



C o m m u n i t y E x p e r i e n c e D i s t i l l e d

IntelliJ IDEA Essentials

Develop better software fast with IntelliJ IDEA

Jarosław Krochmalski

[PACKT]
PUBLISHING

IntelliJ IDEA Essentials

Develop better software fast with IntelliJ IDEA

Jarosław Krochmalski



BIRMINGHAM - MUMBAI

IntelliJ IDEA Essentials

Copyright © 2014 Packt Publishing

All rights reserved. No part of this book may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, without the prior written permission of the publisher, except in the case of brief quotations embedded in critical articles or reviews.

Every effort has been made in the preparation of this book to ensure the accuracy of the information presented. However, the information contained in this book is sold without warranty, either express or implied. Neither the author, nor Packt Publishing, and its dealers and distributors will be held liable for any damages caused or alleged to be caused directly or indirectly by this book.

Packt Publishing has endeavored to provide trademark information about all of the companies and products mentioned in this book by the appropriate use of capitals. However, Packt Publishing cannot guarantee the accuracy of this information.

First published: December 2014

Production reference: 1161214

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

ISBN 978-1-78439-693-0

www.packtpub.com

Credits

Author

Jarosław Krochmalski

Project Coordinator

Shipra Chawhan

Reviewers

Scott Battaglia

Andrew C. Dvorak

Grzegorz Ligas

Jan Thomä

Proofreaders

Simran Bhogal

Stephen Copestake

Maria Gould

Ameesha Green

Paul Hindle

Commissioning Editor

Dipika Gaonkar

Indexer

Monica Ajmera Mehta

Acquisition Editors

Ellen Bishop

Sam Wood

Production Coordinator

Conidon Miranda

Content Development Editor

Govindan K

Cover Work

Conidon Miranda

Technical Editor

Aman Preet Singh

Copy Editors

Roshni Banerjee

Adithi Shetty

About the Author

Jarosław Krochmalski is a passionate software designer and developer who specializes in the financial business domain. He has over 12 years of experience in software development. He is a clean-code and software craftsmanship enthusiast. He is a Certified ScrumMaster and a fan of Agile. His professional interests include new technologies in web application development, design patterns, enterprise architecture, and integration patterns. He likes to experiment with NoSQL and cloud computing.

Jarosław has been working with IDEA since its first release and has observed the IDE grow and mature. He has been designing and developing software professionally since 2000 and has been using Java as his primary programming language since 2002. In the past, he worked for companies such as Kredyt Bank (KBC) and Bank BPS on many large-scale projects such as international money orders, express payments, and collection systems. He currently works as a consultant for the Danish company 7N and writes software for the Nykredit bank. You can reach him via Twitter at @jdkroch or by e-mail at jarek@finsys.pl.

I would like to thank my wife, Marylka, and my two boys, Wojtuś and Mati, for being patient and letting me finish the book. Without their help and understanding, this book would not have been possible.

I would like to thank all the people at Packt Publishing, especially Govindan K, Aman Preet Singh, Ellen Bishop, Richard Gall, and Sam Wood – you've made the entire writing and publishing process very smooth and straightforward. A special thanks to all the technical reviewers and proofreaders for providing me with valuable feedback from which I have learned a lot. Thank you.

Greetings to my friends at 7N, Nykredit, Kredyt Bank, and Bank BPS – I hope you enjoy reading the book as much as I enjoyed writing it.

About the Reviewers

Scott Battaglia is a senior software development engineer for Audible Inc. (<http://www.audible.com/>), which is an Amazon.com, Inc. company and the leading provider of premium digital spoken audio information. He currently leads the shared Android platform team and coaches on a variety of topics, including open source, interviewing, and Scrum. Prior to this, he was an identity management architect and senior application developer with Rutgers, the State University of New Jersey.

He actively contributed to various open source projects, including Apereo Central Authentication Service (CAS) and Inspektr, and has previously contributed to Spring Security, Apereo OpenRegistry, and Apereo uPortal. He has spoken at a variety of conferences, including Jasig, EDUCAUSE, and Spring Forward on topics such as CAS, identity management, Spring Security, and software development practices.

Grzegorz Ligas is a software developer passionate about delivering solutions that are efficient and reliable. He started his career with a small company, writing software for the manufacturing industry, and then moved to retail banking. He currently works for an investment bank in London. Working in various sectors exposed him to technologies varying from mobile to highly distributed grid applications. He's the author of, and main contributor to, the IntelliJ XQuery Support plugin whose goal is to make XQuery development a pleasure.

Jan Thomä is an IT consultant with over 15 years of experience in the IT industry. He has worked for various organizations and businesses, both small and large, including the City of Hamburg, Deutsche Telekom, and the Social Democratic Party of Germany. He has been using and endorsing IntelliJ IDEA ever since he discovered it in 2005 while searching for a truly productive and integrated development environment. He wrote and contributed to several plugins for the IntelliJ platform, including the OSGi and Leiningen plugins.

www.PacktPub.com

Support files, eBooks, discount offers, and more

For support files and downloads related to your book, please visit www.PacktPub.com.

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.PacktPub.com and, as a print book customer, you are entitled to a discount on the eBook copy. Get in touch with us at service@packtpub.com for more details.

At www.PacktPub.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.



<https://www2.packtpub.com/books/subscription/packtlib>

Do you need instant solutions to your IT questions? PacktLib is Packt's online digital book library. Here, you can search, access, and read Packt's entire library of books.

Why subscribe?

- Fully searchable across every book published by Packt
- Copy-and-paste, print, and bookmark content
- On-demand and accessible via a web browser

Free access for Packt account holders

If you have an account with Packt at www.PacktPub.com, you can use this to access PacktLib today and view nine entirely free books. Simply use your login credentials for immediate access.

Table of Contents

Preface	1
Chapter 1: Get to Know Your IDE, Fast	7
Comparing the various editions	7
Installing IntelliJ IDEA	8
An overview of the workspace	9
Tool windows	9
View modes in tool windows	11
Pinned Mode	11
Docked Mode	11
Floating Mode	12
Split Mode	12
Multiple views in tool windows	13
Navigating inside the tool window	14
Tool windows set up for a specific project	15
Editor tabs	16
Crafting your settings	17
Searching for options	17
Setting keyboard shortcuts	17
Colors and fonts	18
Picking your plugins	19
Configuration tips and tricks	21
Exporting and importing settings	21
Sharing settings	22
Tuning IntelliJ IDEA	23
Summary	24

Chapter 2: Off We Go – To the Code	25
What is a project?	26
Project structure and configuration	26
Comparison of Eclipse, NetBeans, and IntelliJ IDEA terminologies	27
The project	27
Modules	28
Folders	29
Libraries	31
Facets	33
Artifacts	35
Creating a project	37
Creating a new project from scratch	38
Importing the existing project	40
Project format	43
The directory-based format	44
The file-based format	44
The directory-based format versus the file-based format	45
Summary	45
Chapter 3: The Editor	47
An overview of the editor and setup	47
The gutter area	48
The Status bar	51
Tabs	53
Scratches	55
Scrollbar	56
Navigating in the editor	57
Navigating between files	58
Navigating within a single file	64
The Search Everywhere feature	66
The editor basics	67
Searching for and replacing text	67
Syntax-aware selection	69
Column selection mode	69
Clipboard history	70
Reformatting the code	70
Code completion	71
Language injection	73
Generating code	74
Code inspection	75
Using Live Templates	76
Postfix code completion	79

Comparing files and folders	80
Comparing files	80
Comparing folders	82
Looking for help	83
Viewing inline documentation	84
Viewing type definitions	84
Looking for usages	85
Viewing method parameters	85
Viewing the external documentation	86
Summary	87
Chapter 4: Make It Better – Refactoring	89
An overview of refactoring	89
Refactoring actions	95
Rename	95
Find and Replace Code Duplicates	96
Copy	96
Move	97
Move Instance Method	99
Safe Delete	99
Change Signature	100
Type Migration	101
Make Static	101
Convert to Instance Method	101
Extract refactorings	102
Extract Variable	102
Extract Constant	102
Extract Field	103
Extract Parameter	104
Introduce Parameter Object	105
Extract Method	105
The Extract Method object	106
Delegate	107
Extract Interface	109
Extract Superclass	110
Inline	110
Remove Middleman	112
Wrap Return Value	112
Invert Boolean	113
Pull Members Up or Push Members Down	113
Replace Inheritance With Delegation	113

Convert Anonymous Class to Inner	114
Encapsulate Fields	115
Replace Constructor with Factory Method / Builder	116
Generify	116
Summary	118
Chapter 5: Make It Happen – Running Your Project	119
A temporary configuration	120
The permanent configuration	121
The Run/Debug configuration for a Java application	122
Creating a Tomcat server local configuration	128
The Node.js configuration	133
Configuration defaults	134
Sharing the configuration	135
Running	136
Summary	138
Chapter 6: Building Your Project	139
Editing Maven settings	140
The Maven tool window	144
Running Maven goals	144
Using Gradle	148
Executing Gradle tasks	150
Summary	151
Chapter 7: Red or Green? Test Your Code	153
Enabling the testing plugins	154
Creating the test	156
Creating a run/debug configuration for the test	160
Running or debugging the test	164
Keyboard shortcuts	170
Summary	170
Chapter 8: Squash'em – The Debugger	171
Debugger settings	171
Setting up the JavaScript debugger	178
Managing breakpoints	180
Starting the debugger	191
The Debug tool window	193
Inspecting variables and evaluating expressions	196
Debugger actions	203
Keyboard shortcuts summary	206
Summary	206

Chapter 9: Working with Your Team	207
Enabling version control	207
Configuring version control	210
Working with version control	217
Changelists	217
Adding files to version control	218
Committing files	220
Getting changes from the repository	224
Browsing the changes	226
Reverting the local changes	227
Using the difference viewer	228
Displaying the history	231
The log viewer	233
Quickly executing VCS actions	234
Keyboard shortcuts	236
Summary	236
Chapter 10: Not Enough? Extend It	237
Setting up the environment and project	238
Developing the plugin functionality	241
Deploying and publishing	247
Summary	251
Index	253

Preface

The first version of IntelliJ IDEA was released in January 2001. It is a mature, integrated development environment (IDE), designed to help you in the coding process, and supports a large number of different frameworks, tools, and targets. It works with multiple programming languages. It now includes full support for Java 8 and Java EE 7.

The key objective of IntelliJ IDEA is to increase and assist developer productivity. Whether you develop in Java, Scala, or PHP, or make the frontend using HTML and JavaScript, IntelliJ IDEA's smart and relevant suggestions and code completion, on-the-fly code analysis, and respectable refactoring tools will support you in every step.

When you are migrating from NetBeans or Eclipse, you will quickly see that IntelliJ IDEA is different because it *understands the context*. The IDE knows where you are in the editor and reacts accordingly; you will be surprised at how smart IntelliJ IDEA behaves.

This tool is a generic workhorse rather than a strict Java IDE. In this book, you will learn how to make IntelliJ IDEA work for you and get your job done in the most efficient and pleasant way.

What this books covers

Although the book describes the latest version of IntelliJ IDEA - 14, most of the concepts will also work on the previous revision of the IDE.

Chapter 1, Get to Know Your IDE, Fast, is a very concise note on editions comparison, requirements and installing IntelliJ IDEA in Windows, OSX, and Linux. This chapter guides you through the main workspace and show you ways to customize it for different tasks, presenting briefly the most useful plugins, IDE settings, and configuration tips.

Chapter 2, Off We Go – To the Code, describes the process of setting up a new project or importing an existing one. The chapter explains terminology differences with NetBeans and Eclipse and presents the concept of modules and artifacts.

Chapter 3, The Editor, describes the core of IntelliJ IDEA – the editor. In this chapter, you use state-of-the-art code completion, templates, and other great IntelliJ IDEA features. This chapter shows how to set up the editor and gives you some productivity tips.

Chapter 4, Make It Better – Refactoring, presents the powerful refactoring toolset of IntelliJ IDEA. You are guided through the most useful refactoring techniques.

Chapter 5, Make It Happen – Running Your Project, covers configuring the runtime environment for your project. We also talk about adding run configurations, either on the server or standalone. This chapter focusses not only on Java, but on other technologies such as Node.js as well.

Chapter 6, Building Your Project, focusses on building a project. You use IntelliJ IDEA's own build system, and Maven and Gradle integration as well.

Chapter 7, Red or Green? Test Your Code, is all about unit testing in IntelliJ IDEA. We focus on setting IntelliJ IDEA up specifically to run tests. You create JUnit and TestNG run configurations and then run and debug the tests. Then, you are given a brief overview of the test runner windows, useful settings, and option suggestions.

Chapter 8, Squash'em – The Debugger, focusses on the IntelliJ IDEA debugger. You get familiar with the debugger tool window and debugger options. We look under the hood – evaluating expressions, using watches, conditional breakpoints, and other debugger features. We also talk briefly about remote debugging.

Chapter 9, Working with Your Team, This chapter is all about version control, and managing change lists and tasks. There is a brief description on how to set up VCS integration, with the main focus on Git. This chapter describes integration with popular bug trackers, such as JIRA and YouTRACK.

Chapter 10, Not Enough? Extend It, describes briefly the plugin architecture of IntelliJ IDEA. We talk about possibilities and develop a simple plugin, so that you have knowledge of how to extend the IDE. You are also presented with useful links and resources to develop your knowledge even further.

What you need for this book

You will need a Mac or PC, running OS X, MS Windows, or Linux, to be able to set up and run IntelliJ IDEA. To learn the presented features, you will also need the tool itself, of course. You can use the free 30-day trial of Ultimate Edition or use the Community Edition, which is available free of charge. We will flag the differences and let you know what features are not available in the free version of the IDE. You can read how to get it in *Chapter 1, Get to Know Your IDE, Fast*.

Who this book is for

This book is a fast-paced introduction to IntelliJ IDEA and is aimed at users who want to learn the essentials of the new IDE in a nimble and efficient way.

Conventions

In this book, you will find a number of text styles that distinguish between different kinds of information. Here are some examples of these styles and an explanation of their meaning.

Code words in text, database table names, folder names, filenames, file extensions, pathnames, dummy URLs, user input, and Twitter handles are shown as follows: "If you keep getting `OutOfMemoryError` in `PermGen` space exceptions, try to change the `-XX:MaxPermSize` setting."


A block of code is set as follows:


```
private boolean isValid(int a) {  
    return a > 15 && a < 100;  
}
```

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

```
git clone git://git.jetbrains.org/idea/community.git idea
```

New terms and **important words** are shown in bold. Words that you see on the screen, for example, in menus or dialog boxes, appear in the text like this: "To do this, first download the IntelliJ Configuration Server plugin, using the Plugins page of the **Settings** dialog box."

[ Warnings or important notes appear in a box like this.]

[ Tips and tricks appear like this.]

Reader feedback

Feedback from our readers is always welcome. Let us know what you think about this book – what you liked or disliked. Reader feedback is important for us as it helps us develop titles that you will really get the most out of.

To send us general feedback, simply e-mail feedback@packtpub.com, and mention the book's title in the subject of your message.

If there is a topic that you have expertise in and you are interested in either writing or contributing to a book, see our author guide at www.packtpub.com/authors.

Customer support

Now that you are the proud owner of a Packt book, we have a number of things to help you to get the most from your purchase.

Errata

Although we have taken every care to ensure the accuracy of our content, mistakes do happen. If you find a mistake in one of our books – maybe a mistake in the text or the code – we would be grateful if you could report this to us. By doing so, you can save other readers from frustration and help us improve subsequent versions of this book. If you find any errata, please report them by visiting <http://www.packtpub.com/submit-errata>, selecting your book, clicking on the **Errata Submission Form** link, and entering the details of your errata. Once your errata are verified, your submission will be accepted and the errata will be uploaded to our website or added to any list of existing errata under the Errata section of that title.

To view the previously submitted errata, go to <https://www.packtpub.com/books/content/support> and enter the name of the book in the search field. The required information will appear under the **Errata** section.

Piracy

Piracy of copyrighted material on the Internet is an ongoing problem across all media. At Packt, we take the protection of our copyright and licenses very seriously. If you come across any illegal copies of our works in any form on the Internet, please provide us with the location address or website name immediately so that we can pursue a remedy.

Please contact us at copyright@packtpub.com with a link to the suspected pirated material.

We appreciate your help in protecting our authors and our ability to bring you valuable content.

Questions

If you have a problem with any aspect of this book, you can contact us at questions@packtpub.com, and we will do our best to address the problem.

1

Get to Know Your IDE, Fast

In this chapter, we will compare IntelliJ IDEA editions and licenses, install the tool, and quickly introduce the main workspace. IntelliJ IDEA comes with many settings; it is not possible to cover all of them in one book so we will focus on the most important ones. We will cover the following topics in this chapter:

- Comparing the various editions
- Installing the tool
- Workspace overview
- IDE settings
- Configuration tips and tricks

Comparing the various editions

IntelliJ IDEA is available as a free Community Edition and full-fledged Ultimate Edition. From the licensing point of view, the good thing is you can use both editions to develop the software you want to sell. It is worth mentioning that the new Android Studio that is used for the development of mobile Android applications is also based on IntelliJ IDEA.

The detailed comparison table can be found on the JetBrains website: http://www.jetbrains.com/idea/features/editions_comparison_matrix.html. To cut a long story short, there are many features missing in the Community Edition, but there are some workarounds available if you look close enough. For example, when you want to use Tomcat or Jetty servers in the Community Edition, you can use Maven plugins to run and debug your web applications freely. We will discuss this in *Chapter 5, Make It Happen – Running Your Project*.


You can use the Community Edition to develop applications using many frameworks such as Play, Struts, or Spring. It's all Java, after all. The IDE will not assist you in that. Most of the configuration hints, warnings, autocompletion, and runtime configuration features will be unavailable.

The Ultimate Edition, on the other hand, is the full-featured commercial IDE. You have the full support of almost all of the modern frameworks and application servers. The IDE will assist you by providing code completion, hints, and diagrams. The language support in this edition is also more comprehensive; you will get HTML and scripting languages analysis available on the fly, for example.

Apart from the provided features, the Ultimate Edition can be categorized based on the license. Depending on your needs, you can purchase any of the following licenses:

- **Commercial license:** IntelliJ IDEA can be used by any developer in your company but the total number of concurrent users cannot exceed the number of purchased licenses.
- **Personal license:** IntelliJ IDEA can be used only by the person who purchased it. You can use it on as many computers as you own, as long as you are the only user. The Personal license, of course, can also be used to develop commercial products.

Additionally, there are some licensing options and discounts based on the target audience, for startups, students, and teachers, for education or training, and finally, for open source projects.


 When you decide to buy the Ultimate Edition, sometimes it is wise to wait till the holidays, for example, Christmas or Easter. The JetBrains team usually provides some discounts on their products then.

Installing IntelliJ IDEA

Installing IntelliJ IDEA is straightforward. Perform the following steps:

1. Go to <http://www.jetbrains.com/idea/download/>.
2. Pick the OS version and edition of your choice.

3. After opening the downloaded installation package in MS Windows, you should see the installation wizard. In Mac OS, double-click on the downloaded .dmg file and then just drag IntelliJ IDEA to the Applications folder.

 When you install IntelliJ IDEA over an existing installation, the installation wizard will ask if you want to import settings from the previous set up. Don't worry, your settings will be preserved.

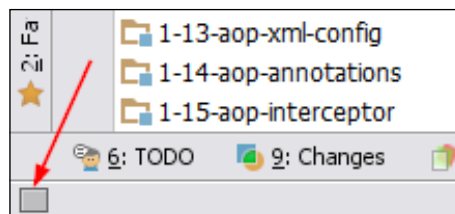
During the first startup, IntelliJ IDEA will ask you which plugins should be enabled by default. Usually, it's best to enable only what you need, so the IDE loads and works faster with fewer plugins enabled. Don't worry if you don't know what to select; you can always change your mind later by editing the IDE settings. On the first startup, you will not be able to see the workspace without the project opened. While we will go through the details of creating the project in the next chapter, you can now just create the basic Java project by choosing **New Project** from the **File** menu, selecting **Java**, and proceeding with the **New Project** wizard by clicking on **Next** a couple of times.

An overview of the workspace

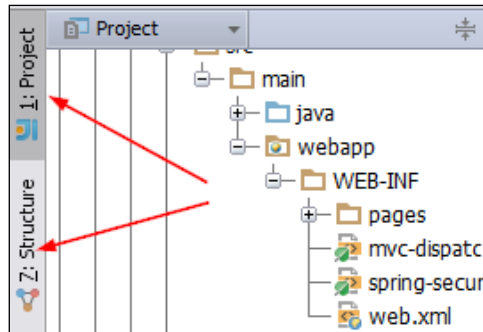
Basically, the workspace in IntelliJ IDEA consists of the main editor with tabs, menus, and many tool windows.

Tool windows

The tool windows are hidden, by default, in Version 13 and later of IDEA. You can toggle them on and off using the button in the bottom-left corner of the screen:



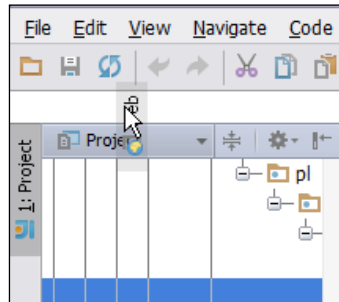
Tool windows are those little "tabs" visible at the edges of the workspace. These edges are called tool window bars, as shown in the following screenshot:



Some of the tool windows are always available, such as **Project** or **Structure**, while some of them are available only when the corresponding plugins are enabled.

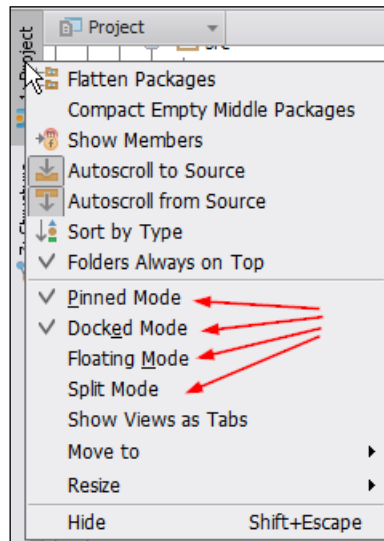
You can arrange the order of the tool windows by dragging them with your mouse. You can drag the tool window to other screen edges as well.

There's a fourth tool window bar available at the top of the screen, which is hidden. Just drag any tool window to the top of your workspace to use it, as shown in the following screenshot:



View modes in tool windows

The tool windows have a context menu available when you right-click on them. The context menu contains items specific to a particular tool window and some possible view modes, as shown in the following screenshot:



Pinned Mode

The pinned tool window will stay open even when it becomes inactive by losing focus. You may prefer to have the **Project** tool window pinned to have a constant overview of the project structure. Only docked windows can be pinned. On the other hand, you can keep the project view closed almost all the time and simply use the keyboard shortcuts to navigate. On large projects, this approach is much faster than searching the tree manually for your file.

Docked Mode

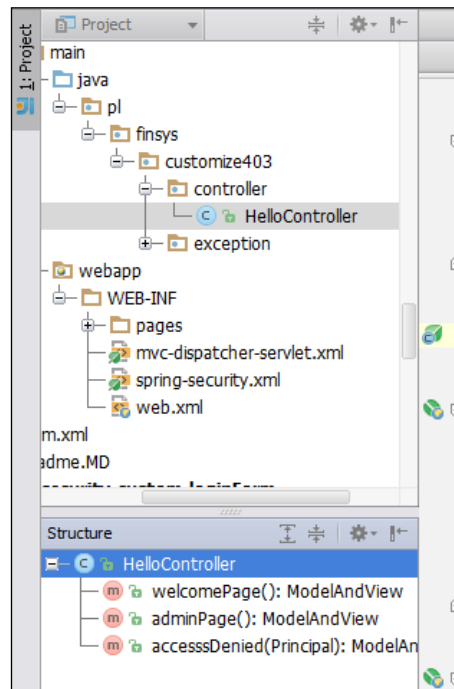
When docked, the tool window will share the total workspace area with other workspace elements such as the editor. On the other hand, when undocked, the tool window will overlap the other workspace elements when resized. An undocked window will go away if inactive. For example, it is especially useful to have the console tool window undocked and resized; reading huge logfiles or console output will be a lot easier.

Floating Mode

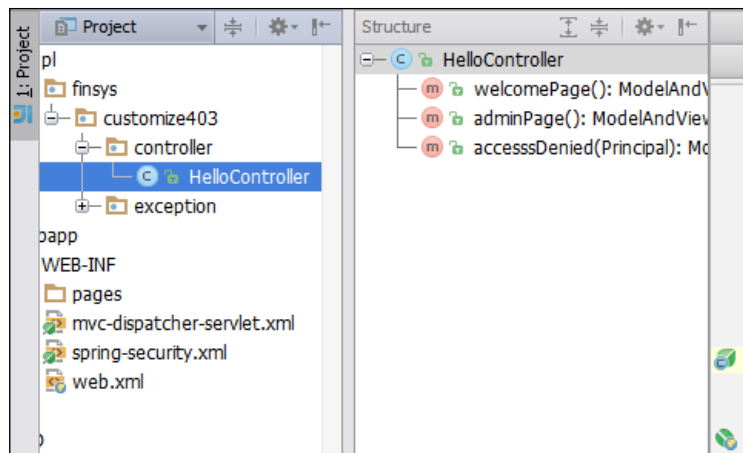
Floating, as the name suggests, allows the tool window to float over the workspace and be detached from the screen edges. It may be useful when you work on multimonitor environments with huge display resolutions set. There are no limiting factors for the number of floating windows shown simultaneously. When floating, tool windows can be easily arranged to suit your needs.

Split Mode

The tool window will share the tool window bar with other tool windows when it has **Split Mode** enabled. This gives you the ability to see two tool windows at once. It's nice to see the project structure and file structure at the same time, as shown in the following screenshot:



When you use the *Ctrl* + left-click (PC) or *cmd* + left-click (Mac) keyboard shortcuts, the splitter between the two tool windows is displayed at once; IntelliJ IDEA will switch them to the wide screen mode and display them in a horizontal layout. It is priceless when you work on a fancy panoramic display and would like to use the screen space effectively, as shown in the following screenshot:



You can quickly go to the specific tool window by using the mnemonic shortcut displayed before its name, for example, *Alt + 1* (PC) or *cmd + 1* (MAC) will take you to the **Project** tool window. It works for hidden tool windows, too.

Multiple views in tool windows

Some tool windows have more than one view available. For example, the **Project** tool window can show **Project**, **Packages**, or **Problems**, as shown in the following screenshot:

