Beginning IntelliJ IDEA

Integrated Development Environment for Java Programming

Ted Hagos



Beginning IntelliJ IDEA

Integrated Development Environment for Java Programming

Ted Hagos

Apress[®]

Beginning IntelliJ IDEA: Integrated Development Environment for Java Programming

Ted Hagos Makati, Philippines

ISBN-13 (pbk): 978-1-4842-7445-3 https://doi.org/10.1007/978-1-4842-7446-0

ISBN-13 (electronic): 978-1-4842-7446-0

Copyright © 2022 by Ted Hagos

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

Trademarked names, logos, and images may appear in this book. Rather than use a trademark symbol with every occurrence of a trademarked name, logo, or image we 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 use in this publication of trade names, trademarks, service marks, and similar terms, even if they are not identified as such, is not to be taken as an expression of opinion as to whether or not they are subject to proprietary rights.

While the advice and information in this book are believed to be true and accurate at the date of publication, neither the authors nor the editors nor the publisher can accept any legal responsibility for any errors or omissions that may be made. The publisher makes no warranty, express or implied, with respect to the material contained herein.

Managing Director, Apress Media LLC: Welmoed Spahr Acquisitions Editor: Steve Anglin Development Editor: Matthew Moodie Coordinating Editor: Mark Powers

Cover designed by eStudioCalamar

Cover image by Pixabay (www.pixabay.com)

Distributed to the book trade worldwide by Apress Media, LLC, 1 New York Plaza, New York, NY 10004, U.S.A. Phone 1-800-SPRINGER, fax (201) 348-4505, e-mail orders-ny@springer-sbm.com, or visit www. springeronline.com. Apress Media, LLC is a California LLC and the sole member (owner) is Springer Science + Business Media Finance Inc (SSBM Finance Inc). SSBM Finance Inc is a **Delaware** corporation.

For information on translations, please e-mail booktranslations@springernature.com; for reprint, paperback, or audio rights, please e-mail bookpermissions@springernature.com.

Apress titles may be purchased in bulk for academic, corporate, or promotional use. eBook versions and licenses are also available for most titles. For more information, reference our Print and eBook Bulk Sales web page at http://www.apress.com/bulk-sales.

Any source code or other supplementary material referenced by the author in this book is available to readers on GitHub via the book's product page, located at www.apress.com/9781484274453. For more detailed information, please visit http://www.apress.com/source-code.

Printed on acid-free paper

For Adrianne, Stephanie, JB, and Charlie.

Table of Contents

About the Author	xi
About the Technical Reviewer	xiii
Acknowledgments	XV
Introduction	xvii
Chapter 1: Getting Started	1
Which Version to Use	1
Getting the Java Development Kit	6
Installing on macOS	7
Installing on Windows	7
Installing on Linux	7
Getting and installing IntelliJ IDEA	
Installing IntelliJ IDEA	
Configuring IntelliJ	
Key Takeaways	
Chapter 2: Creating and Running a Project	
Building a Basic Java Project	
Building a Large Project	
Key Takeaways	

TABLE OF CONTENTS

Chapter 3: Project Files	
The iml File	
The .idea Folder	
The SRC Folder	
The Out Folder	
External Libraries	
Key Takeaways	
Chapter 4: IDE Tools	
The IDE	
The Project Tool Window	41
Structure Tool Window	
Navigation Bar	
Scratch File	
ТОДО	
Problems	
Terminal	
The Main Editor Windows	
Key Takeaways	
Chapter 5: Code Navigation and Generation	
Navigation	
Search Everywhere	
Finding Actions	60
Opening Files	61
Opening Classes	
Go to Symbol	
Recent Changes and Files	
Open Target Type	
Peek to Definition	

Show Members	73
View Class Hierarchy	74
Code Generation	75
Key Takeaways	80
Chapter 6: Inspections and Intentions	
Code Inspections	
Addressing Inspections	
Inspecting Code	
Inspecting the Whole Project	
Intention Actions	
Key Takeaways	
Chapter 7: Refactoring	101
Refactoring	102
When to Refactor	104
Refactoring in IntelliJ	105
Some More Refactorings in IntelliJ	111
Extract Method	111
Move Members	113
Change Signature	117
Key Takeaways	121
Chapter 8: Live Templates	123
So What Are Live Templates?	123
Parameterized Templates	126
Showing All Available Templates	
Surround Live Templates	
Creating Your Own Templates	135
Share Templates	143
Key Takeaways	145

TABLE OF CONTENTS

Chapter 9: Debugging	
Types of Errors	
Syntax Errors	
Runtime Errors	
Logic Errors	150
Debugger	151
Step Actions	160
Breakpoints	161
Key Takeaways	
Chapter 10: Source Control	
Git	
Create Git As a Local Repository	
Adding and Committing Changes	174
Branches	177
Changes in the Changelist	
Ignore Files	
GitHub Integration	
Committing and Pushing to a Remote Repo	
Creating Gist	195
Key Takeaways	
Chapter 11: Testing	
Types of Testing	199
Unit Testing	201
Why You Should Do Unit Testing	
When to Write Tests	
When to Run Tests	
JUnit in IntelliJ	203
Testing an Actual Class	
More Examples	
Key Takeaways	

Chapter 12: JavaFX	235
A Brief History	235
Setup	237
Stages, Scenes, and Nodes	249
Hello World	251
Life Cycle of a JavaFX App	253
Main.java	254
Scene Builder	255
Building FXML Files	258
Configure IntelliJ for Scene Builder	
Opening Files in Scene Builder	
Key Takeaways	268
Index	

About the Author

Ted Hagos has been a professional developer since the late 1990s. Right now, he's chief technology officer and data protection officer of RenditionDigital International, a software development company based out of Dublin. In his more than 20 years in software development, Ted wore many hats, for example, team lead, project manager, architect, and director for development. He worked with quite a few languages and tech stacks through the years, like C#, C++, JavaScript, NodeJS, and Java — most of it in Java. He also spent time as a trainer at IBM Advanced Career Education, Ateneo ITI, and Asia Pacific College.

About the Technical Reviewer

Andres Sacco has been a professional developer since 2007, working with a variety of languages, including Java, Scala, PHP, NodeJs, and Kotlin. Most of his background is in Java and the libraries or frameworks associated with it, but since 2016, he has utilized Scala as well, depending on the situation. He is focused on researching new technologies to improve the performance, stability, and quality of the applications he develops.

Acknowledgments

Loads of thanks to those who made this book possible. Special thanks to Steve Anglin and Mark Powers. Not to forget Andres Sacco, who reviewed the book and gave me many tips for improving it.

Introduction

Welcome to *Beginning IntelliJ IDEA*. You've already downloaded IntelliJ IDEA, played around with it a bit, and maybe even used it in a small project. Now, you want to understand it a bit more so you can take advantage of the productivity tools the evangelists at JetBrains are harping about — that's why you bought this book (thank you, by the way). Well, the book aims to do just that – to get you more productive.

IDEA is a big and robust IDE. There's more than one way to get something done — sometimes it's in the toolbar, other times it's in the context menu or the main menu bar, and lots of times, the task has a keyboard shortcut. The keyboard shortcut is always the fastest. Whenever possible, I included the keyboard shortcuts for the tasks presented in the chapters — so you can improve your *keyboard-fu*.

Who This Book Is For

IntelliJ supports various programming languages, but the book is Java-centric, so it's aimed squarely at Java devs. IntelliJ is a commercial IDE, but there is a community edition that you can download and use (free of charge). I used the community edition for most parts of the book. Whenever I performed a task that required the Ultimate Edition (paid version), I indicated it in the book.

Chapter Overviews

Chapter 1 – We'll walk through the installation and setup of IntelliJ for macOS, Linux, and Windows folks.

Chapter 2 – We'll create a small project, edit our codes a bit, compile the project, and run it as well so you'll get into the programming groove.

Chapter 3 – After building a small project, we will look closer at the project structure in IntelliJ. We get to examine what's inside the **.idea** folder.

Chapter 4 – We get to see the IDE up close and personal. In this chapter, we'll explore the various Tool Windows of IntelliJ.

INTRODUCTION

Chapter 5 – We start exploring some of the productivity boosters in IntelliJ. We'll look at code generators and some important keyboard shortcuts so you can get to where you need to be (quickly).

Chapter 6 – This chapter deals with one of IntelliJ's special sauces — code inspections and intentions. If you're like me — can't be bothered to remember the correct syntax — this chapter is especially for you. You'll learn how to quickly fix syntactical problems using the quick fix shortcut.

Chapter 7 – IntelliJ has great support for code refactoring; this chapter walks us through it.

Chapter 8 – This chapter discusses more on IntelliJ's productivity boosters. This chapter discusses live templates. If you're a fan of expanding code snippets, this chapter has got your back.

Chapter 9 – You've got to deal with errors sometimes. IntelliJ's debugging facilities are top-notch, which is what this chapter is about.

Chapter 10 – IntelliJ supports a variety of source control systems. Git is very popular among devs. This chapter talks about that.

Chapter 11 – If you're a fan of unit testing — and why shouldn't you be? — this is for you. We'll walk through the steps on how to write and run unit tests using JUnit5.

Chapter 12 – JavaFX is a popular desktop UI library for Java. This chapter provides an overview on how to get started with JavaFX in IntelliJ.

Additionally, the following online-only appendixes will be available as part of the source code, which can be accessed at github.com/apress/beginning-intellij-idea.

Appendix A (JakartaEE apps) – If back-end dev is more your thing, this is for you. We'll walk through how to build some simple JakartaEE apps. This is available online.

Appendix B (Customizing IntelliJ) – IntelliJ works great out of the box. Most of the time, you don't need to mess around with it; but if you'd like to customize it to suit your taste (or some coding standards), then this appendix is for you. This is available online.

Appendix C (Tips and Tricks) – Some more developer productivity tips for you. This is available online.

CHAPTER 1

Getting Started

In this chapter, we will cover the following:

- Which version to use
- Download
- Install and configure

Which Version to Use

IDEA comes in two flavors, Ultimate and Community editions.

The Community edition (CE hereafter) is the free and open source edition of IntelliJ IDEA. This edition is probably best for you if you're a beginner programmer or coding as a weekend warrior. You can use IntelliJ without any cost. IDEA CE supports some programming languages like Java, Groovy, Kotlin, Scala, Python, Dart, HTML, etc.

The CE is not a crippled version, not by a long shot; it's a competent editor; there's also no angle nor strings attached. There are no expiration dates. You can use it for free. Forever.

The Ultimate edition is the paid option for IDEA. It does everything the CE does (of course), plus a lot more. It supports more languages, more frameworks, a lot more tools, etc. It's a lot easier to compare the two editions on a table, so let's do that now. Table 1-1 compares the features and support for Community and Ultimate editions.

CHAPTER 1 GETTING STARTED

		Ultimate edition	Community edition
Language	Java	1	1
support	Groovy	\checkmark	1
	Kotlin	\checkmark	1
	Scala (via plugin)	1	1
	Python and Jython (via plugin)	\checkmark	1
	Dart (via plugin)	\checkmark	1
	Rust (via plugin)	\checkmark	1
	HTML, XML, JSON, YAML	\checkmark	1
	XSL, XPath	\checkmark	1
	Markdown	\checkmark	1
	JavaScript, TypeScript	\checkmark	
	CoffeeScript, ActionScript	\checkmark	
	SQL	\checkmark	
	CSS, Saas, SCSS, Less, Stylus	\checkmark	
	Ruby and JRuby	\checkmark	
	PHP	\checkmark	
	Go	\checkmark	

Table 1-1. IDEA Community vs. Ultimate Editions

(continued)

		Ultimate edition	Community edition
Framework	Android	1	1
support	Swing	\checkmark	\checkmark
	JavaFX	\checkmark	\checkmark
	Spring (Spring MVC, Spring Boot, Spring Integration, Spring Security, etc.)	1	
	Spring Cloud	\checkmark	
	Java EE/Jakarta EE (JSF, JAX-RS, CDI, JPA, etc.)	\checkmark	
	Hibernate	\checkmark	
	Micronaut	1	
	Grails	1	
	GWT	\checkmark	
	Play (via plugin)	1	
	Thymeleaf, FreeMarker, Velocity	\checkmark	
	AspectJ, OSGi	1	
	Akka, SSP, Play2	1	
	Selenium	1	
	React, React Native	\checkmark	
	Angular, AngularJS	1	
	Vue.js	\checkmark	
	Ruby on Rails	1	
	Django, Flask, Pyramid	1	
	Drupal, WordPress, Laravel, Symfony	<i>√</i>	

Table 1-1. (continued)

(continued)

CHAPTER 1 GETTING STARTED

Table 1-1. (continued)

		Ultimate edition	Community edition
Build tools	Maven	1	1
	Gradle	\checkmark	\checkmark
	Ant	\checkmark	1
	sbt, Bllop, Fury	\checkmark	\checkmark
	npm	\checkmark	
	Webpack	\checkmark	
	Gulp, Grunt	\checkmark	
Integrated	Debugger	\checkmark	\checkmark
developer tools	Decompiler	\checkmark	\checkmark
	Bytecode Viewer	\checkmark	\checkmark
	Test Coverage	\checkmark	\checkmark
	Test runners (JUnit, TestNG, Spock, Cucumber, ScalaTest, spec2, etc.)	1	1
	Embedded Terminal	\checkmark	\checkmark
	Database tools/SQL	\checkmark	
	HTTPClient	\checkmark	
	Profiling Tools	\checkmark	
Version control	Git, GitHub	\checkmark	\checkmark
	Subversion	\checkmark	\checkmark
	Mercurial	\checkmark	\checkmark
	Team Foundation Server (via plugin)	\checkmark	\checkmark
	Perforce	\checkmark	

(continued)

		Ultimate edition	Community edition
Deployment	Docker, Docker Compose	1	1
	Tomcat	\checkmark	
	TomEE	\checkmark	
	GlassFish	1	
	Resin	\checkmark	
	Jetty	1	
	Virgo	1	
	JBoss, WildFly	\checkmark	
	Weblogic	1	
	WebSphere, Liberty	1	
	Kubernetes	1	
Others	Custom Themes (via plugin)	1	1
	Issue tracker integration (YouTrack, JIRA, GitHub, TFS, Lighthouse, Pivotal Tracker, Redmine, Trac, etc.)	\checkmark	1
	Diagrams (UML, Dependencies)	1	
	Dependency Structure Matrix	\checkmark	
	Detecting Duplicates	1	
	Settings synchronization via JetBrains account	\checkmark	
Customer support	Resources	lssue tracker and Community Forums	lssue tracker and Community Forums
	Support	Guaranteed	Possible
License		Commercial	Opensource, Apache 2.0

Table 1-1. (continued)

The pricing information for the IDEA Ultimate edition is at www.jetbrains.com/store.

Getting the Java Development Kit

It's best to install the Java Development Kit (JDK henceforth) Standard Edition before installing IDEA. When you do this, IDEA detects the JDK installation automatically — that will save you some time on configuration work.

You have a couple of options on which JDK to use, but the two more popular choices are

- **Oracle JDK** You can download the installers from here: www.oracle.com/java/technologies/javase-downloads.html.
- **OpenJDK** You can find the download link and installation instructions here: https://openjdk.java.net/install/.

For this book, I used the Oracle JDK. You don't have to do the same. You can use OpenJDK if that's what you prefer.

Get the appropriate installer for your OS from the preceding links I provided. Figure 1-1 shows the Oracle page for the JDK download — at the time of this writing, JDK is version 15.



Figure 1-1. Java SE download page

You'd want to click the "JDK Download" link. It's best to download the Documentation too. That will come in handy later.

Oracle will ask you to agree to a license agreement before you can proceed.

Installing on macOS

Installing JDK on macOS is very straightforward: simply double-click the downloaded DMG file, and then follow the prompts until completion. The installer takes care of updating the system path, so you don't need to perform any further action after the installation.

Installing on Windows

On Windows, double-click the downloaded installer file, and then follow the prompts until completion, just like in macOS; but unlike in macOS, you'll need to include the JDK tools and binaries in the system PATH. You will need to know how to do the following on Windows:

- 1. Include Java/bin in your OS system path
- 2. Add a CLASSPATH definition in the System Path

Note that you only need to do the preceding steps if you want to use the JDK tools from the Windows command line. You don't need to do it if you will use the JDK exclusively from an IDE like IDEA.

Installing on Linux

There is a tarball option to install Java on Linux. There's also an RPM option if you're on RHEL, Fedora, or CentOS. These are all available from the Oracle link I mentioned earlier.

Alternatively, you may install the JDK using PPA. This instruction applies to Debian and its derivatives, for example, Ubuntu, Mint, etc.

On a terminal window, type the following command:

```
sudo add-apt-repository ppa:linuxuprising/java
```

Enter your user password, as usual. Then, check for updates and install the script

sudo apt update sudo apt install oracle-java15-installer oracle-java15-set-default

When the script finishes, you'd have JDK 15 on your system.

Getting and installing IntelliJ IDEA

Before you install IDEA, ensure that your machine meets recommended requirements let's skip the minimum hardware requirements because, who are we kidding, it's next to impossible working on a machine that barely meets minimum requirements. The recommended hardware specs to install IDEA are as follows:

- **RAM** 8GB RAM (more is better).
- **Disk** SSD drive with plenty of room to spare.
- **Monitor resolution** Full HD (1920x1080); go 4K if at all possible. The more screen real-estate you can afford, the better.
- **OS** Latest 64-bit versions of Windows, macOS, or Linux (e.g., Debian, Ubuntu, or RHEL).

JetBrains recommends that you use the ToolBox app to install IntelliJ IDEA, but that's not what we will use here. For our purposes, we'll use the stand-alone method of installation.

Let's get the IDEA CE installer from www.jetbrains.com/idea/download/. Download the appropriate installer for your platform. You can install IDEA on Windows, macOS, and Linux.

Installing IntelliJ IDEA

On Windows

Download the installer. Double-click the installer to run it, and then follow the wizard to completion. During the installation, you can configure the following:

- Create a desktop shortcut.
- Add the IDEA command-line launchers to the system path. This can be handy, and I suggest you tick this box.

- Add an item **Open Folder as Project** to the system context menu (when you right-click a folder).
- Associate specific file extensions with IntelliJ IDEA to open them with a double-click.

Figure 1-2 shows the setup options.

🖳 IntelliJ IDEA Setu	p		—		\times
P	Installation Op Configure your I	otions ntelliJ IDEA installation			
Create Desktop 32-bit launch Update context Add "Open Fo	er 64-bit launcher menu	Update PATH va Add launchers			ed)
Create Associat		.kts			
Download and	d install 32-bit JetBrains Run	time			
		< Back Nex	d >	Cano	cel

Figure 1-2. IntelliJ installation options

When the setup finishes, launch IntelliJ.

On macOS

Follow these steps for macOS:

- 1. Download the DMG installer.
- 2. Double-click the installer to mount it.

- 3. Drag and drop the IntelliJ IDEA app to the /Applications folder.
- 4. In Finder, go to the /**Applications** folder, right-click IntelliJ IDEA, and then choose "Open"; macOS may ask you if you want to open the application. This may happen the first time you launch IntelliJ in macOS.

On Linux

On Linux, the binary installer comes in tarball format (ideaC-2020.3.tar.gz); download it. Then, extract it in a folder where you have "execute" permissions, like this

```
sudo tar -xzf ideaIU.tar.gz -C /opt
```

Note Do not extract the tarball over a directory with an existing IntelliJ installation, lest you want to overwrite the current app — that may cause conflicts.

Next, go to the directory where you extracted IntelliJ, and then run the idea.sh file.

Configuring IntelliJ

When you run IntelliJ for the first time, it'll need some inputs from you. First, it'll ask if you want to import some previous settings you have for IntelliJ — that is if you've installed it before. Figure 1-3 shows this prompt.



Figure 1-3. Import IntelliJ IDEA Settings

You can choose a dark or light theme in the window that follows, as shown in Figure 1-4.



Figure 1-4. Customize IntelliJ

You may continue to explore the first-time launch configuration options, or you may skip — which is what I did.

CHAPTER 1 GETTING STARTED

After that, you'll see IntelliJ's welcome screen, as shown in the following:



Key Takeaways

Not much to note here since all we did was just to set up; but if there is something to take away, I think it'll be the following:

- Make sure you've installed the JDK before you install IntelliJ.
- Ensure that your machine meets the requirements of IntelliJ before installing it. It's not a very heavy IDE (relatively, compared to its peers like Eclipse or NetBeans), but it's also not exactly light like a program editor (think of Sublime).
- In Linux, make sure to install IntelliJ in a folder where you have "execute" permissions. Installing it in the /opt folder is recommended by JetBrains, but you can install it anywhere you like.

CHAPTER 2

Creating and Running a Project

What we'll cover in this chapter is as follows:

- Creating a project
- Building it
- Running it

The essential task you need to know is creating, building, and running projects in IntelliJ. That's what we will do in this chapter.

When working with IntelliJ, you need to get used to the concept of a project because you can't do much in IntelliJ without a project structure. If you want to build an application, you need to create a project and add the source codes (and other assets) to that project.

Building a Basic Java Project

If you haven't launched IntelliJ yet, now is a good time to do so; when it's opened, you'll see the Welcome window (shown in Figure 2-1).

CHAPTER 2 CREATING AND RUNNING A PROJECT

•••	Welcome to Intelli	JIDEA
Intellij IDEA	Q Search projects	New Project Open Get from VCS
Projects		
Customize		
Plugins		
Learn IntelliJ IDEA		
		Nothing to show
\$		

Figure 2-1. Welcome to IntelliJ IDEA

The four options that are prominently displayed allow us to work on a project:

- **New Project** This option is quite simple and obvious. It will let you create a project from scratch.
- **Open or Import** This will let us point to an existing project and have IntelliJ bring in all the artifacts from that folder. In the process, IntelliJ will create a new project configuration file as it loads the project.
- **Get from VCS** If you've set up version control (which we haven't), you can use this option to load a project from repos like GitHub, Bitbucket, or a local repo.
- **Search Projects** There would be a long list in this opening screen when you've created several projects. You can use the "Search Projects" function to launch a project quickly.

Let's create a new project. Choose **New Project**. You'll see the **New Project** dialog (shown in Figure 2-2).

Java 77 Maven	Project SDK:	📜 15 java version "15.0.1"	
	Additional Lib	praries and Frameworks:	
Java FX	Groov	у	
Android IntelliJ Platform Plugin	🗌 <u> K</u> otlin/	/JVM	
Groovy			
【 Kotlin			
Empty Project			
Empty Project	Use library:	[No library selected]	Create
Empty Project		[No library selected]	Create

Figure 2-2. New Project

As you can see, IntelliJ lets us work with various projects using a couple of programming languages like Kotlin, Groovy, and Java. We'll use Java.

As you can see in Figure 2-2, IntelliJ has detected my installed JDK; in my case, it's JDK 15. If you have several JDK versions in your machine, you can click the Project SDK dropdown button (shown in Figure 2-3) to select which JDK you want to use.