

@agoncal

Understanding LangChain4j

Antonio Goncalves

Foreword by Dmytro Liubarskyi

Understanding LangChain4j

LangChain for Java

Antonio Goncalves

2024-10-04

Table of Contents

Foreword	3
About the Author	4
Acknowledgments	6
Introduction	8
Where Does This Fascicle Come From?	8
Who Is This Fascicle For?	9
How Is This Fascicle Structured?	9
Conventions	11
The Sample Application	11
Downloading and Running the Code	13
Getting Help	13
Contacting the Author	13
I: First Steps	15
1. First Look at LangChain4j	16
2. Understanding LangChain for Java	18
2.1. Understanding AI Models	18
2.1.1. Generative AI	18
2.1.2. Large and Small Language Models	19
2.1.3. Training	20
2.1.4. Types of Models	20
2.1.5. Model Providers	21
2.1.6. Prompts	21
2.1.7. Roles and Messages	22
2.1.8. Hallucinations	25
2.1.9. Tokens	25
2.1.10. Context Window	26
2.1.11. Function Calling	26
2.1.12. Use Cases	27
2.2. Understanding Embeddings	27
2.2.1. Dimensions	28
2.2.2. Types of Embedding Models	29
2.2.3. Vector Databases	29
2.2.4. Semantic Search	30
2.3. Understanding Retrieval-Augmented Generation (RAG)	31
2.3.1. Fine Tuning vs RAG	32
2.3.2. Processing In-house Data	32
2.3.3. Splitting Documents	33
2.4. LangChain4j Overview	35

2.4.1. Internal Architecture of LangChain for Java	35
2.4.2. A Brief History of LangChain for Java	36
2.4.3. Other Competing Technologies	36
2.5. Summary	37
3. Getting Started	39
3.1. Developing Your First LangChain4j Application	39
3.1.1. Setting up the Maven Dependencies	40
3.1.2. The Jazz Musician Record	44
3.1.3. Invoking an OpenAI GPT Model	44
3.1.4. Getting an OpenAI Key	46
3.1.5. Executing the Code	46
3.1.6. Checking the Request and the Response	46
3.1.7. Testing the Code	48
3.1.8. Executing the Tests	50
3.2. A Technical Look at LangChain4j	52
3.2.1. Main LangChain4j Source Code Repositories	52
3.2.2. Main LangChain4j Modules	52
3.2.3. Main LangChain4j Packages	55
3.2.4. Main LangChain4j APIs	55
3.2.5. Main LangChain4j Annotations	56
3.2.6. Main LangChain4j External Dependencies	57
3.2.7. LangChain4j Configuration Files	58
3.3. Summary	58
II: AI Models	61
4. Accessing Models	62
4.1. REST APIs vs SDKs	62
4.2. Models	64
4.3. Supported Types of Models	66
4.3.1. Language Models	66
4.3.2. Chat Models	67
4.3.3. Image Models	69
4.3.4. Moderation Models	71
4.3.5. Scoring Models	73
4.4. Handling the Model's Response	74
4.4.1. Typed and Untyped Response	75
4.4.2. Streaming the Response	76
4.5. Configuring the Model's Parameters	79
4.5.1. Hallucinations	81
4.6. Supported Model Providers	82
4.6.1. Amazon Bedrock	85
4.6.2. Azure OpenAI	86

4.6.3. Google Vertex AI and Vertex AI Gemini	88
4.6.4. HuggingFace	89
4.6.5. MistralAI	90
4.6.6. Ollama	91
4.6.7. OpenAI	92
4.7. Summary	94
5. Invoking Models	96
5.1. Tokens and Context Window	96
5.1.1. Tokens	96
5.1.2. Context Window	99
5.2. Roles and Messages	100
5.2.1. Roles	100
5.2.2. Message Types	101
5.2.3. Content Types	103
5.2.4. Message Templates	104
5.3. Maintaining a Conversation	106
5.3.1. Sending Previous Messages	107
5.3.2. Chat Memory	108
5.3.3. Supported Memory Store	112
5.3.4. Chat Memory Store	112
5.4. Summary	115
6. Extending Models	117
6.1. Chains	117
6.2. Tools	118
6.3. Summary	124
III: Retrieval-Augmented Generation	126
7. Processing Documents	127
7.1. Documents	127
7.2. Parsing Documents	130
7.3. Loading Documents	132
7.4. Transforming Documents	136
7.5. Splitting Documents into Segments	137
7.6. Summary	140
8. Handling Embeddings	141
8.1. Embeddings	141
8.2. Supported Embedding Models	144
8.2.1. Remote Embedding Models	144
8.2.2. Local In-Process Embedding Models	147
8.3. Supported Embedding Stores	149
8.3.1. Azure AI Search	151
8.3.2. Elasticsearch	152

8.3.3. MongoDB	153
8.3.4. PGVector	154
8.3.5. Qdrant	155
8.3.6. In-memory Embedding Store	156
8.4. Manipulating Embeddings	156
8.4.1. Storing Embeddings	157
8.4.2. Removing Embeddings	158
8.4.3. Similarity Search	159
8.5. Summary	164
9. RAG	166
9.1. Limitation of AI Models	166
9.2. Retrieval-Augmented Generation	167
9.3. The Naive RAG Workflow	168
9.4. Implementing RAG with LangChain4j	169
9.5. Summary	173
IV: Simplifying Generative AI	174
10. AI Services	175
10.1. AI Services	175
10.2. Accessing Models	176
10.2.1. Streaming the Response	177
10.3. Using Messages and Templates	179
10.4. Maintaining a Conversation	181
10.5. Moderating Chats	182
10.6. Extending Models with Tools	185
10.6.1. Tools with Parameters	190
10.7. Getting Structured Outputs	194
10.8. Summary	197
V: Conclusion	198
11. Putting It All Together	199
11.1. Presenting the Vintage Store Chatbot Application	199
11.2. Setting up the Maven Dependencies	201
11.3. Writing the Document Ingestor	203
11.4. Writing the Vintage Store Assistant	206
11.4.1. Defining the ChatAssistant Interface	207
11.4.2. Defining the ChatService Class	208
11.4.3. Defining the Tools for Legal Documents	210
11.5. Executing the Application	211
11.5.1. Getting an OpenAI Key	211
11.5.2. Compiling the Code	211
11.5.3. Running the Qdrant Vector Database	211
11.5.4. Executing the Document Ingestor	212

11.5.5. Executing the Chat Assistant	214
11.5.6. Checking the Request and the Response	215
11.6. Summary	220
12. Summary	221
VI: Appendixes	223
Appendix A: Setting up the Development Environment on macOS 14	224
A.1. Homebrew 4.x	224
A.1.1. A Brief History of Homebrew	224
A.1.2. Installing Homebrew on macOS	224
A.1.3. Checking for Homebrew Installation	225
A.1.4. Some Homebrew Commands	225
A.2. SDKMAN! 5.x	225
A.2.1. A Brief History of SDKMAN!	225
A.2.2. Installing SDKMAN! on macOS	226
A.2.3. Checking for SDKMAN! Installation	226
A.2.4. Some SDKMAN! Commands	226
A.3. Java 17	226
A.3.1. A Brief History of Java	226
A.3.2. Installing the JDK on macOS Using SDKMAN!	227
A.3.3. Checking for Java Installation	228
A.4. Maven 3.9.x	229
A.4.1. A Brief History of Maven	229
A.4.2. Project Descriptor	229
A.4.3. Managing Artifacts	230
A.4.4. Installing Maven on macOS	231
A.4.5. Checking for Maven Installation	232
A.4.6. Some Maven Commands	232
A.5. Docker 27.x	233
A.5.1. A Brief History of Docker	233
A.5.2. Installing Docker on macOS	233
A.5.3. Checking for Docker Installation	234
A.5.4. Building, Running, Pushing and Pulling Images	237
A.5.5. Some Docker Commands	241
A.6. Testing Frameworks	242
A.6.1. JUnit 5	242
A.6.2. TestContainers 1.20.x	247
A.7. Git 2.x	249
A.7.1. A Brief History of Git	249
A.7.2. Installing Git on macOS	249
A.7.3. Checking for Git Installation	250
A.7.4. Cloning Repository	250

Appendix B: Setting up Model Providers	251
B.1. Amazon Bedrock	251
B.1.1. Getting an Amazon Bedrock Key	251
B.1.2. Accessing Amazon Bedrock with LangChain4j	253
B.2. Azure OpenAI	253
B.2.1. Getting an Azure Subscription	254
B.2.2. Getting an Azure AI Key	254
B.2.3. Accessing Azure OpenAI with LangChain4j	256
B.3. Cohere	257
B.3.1. Getting a Cohere Key	257
B.3.2. Accessing Cohere with LangChain4j	259
B.4. Hugging Face	259
B.4.1. Getting a Hugging Face Key	259
B.4.2. Accessing Hugging Face with LangChain4j	261
B.5. Ollama	261
B.5.1. Installing Ollama on macOS	262
B.5.2. Checking for Ollama Installation	263
B.5.3. Some Ollama Commands	264
B.5.4. Accessing Ollama with LangChain4j	265
B.6. OpenAI	265
B.6.1. Getting an OpenAI Key	265
B.6.2. Accessing OpenAI with LangChain4j	267
Appendix C: LangChain4j Versions	269
C.1. LangChain4j 0.35.0 (25 Sep 2024)	269
C.2. LangChain4j 0.34.0 (05 Sep 2024)	269
C.3. LangChain4j 0.33.0 (25 Jul 2024)	269
C.4. LangChain4j 0.32.0 (04 Jul 2024)	270
C.5. LangChain4j 0.31.0 (23 May 2024)	270
C.6. LangChain4j 0.30.0 (16 Apr 2024)	270
C.7. LangChain4j 0.29.1 (28 Mar 2024)	271
C.8. LangChain4j 0.29.0 (26 Mar 2024)	271
C.9. LangChain4j 0.28.0 (11 Mar 2024)	271
C.10. LangChain4j 0.27.1 (9 Feb 2024)	271
C.11. LangChain4j 0.27.0 (9 Feb 2024)	272
C.12. LangChain4j 0.26.1 (30 Jan 2024)	272
C.13. LangChain4j 0.25.0 (22 Dec 2023)	272
C.14. LangChain4j 0.24.0 (12 Nov 2023)	272
C.15. LangChain4j 0.23.0 (29 Sep 2023)	273
C.16. LangChain4j 0.22.0 (29 Aug 2023)	273
C.17. LangChain4j 0.21.0 (19 Aug 2023)	273
C.18. LangChain4j 0.20.0 (14 Aug 2023)	273

C.19. LangChain4j 0.19.0 (10 Aug 2023)	274
C.20. LangChain4j 0.18.0 (26 Jul 2023)	274
C.21. LangChain4j 0.17.0 (18 Jul 2023)	274
C.22. LangChain4j 0.16.0 (18 Jul 2023)	274
C.23. LangChain4j 0.15.0 (18 Jul 2023)	275
C.24. LangChain4j 0.14.0 (16 Jul 2023)	275
C.25. LangChain4j 0.13.0 (15 Jul 2023)	275
C.26. LangChain4j 0.11.0 (11 Jul 2023)	275
C.27. LangChain4j 0.10.0 (5 Jul 2023)	275
C.28. LangChain4j 0.9.0 (3 Jul 2023)	276
C.29. LangChain4j 0.8.0 (2 Jul 2023)	276
C.30. LangChain4j 0.7.0 (2 Jul 2023)	276
C.31. LangChain4j 0.6.0 (29 Jun 2023)	276
C.32. LangChain4j 0.5.0 (26 Jun 2023)	276
C.33. LangChain4j 0.4.0 (20 Jun 2023)	276
C.34. LangChain4j 0.3.0 (20 Jun 2023)	277
C.35. LangChain4j 0.2.0 (20 Jun 2023)	277
C.36. LangChain4j 0.1.0 (20 Jun 2023)	277
Appendix D: References	279
Appendix E: Revisions of the Fascicle	280
E.1. 2024-10-04	280
Appendix F: Resources by the Same Author	281
F.1. Fascicles	281
F.1.1. Understanding Bean Validation 3.0	281
F.1.2. Understanding JPA 3.1	282
F.1.3. Understanding LangChain4j	282
F.1.4. Understanding Quarkus 3.x	283
F.1.5. Practising Quarkus 3.x	284
F.2. Online Courses	285
F.2.1. Starting With Quarkus	285
F.2.2. Building Microservices With Quarkus	286
F.2.3. Accessing Relational Databases with Quarkus	287
F.2.4. Quarkus: Fundamentals (<i>PluralSight</i>)	287
F.2.5. Microservices: The Big Picture (<i>PluralSight</i>)	288
F.2.6. Java EE: The Big Picture (<i>PluralSight</i>)	288
F.2.7. Java EE: Getting Started (<i>PluralSight</i>)	288
F.2.8. Java EE 7 Fundamentals (<i>PluralSight</i>)	289
F.2.9. Java Persistence API 2.2 (<i>PluralSight</i>)	289
F.2.10. Contexts and Dependency Injection 1.1 (<i>PluralSight</i>)	290
F.2.11. Bean Validation 1.1 (<i>PluralSight</i>)	290
Appendix G: Printed Back Cover	292

Understanding LangChain4j

Copyright © 2018-2024 by Antonio Goncalves

All rights reserved. No part of this publication may be reproduced, distributed, or transmitted in any form or by any means, including photocopying, recording, or other electronic or mechanical methods, without the prior written permission of the publisher, except in the case of brief quotations embodied in critical reviews and certain other non-commercial uses permitted by copyright law. For permission requests, write to the publisher, addressed "*Attention: Permissions Coordinator*," at the email address below:

agoncal.fascicle@gmail.com

Trademarked names, logos, and images may appear in this fascicle. Rather than use a trademark symbol with every occurrence of a trademarked name, logo, or image, I use the names, logos, and images only in an editorial fashion and to the benefit of the trademark owner, with no intention of infringement of the trademark.

The distribution of this fascicle is made through Amazon KDP (*Kindle Direct Publishing*^[1]).

Any source code referenced by the author in this text is available to readers at <https://github.com/agoncal/agoncal-fascicle-langchain4j/tree/0.35>. This source code is available for reproduction and distribution as it uses an MIT licence^[2].

- www.antonigoncalves.org
- agoncal.teachable.com
- www.amazon.com/author/agoncal

You can find two different formats of this fascicle:

- eBook (PDF/EPUB) at <https://agoncal.teachable.com/p/ebook-understanding-langchain4j>
- Paper book (ISBN: 9798338389195) and eBooks at <https://www.amazon.com/dp/B0DHXGS1LM>

Version Date: 2024-10-04

Foreword

Dear Reader,

Welcome to this book on LangChain4j!

I'm Dmytro Liubarskyi, the developer behind LangChain4j. My passion for both Java and artificial intelligence led me to create LangChain4j, a library that connects the worlds of Generative AI and Java.

In the rapidly changing field of GenAI (Generative AI), finding comprehensive and up-to-date resources can be challenging. That's why this book stands out. It not only demonstrates the capabilities of LangChain4j but also provides an in-depth exploration of the latest GenAI concepts.

I'm honoured that Antonio has dedicated his time and expertise to this book. He is an exceptional writer, known for his ability to make complex topics easy to understand. In these pages, he makes GenAI concepts clear and practical for Java developers.

Whether you're new to AI development or already experienced, I'm confident you'll find valuable insights in these pages. I'm excited for you to explore how LangChain4j can help you build AI-enhanced applications.

Dmytro Liubarskyi

Creator and Lead Developer of LangChain4j

[@LangChain4j](#)

[1] KDP <https://kdp.amazon.com>

[2] MIT licence <https://opensource.org/licenses/MIT>

About the Author



I am a Principal Software Engineer at Microsoft living in Paris. Having been focused on Java development since the late 1990s, my career took me to many different countries and companies where I worked as a consultant. As a former employee of BEA Systems (acquired by Oracle), I developed a very early expertise on distributed systems. Today, with my role at Microsoft, I help customers to build and run their intelligent Java applications on Azure. I am particularly fond of open source and I am a member of the OSSGTP^[1] (*Open Source Solution Get Together Paris*). I love to create bonds with the community. So, I created the Paris Java User Group^[2] in 2008 and co-created Devvxx France^[3] in 2012 and Voxxed Microservices in 2018^[4].

I wrote my first book on Java EE 5^[5], in French, in 2007. I then joined the *Java Community Process* (JCP)^[6] to become an Expert Member of various *Java Specification Requests* (JSRs) (Java EE 8, Java EE 7, Java EE 6, CDI 2.0, JPA 2.0, and EJB 3.1) and wrote *Beginning Java EE 6* and *Beginning Java EE 7* with Apress^[7]. Still hooked on sharing my knowledge, I decided to then self-publish my later fascicles (*Java Persistence*, *Java Validation* and *Quarkus*) as well as online video courses (see [Appendix F](#)).

For the last decades, I have given talks at international conferences (JavaOne, Devvxx, GeeCon, and many *Java User Groups*), mainly on Java, distributed systems, microservices, cloud computing and artificial intelligence. I also wrote numerous technical papers and articles for IT websites (DevX) and IT magazines (Java Magazine, Programmez, Linux Magazine). Since 2009, I have been part of the French Java podcast called *Les Cast Codeurs*^[8].

In recognition of my expertise and all of my work for the Java community, I was elected **Java Champion**^[9].

I am a graduate of the *Conservatoire National des Arts et Métiers*^[10] (CNAM) in Paris (with an engineering degree in IT), *Brighton University*^[11] (with an MSc in object-oriented design), *Universidad del Pais Vasco*^[12] in Spain, and *UFSCar University*^[13] in Brazil (MPhil in Distributed Systems). I also taught for more than 10 years at the Conservatoire National des Arts et Métiers where I previously studied.

Follow me on X/Twitter ([@agoncal](#)) and on my blog (www.antoniongoncalves.org).

[1] OSSGTP <https://www.ossntp.org>

[2] Paris JUG <https://www.parisjug.org>

[3] Devvxx France <https://devvxx.fr>

[4] Voxxed Microservices <https://voxxeddays.com/microservices>

[5] Amazon <https://www.amazon.com/author/agoncal>

[6] JCP <https://jcp.org>

[7] Amazon <https://www.amazon.com/author/agoncal>

[8] Les Cast Codeurs <https://lescastcodeurs.com>

[9] Java Champions <https://developer.oracle.com/javachampions>

[10] CNAM <https://www.cnam.eu/site-en>

[11] Brighton University <https://www.brighton.ac.uk>