0.15.Final</version>
    </dependency>
    <dependency>
        <groupId>org.glassfish</groupId>
        <artifactId>javax.el</artifactId>
        <version>3.0.1-b11</version>
    </dependency>
</dependencies>
```

How to do it...

We need to perform the following steps to try this recipe:

1. First, we need to create an object with some fields to be validated:

```
public class User {

    @NotBlank
    private String name;
    @Email
    private String email;
    @NotEmpty
    private List<@PositiveOrZero Integer> profileId;

    public User(String name, String email, List<Integer> profileId)
{
        this.name = name;
        this.email = email;
        this.profileId = profileId;
    }
}
```

2. Then, we create a `UserTest` class to validate those constraints:

```
public class UserTest {

    private static Validator validator;

    @BeforeClass
    public static void setUpClass() {
        validator = Validation.buildDefaultValidatorFactory()
        .getValidator();
    }

    @Test
    public void validUser() {
        User user = new User(
            "elder",
            "elder@eldermoraes.com",
            asList(1,2));

            Set<ConstraintViolation<User>> cv = validator
            .validate(user);
            assertTrue(cv.isEmpty());
    }

    @Test
```

```
    public void invalidName() {
        User user = new User(
            "",
            "elder@eldermoraes.com",
            asList(1,2));

            Set<ConstraintViolation<User>> cv = validator
            .validate(user);
            assertEquals(1, cv.size());
    }

    @Test
    public void invalidEmail() {
        User user = new User(
        "elder",

        "elder-eldermoraes_com",
        asList(1,2));

        Set<ConstraintViolation<User>> cv = validator
        .validate(user);
        assertEquals(1, cv.size());
    }

    @Test
    public void invalidId() {
        User user = new User(
            "elder",
            "elder@eldermoraes.com",
            asList(-1,-2,1,2));

            Set<ConstraintViolation<User>> cv = validator
            .validate(user);
            assertEquals(2, cv.size());
    }
}
```

After this, let's see how the recipe works.