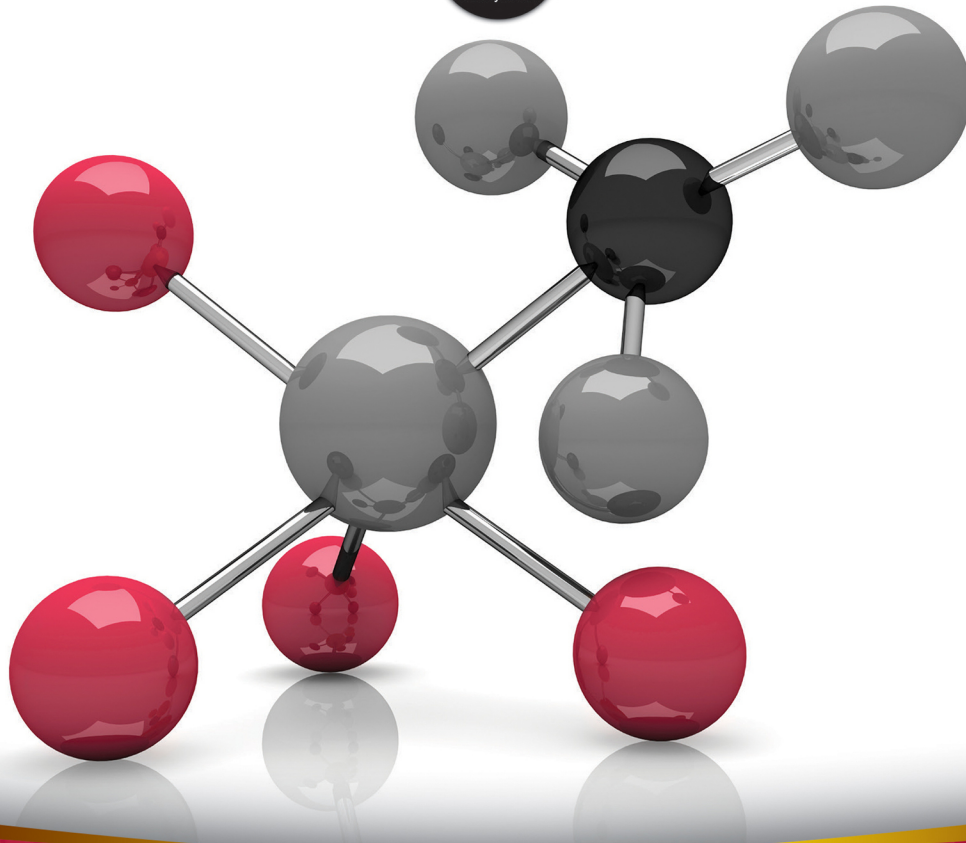


Join the discussion @ p2p.wrox.com



Wrox Programmer to Programmer™



Professional

Java® EE Design Patterns

Murat Yener, Alex Theedom

PROFESSIONAL JAVA® EE DESIGN PATTERNS

INTRODUCTION	xxv
▶ PART I INTRODUCTION TO JAVA EE DESIGN PATTERNS	
CHAPTER 1 A Brief Overview of Design Patterns	3
CHAPTER 2 The Basics of Java EE	13
▶ PART II IMPLEMENTING DESIGN PATTERNS IN JAVA EE	
CHAPTER 3 Façade Pattern	25
CHAPTER 4 Singleton Pattern	33
CHAPTER 5 Dependency Injection and CDI	49
CHAPTER 6 Factory Pattern	63
CHAPTER 7 Decorator Pattern	83
CHAPTER 8 Aspect-Oriented Programming (Interceptors)	97
CHAPTER 9 Asynchronous	113
CHAPTER 10 Timer Service	127
CHAPTER 11 Observer Pattern	139
CHAPTER 12 Data Access Pattern	153
CHAPTER 13 RESTful Web Services	165
CHAPTER 14 Model View Controller Pattern	183
CHAPTER 15 Other Patterns in Java EE	195
▶ PART III SUMMARY	
CHAPTER 16 Design Patterns: The Good, the Bad, and the Ugly	209
INDEX	215

PROFESSIONAL

Java[®] EE Design Patterns

PROFESSIONAL

Java[®] EE Design Patterns

Murat Yener
Alex Theedom



Professional Java® EE Design Patterns

Published by
John Wiley & Sons, Inc.
10475 Crosspoint Boulevard
Indianapolis, IN 46256
www.wiley.com

Copyright © 2015 by John Wiley & Sons, Inc., Indianapolis, Indiana

Published simultaneously in Canada

ISBN: 978-1-118-84341-3
ISBN: 978-1-118-84358-1 (ebk)
ISBN: 978-1-118-84345-1 (ebk)

Manufactured in the United States of America

10 9 8 7 6 5 4 3 2 1

No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, except as permitted under Sections 107 or 108 of the 1976 United States Copyright Act, without either the prior written permission of the Publisher, or authorization through payment of the appropriate per-copy fee to the Copyright Clearance Center, 222 Rosewood Drive, Danvers, MA 01923, (978) 750-8400, fax (978) 646-8600. Requests to the Publisher for permission should be addressed to the Permissions Department, John Wiley & Sons, Inc., 111 River Street, Hoboken, NJ 07030, (201) 748-6011, fax (201) 748-6008, or online at <http://www.wiley.com/go/permissions>.

Limit of Liability/Disclaimer of Warranty: The publisher and the author make no representations or warranties with respect to the accuracy or completeness of the contents of this work and specifically disclaim all warranties, including without limitation warranties of fitness for a particular purpose. No warranty may be created or extended by sales or promotional materials. The advice and strategies contained herein may not be suitable for every situation. This work is sold with the understanding that the publisher is not engaged in rendering legal, accounting, or other professional services. If professional assistance is required, the services of a competent professional person should be sought. Neither the publisher nor the author shall be liable for damages arising herefrom. The fact that an organization or Web site is referred to in this work as a citation and/or a potential source of further information does not mean that the author or the publisher endorses the information the organization or Web site may provide or recommendations it may make. Further, readers should be aware that Internet Web sites listed in this work may have changed or disappeared between when this work was written and when it is read.

For general information on our other products and services please contact our Customer Care Department within the United States at (877) 762-2974, outside the United States at (317) 572-3993 or fax (317) 572-4002.

Wiley publishes in a variety of print and electronic formats and by print-on-demand. Some material included with standard print versions of this book may not be included in e-books or in print-on-demand. If this book refers to media such as a CD or DVD that is not included in the version you purchased, you may download this material at <http://booksupport.wiley.com>. For more information about Wiley products, visit www.wiley.com.

Library of Congress Control Number: 2014946684

Trademarks: Wiley, the Wiley logo, Wrox, the Wrox logo, Programmer to Programmer, and related trade dress are trademarks or registered trademarks of John Wiley & Sons, Inc. and/or its affiliates, in the United States and other countries, and may not be used without written permission. Java is a registered trademark of Oracle America, Inc. All other trademarks are the property of their respective owners. John Wiley & Sons, Inc., is not associated with any product or vendor mentioned in this book.

*To Nilay and all my family (Semra and
Musfata Yener), for all your support and time I needed
to write this book.*

—MURAT

To Mariu, for all your support and encouragement.

—ALEX

ABOUT THE AUTHORS

MURAT YENER is a code geek and open source committer, working at Intel New Devices Group as an Android developer. He has extensive experience with developing Java, web frameworks, JavaEE, and OSGi applications, in addition to teaching courses and mentoring. Murat is an Eclipse committer and one of the initial committers of the Eclipse Libra project, he is currently working on building native and Hybrid mobile apps with HTML5 and mGWT.

Murat has been a user group leader at GDG Istanbul since 2009, organizing, participating, and speaking at events. He is also a regular speaker at JavaOne, EclipseCon, and Devvxx conferences.

LinkedIn—www.linkedin.com/in/muratyener

twitter—[@yenerm](https://twitter.com/yenerm)

blog—www.devchronicles.com

ALEX THEEDOM is a Senior Java Developer at Indigo Code Collective indigocodecollective.com (part of the E-scape Group) where he played a pivotal role in the architectural design and development of a microservice based, custom built lottery and instant win game platform.

Prior to that, he developed ATM software for an international Spanish bank and code quality analysis software for a software consultancy.

Alex is experienced with Java web application development in a diverse range of fields including finance, e-learning, lottery and software development. His passion for development has taken him to projects throughout Europe and beyond. He is a blogger at alextheedom.com and can be found helping fellow problem solvers in online forums.

LinkedIn—www.linkedin.com/in/alextheedom

Twitter—[@alextheedom](https://twitter.com/alextheedom)

Blog—www.alextheedom.com

ABOUT THE TECHNICAL EDITOR

MOHAMED SANAULLA is a Software Developer with over five years of professional experience developing software. He is currently working for India's largest e-Commerce establishment and is also a moderator on the JavaRanch Forums. When he is not working on his PC, he is busy tending to his cute little daughter. He shares his experiments and thoughts on software development at <http://blog.sanaulla.info>.

CREDITS

ASSOCIATE PUBLISHER

Jim Minatel

PROJECT EDITOR

Adaobi Obi Tulton

TECHNICAL EDITOR

Mohamed Sanoulla

PRODUCTION MANAGER

Kathleen Wisor

COPY EDITOR

Karen A. Gill

**MANAGER OF CONTENT DEVELOPMENT
AND ASSEMBLY**

Mary Beth Wakefield

MARKETING DIRECTOR

David Mayhew

MARKETING MANAGER

Carrie Sherrill

**PROFESSIONAL TECHNOLOGY AND
STRATEGY DIRECTOR**

Barry Pruett

BUSINESS MANAGER

Amy Knies

PROJECT COORDINATOR, COVER

Patrick Redmond

PROOFREADER

Nancy Carrasco

INDEXER

John Sleeva

COVER DESIGNER

Wiley

COVER IMAGE

© iStock.com/pavlen

BACKGROUND

© PhotoAlto Images/Fotosearch

ACKNOWLEDGMENTS

AS MY COAUTHOR Alex always says, we wanted to write a book we would like to own and read for ourselves. To begin with, I want to thank Alex for all his patience, hard work, and great knowledge. Without him, this book wouldn't be nearly as good.

I am grateful to Mary James, our former acquisitions editor, who contacted me about writing a book on Spring but listened to my ideas that formed the basis of this book. Without her support and guidance, this book wouldn't have become a reality. No words would be enough to thank Adaobi Obi Tulton, who patiently worked on all the details while keeping most of the schedule stresses away from us. And thanks, of course, to everyone at Wrox/Wiley who got this book on the shelves. Thanks, also, to Reza Rahman for all his encouragement.

I must thank three important people who had a huge impact on where I am in my professional life in terms of software.

First, thanks to my dad, Mustafa Yener, for buying me my first computer, a C64, at an early age while I was asking for slot cars. That computer is where I wrote my very first codes.

Second, thanks to my thesis advisor, Prof. Mahir Vardar, whom I owe all the early guidance I needed to start my career.

Finally, thanks to my life-time mentor and friend (also my ex-boss) Naci Dai, who taught me almost anything I know about being a professional software developer.

—MURAT

WE ARE VERY PROUD of this, our first book, and hope that you will get as much from reading it as we have writing it. We approached writing this with the perspective that it should be the kind of book we would buy if we hadn't written it. We have achieved that.

However, this book would not have been possible without the dedication, patience, and understanding of the many others who have contributed directly and indirectly to its creation. We would like to acknowledge the contributions made by the dedicated and experienced team at Wiley Publishing. They have stuck with us through thick and thin and believed that it was all possible. We would like to give special thanks to Mary James, our acquisitions editor, whose support made this book a reality. Thanks also to Adaobi Obi Tulton, whose patience and gentle nudges kept us on our toes and whose attention to detail saved us from tripping over ourselves. I would like to thank my coauthor, Murat Yener, for his inspiration and sense of humor that makes this book unique; and finally, but not least, I would like to thank my wife, Maria Eugenia García García, for her total support and understanding while writing this book. Thank you.

—ALEX

CONTENTS

<i>FOREWORD</i>	<i>xxiii</i>
<i>INTRODUCTION</i>	<i>xxv</i>

PART I: INTRODUCTION TO JAVA EE DESIGN PATTERNS

CHAPTER 1: A BRIEF OVERVIEW OF DESIGN PATTERNS **3**

What Is a Design Pattern?	4
How Patterns Were Discovered and Why We Need Them	5
Patterns in the Real World	5
Design Pattern Basics	6
Enterprise Patterns	7
Java to Enterprise Java	7
The Emergence of Enterprise Java Patterns	8
Design Patterns Versus Enterprise Patterns	8
Plain Old Design Patterns Meet Java EE	9
When Patterns Become Anti-Patterns	10
Summary	10
Notes	10

CHAPTER 2: THE BASICS OF JAVA EE **13**

Multitier Architecture	14
The Client Tier	15
The Middle Tier	16
Web Layer	16
Business Layer	16
The EIS Tier	18
Java EE Servers	18
The Java EE Web Profile	18
Core Principles of Java EE	19
Convention over Configuration	19
Context and Dependency Injection	20
Interceptors	21
Summary	22
Exercises	22

PART II: IMPLEMENTING DESIGN PATTERNS IN JAVA EE

CHAPTER 3: FAÇADE PATTERN	25
What Is a Façade?	26
Façade Class Diagram	27
Implementing the Façade Pattern in Plain Code	27
Implementing the Façade Pattern in Java EE	29
Façade with Stateless Beans	29
Façade with Stateful Bean	31
Where and When to Use the Façade Pattern	31
Summary	31
Exercises	32
Notes	32
CHAPTER 4: SINGLETON PATTERN	33
What Is a Singleton?	34
Singleton Class Diagram	34
Implementing the Singleton Pattern in Plain Code	35
Implementing the Singleton Pattern in Java EE	38
Singleton Beans	39
Using Singletons at Startup	39
Determining Startup Order	40
Managing Concurrency	42
Where and When to Use the Singleton Pattern	45
Summary	46
Exercises	46
Notes	47
CHAPTER 5: DEPENDENCY INJECTION AND CDI	49
What Is Dependency Injection?	50
Implementing DI in Plain Code	50
Implementing DI in Java EE	53
The @Named Annotation	54
Context and Dependency Injection (CDI)	55
CDI Versus EJB	56
CDI Beans	56
The @Inject Annotation	57
Contexts and Scope	57
Naming and EL	58
CDI Beans for Backing JSF	58
Qualifiers	59
Alternatives	59

Stereotypes	60
Other Patterns via CDI	60
Summary	61
Exercises	61
Notes	62
CHAPTER 6: FACTORY PATTERN	63
<hr/>	
What Is a Factory?	64
Factory Method	64
Implementing the Factory Method in Plain Code	66
Abstract Factory	68
Implementing the Abstract Factory in Plain Code	69
Implementing the Factory Pattern in Java EE	70
Harness the Power of CDI	76
Where and When to Use the Factory Patterns	80
Summary	80
Exercises	81
Notes	81
CHAPTER 7: DECORATOR PATTERN	83
<hr/>	
What Is a Decorator?	84
Decorator Class Diagram	85
Implementing the Decorator Pattern in Plain Code	86
Implementing the Decorator Pattern in Java EE	89
Decorators Without XML Configuration	94
Where and When to Use the Decorator Pattern	94
Summary	95
Exercises	96
Notes	96
CHAPTER 8: ASPECT-ORIENTED PROGRAMMING (INTERCEPTORS)	97
<hr/>	
What Is Aspect-Oriented Programming?	98
Implementing AOP in Plain Code	100
Aspects in Java EE, Interceptors	102
Interceptor Life Cycle	105
Default-Level Interceptors	106
Interceptor Order	107
CDI Interceptors	109
Where and When to Use Interceptors	111
Summary	112
Notes	112

CHAPTER 9: ASYNCHRONOUS	113
<hr/>	
What Is Asynchronous Programming?	114
Asynchronous Pattern	114
Implementing Asynchronous Pattern in Plain Code	116
Asynchronous Programming in Java EE	118
Asynchronous Beans	118
Asynchronous Servlets	120
Where and When to Use Asynchronous Programming	124
Summary	125
Exercises	125
Notes	126
CHAPTER 10: TIMER SERVICE	127
<hr/>	
What Is the Timer Service?	127
Implementing a Timer in Java EE	130
Automatic Timers	130
Programmatic Timers	131
Timer Expression	134
Transactions	136
Summary	137
Exercises	137
Notes	138
CHAPTER 11: OBSERVER PATTERN	139
<hr/>	
What Is an Observer?	139
Description	140
Observer Class Diagram	141
Implementing the Observer Pattern in Plain Code	142
Implementing the Observer Pattern in Java EE	144
Where and When to Use the Observer Pattern	149
Summary	150
Exercises	151
Notes	151
CHAPTER 12: DATA ACCESS PATTERN	153
<hr/>	
What Is a Data Access Pattern?	154
Data Access Class Diagram	154
Overview of the Data Access Pattern	155
Data Transfer Object Pattern	155

Java Persistence Architecture API and Object Relational Mapping	156
Implementing the Data Access Pattern in Java EE	157
Type-Safe DAO Implementation	162
Where and When to Use the Data Access Pattern	163
Summary	163
Exercises	163
Notes	163
CHAPTER 13: RESTFUL WEB SERVICES	165
<hr/>	
What Is REST?	166
The Six Constraints of REST	167
Client-Server	167
Uniform Interface	167
Stateless	168
Cacheable	168
Layered System	168
Code on Demand	168
Richardson Maturity Model of REST API	168
Level 0: The Swamp of POX (Plain Old XML)	169
Level 1: Resources	169
Level 2: HTTP Verbs	169
Level 3: Hypermedia Controls	169
Designing a RESTful API	169
Resource Naming	170
Nouns Not Verbs	170
Self-Descriptive	170
Plural Not Singular	171
HTTP Methods	171
Get	171
Post	171
Put	172
Delete	172
Rest in Action	172
The <i>users</i> noun	172
The <i>topics</i> noun and the <i>posts</i> noun	173
Implementing REST in Java EE	175
HATEOAS	178
Where and When to Use REST	180
Summary	181
Exercises	181
Notes	182

CHAPTER 14: MODEL VIEW CONTROLLER PATTERN	183
What Is the MVC Design Pattern?	184
MVC Types	185
Implementing the MVC Pattern in Plain Code	186
Implementing the MVC Pattern in Java EE	190
The FacesServlet	190
MVC Using the FacesServlet	190
Where and When to Use the MVC Pattern	193
Summary	193
Exercises	193
Note	193
CHAPTER 15: OTHER PATTERNS IN JAVA EE	195
What Are WebSockets?	195
What Is Message-Orientated Middleware	198
What Is the Microservice Architecture?	199
Monolithic Architecture	199
Scalability	200
Decomposing into Services	201
Microservice Benefits	202
Nothing in Life Is Free	203
Conclusions	204
Finally, Some Anti-Patterns	204
Uber Class	204
Lasagna Architecture	204
Mr. Columbus	205
Friends with Benefits	205
Bleeding Edge	205
Utilityman	206
Notes	206
PART III: SUMMARY	
CHAPTER 16: DESIGN PATTERNS: THE GOOD,THE BAD, AND THE UGLY	209
The Good: Patterns for Success	209
The Bad: Over and Misuse of Patterns	211
...and The Ugly	212
Summary	214
Notes	214
INDEX	215

FOREWORD

Ignorant men raise questions that wise men answered a thousand years ago

—JOHANN WOLFGANG VON GOETHE

Design patterns are our link to the past and the future. They make up a foundational language that represents well understood solutions to common problems that talented engineers before us have added to our collective knowledge base. Design patterns or blueprints exist in every engineering field in one way or another. Software development is no different. Indeed, design patterns are probably our most tangible link to engineering rather than the more organic and less regimented world of the artisan or craftsman. The art and science of design patterns was brought to the world of software engineering—and more specifically to enterprise Java—by the seminal Gang of Four (GoF) book. They have been with us ever since through our adventures in J2EE, Spring, and now modern lightweight Java EE. This is for very good reasons. Server-side Java developers tend to write the type of mission critical applications that need to stand the test of time and hence benefit the most from the discipline that design patterns represent.

It really takes a special kind of person to write a book on design patterns, let alone a book on how to utilize design patterns in Java EE applications. You require not only basic knowledge of APIs and the patterns themselves, but deep insight that can only come with hard-earned experience, as well as an innate ability to explain complex concepts elegantly. I am glad Java EE now has Murat and Alex to accomplish the mighty feat.

This book fulfills a much needed gap and fills it well. It is also very good that the book is on the cutting edge and covers Java EE 7 and not just Java EE 6 or Java EE 5. In fact many of the design patterns covered, like Singleton, Factory, Model-View-Controller (MVC), Decorator, and Observer, are now incorporated right into the Java EE platform. Others like Facade, Data Access Object (DAO), and Data Transfer Object (DTO) fit elegantly on top. Murat and Alex tackle each pattern, explain its pragmatic motivation, and discuss how it fits into Java EE.

It is an honor and a privilege to write a small opening part of this very important book that I hope will become a very useful part of every good Java EE developer's bookshelf. I hope you enjoy the book, and that it helps you write better, more satisfying enterprise Java applications.

M. REZA RAHMAN
Java EE/GlassFish Evangelist
Oracle Corporation

INTRODUCTION

THIS BOOK DISCUSSES THE CLASSIC DESIGN PATTERNS that were first mentioned in the famous book by the GoF¹ and updates them specifically for Java EE 6 and 7.

In every chapter we describe the traditional implementation of each pattern and then show how to implement it using Java EE-specific semantics.

We use full code examples to demonstrate both the traditional and Java EE implementations and color each chapter with real-life stories that show the use (or misuse) of the pattern.

We investigate the pros and cons of each pattern and examine their usages. Each chapter finishes with some exercises that challenge your understanding of the pattern in Java EE.

WHO THIS BOOK IS FOR

This book is for everyone with any level of experience. It covers almost everything about a pattern, from how it is referred to in other books, to code on basic Java implementation, to Java EE implementation, and finally real life examples of how and when to use a specific pattern. It also has real life war stories that talk about good and bad practices.

Having some basic knowledge of design patterns and Java EE will aid you as you read this book.

If you are already experienced with patterns and basic Java implementations, you may prefer to jump into Java EE implementations. Refreshing your memory and knowledge of design patterns could prove helpful.

WHAT THIS BOOK COVERS

This book covers all classical design patterns that Java EE offers as part of standard implementation, besides some new patterns. The coverage goes back to Java EE5 and is up to date for the latest version available, which is Java EE 7.

We hope this book will be a reference you will keep on your shelf for a long time.

HOW THIS BOOK IS STRUCTURED

Each chapter focuses on a design pattern. If the pattern is classical, a simple Java implementation is given after the explanation of the pattern. Each chapter offers war stories telling a good or bad real life example about the pattern focused on/in the chapter. The war story is followed by a Java EE implementation, example, and explanation. Each code sample given can be run by itself. Finally, each chapter ends with when and how to use the pattern effectively.