LEARNING
intellij-idea

Free unaffiliated eBook created from Stack Overflow contributors.
# Table of Contents

## About

Chapter 1: Getting started with intellij-idea

Remarks

Examples

Installation or Setup

Windows & Linux

OS X / macOS

Arch Linux

Ubuntu

Other

Hello, World!

Migrating from Eclipse

Chapter 2: Database Tools

Examples

Creating new data source

Chapter 3: Exporting

Examples

Building a .jar

Chapter 4: Hibernate HQL console and inspections

Introduction

Examples

Configuring HQL Inspections

Chapter 5: How to Install Plugins

Introduction

Examples

To download and install a repository plugin

To install a plugin from the disk

Chapter 6: IdeaVim

Remarks
Select configuration and debug ................................................................. 46
  Windows: Alt + Shift + F9 ................................................................. 46
  OS X / macOS: Option + Shift + F9 ..................................................... 46
Run ............................................................................................................. 46
Debug ........................................................................................................ 47
Run context configuration from editor .................................................. 47
  Windows: Ctrl + Shift + F10 ............................................................ 47
  OS X / macOS: Cmd + Shift + F10 .................................................... 47
  Code Completion ............................................................................. 47
Basic code completion (the name of any class, method or variable) ....... 47
  Windows: Ctrl + Space ................................................................. 47
  OS X / macOS: Cmd + Space ........................................................ 47
Smart code completion (filters the list of methods and variables by expected type) ................................. 47
  Windows: Ctrl + Shift + Space ...................................................... 47
  OS X / macOS: Cmd + Shift + Space ............................................ 47
Overwriting code with a suggestion ..................................................... 47
Adding code from a completion suggestion ......................................... 47
  Search/Replace ............................................................................. 48
Search everywhere .................................................................................. 48
Find ....................................................................................................... 48
  Windows / Linux: Ctrl + F ............................................................ 48
  OS X / macOS: Cmd + F .............................................................. 48
Find next ................................................................................................. 48
Find previous ......................................................................................... 48
Replace .................................................................................................. 48
  Windows / Linux: Ctrl + R ............................................................ 48
  OS X / macOS: Cmd + R .............................................................. 48
Find in path ............................................................................................. 48
  Windows / Linux: Ctrl + Shift + F ............................................... 48
  OS X / macOS: Cmd + Shift + F ............................................... 48
Replace in path........................................................................................................48
  Windows / Linux: Ctrl + Shift + R........................................................................49
  OS X / macOS: Cmd + Shift + R...........................................................................49

Refactoring.............................................................................................................49
  Copy.......................................................................................................................49
  Move......................................................................................................................49
  Safe delete............................................................................................................49
    Windows / Linux: Alt + Delete............................................................................49
    OS X / macOS: Cmd + Delete.............................................................................49
  Rename.................................................................................................................49
  Extract Method.....................................................................................................49
    Windows / Linux: Ctrl + Alt + M.......................................................................49
    OS X / macOS: Cmd + Option + M...................................................................49
  Extract Field.........................................................................................................49
    Windows / Linux: Ctrl + Alt + F.......................................................................50
    OS X / macOS: Cmd + Option + F...................................................................50
  Extract Variable....................................................................................................50
    Windows / Linux: Ctrl + Alt + V.......................................................................50
    OS X / macOS: Cmd + Option + V...................................................................50
  Extract Constant..................................................................................................50
    Windows / Linux: Ctrl + Alt + C.......................................................................50
    OS X / macOS: Cmd + Option + C...................................................................50
  Extract Parameter.................................................................................................50
    Windows / Linux: Ctrl + Alt + P.......................................................................50
    OS X / macOS: Cmd + Option + P...................................................................50
  Other....................................................................................................................50

Surround with.........................................................................................................50
  Windows / Linux: Ctrl + Alt + T...........................................................................50
  OS X / macOS: Cmd + Option + T.......................................................................50
  Basic navigation....................................................................................................50
Go to editor (from tool window) .......................................................... 50

Switching focus to corresponding tool window ....................................... 51
  Windows: Alt + <tool window number> ........................................... 51
  OS X / macOS: Cmd + <tool window number> ................................... 51

For example switching focus to the project window .................................. 51
  Windows: Alt + 1 ........................................................................... 51
  OS X / macOS: Cmd + 1. ............................................................... 51

Recent files popup ............................................................................. 51
  Windows: Ctrl + E ........................................................................ 51
  OS X / macOS: Cmd + E ................................................................. 51

Find Action ....................................................................................... 51
  Windows: Ctrl + Shift + A ............................................................... 51
  OS X / macOS: Cmd + Shift + A ..................................................... 51

Navigate to ........................................................................................ 51

File : ............................................................................................. 51
  Windows: Ctrl + Shift + N ............................................................... 51
  OS X / macOS: Cmd + Shift + N ..................................................... 52

Class : ............................................................................................ 52
  Windows: Ctrl + N ........................................................................ 52
  OS X / macOS: Cmd + N ................................................................. 52

Symbol (class/method/variable/constant name) : ...................................... 52
  Windows: Ctrl + Alt + Shift + N ...................................................... 52
  OS X / macOS: Cmd + Option + Shift + N ...................................... 52

Everywhere : .................................................................................. 52
  Windows: Shift + Shift ................................................................... 52
  OS X / macOS: Shift + Shift .......................................................... 52

Go to line number ............................................................................... 52
  Windows: Ctrl + G ......................................................................... 52
  OS X / macOS: Cmd + L ................................................................. 52

Go back to last edit location .................................................................. 52
Windows: Ctrl + Shift + Backspace ................................................................. 53
OS X / macOS: Cmd + Shift + Backspace ....................................................... 53
Usage Search .................................................................................................. 53

Find usages / Find usages in file ................................................................. 53
Windows / Linux: Alt + F7 / Ctrl + F7 .......................................................... 53
OS X / macOS: Option + F7 / Ctrl + F7 ......................................................... 53

Highlight usages in file ............................................................................... 53
Windows / Linux: Ctrl + Shift + F7 .............................................................. 53
OS X / macOS: Cmd + Shift + F7 ................................................................. 53

Show usages ............................................................................................... 53
Windows / Linux: Ctrl + Alt + F7 ................................................................. 53
OS X / macOS: Cmd + Option + F7 .............................................................. 53
Show Method Parameters ........................................................................... 53
Windows / Linux: Ctrl + P .......................................................................... 53
OS X / macOS: Cmd + P ............................................................................. 53
Selection ...................................................................................................... 54
Windows: Ctrl + W ..................................................................................... 54
OS X / macOS: Cmd + W ........................................................................... 54
Windows: Ctrl + Shift + W ......................................................................... 54
OS X / macOS: Cmd + Shift + W ................................................................. 54
Windows: Alt ............................................................................................... 54
OS X / macOS: Opt .................................................................................... 54
Windows: Alt + Shift .................................................................................. 55
OS X / macOS: Opt + Shift ........................................................................ 55
Windows: Alt + J ......................................................................................... 55
OS X / macOS: ctrl + G ............................................................................. 55

Credits ......................................................................................................... 57
About

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: IntelliJ-Idea

It is an unofficial and free IntelliJ-Idea ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official IntelliJ-Idea.

The content is released under Creative Commons BY-SA, and the list of contributors to each chapter are provided in the credits section at the end of this book. Images may be copyright of their respective owners unless otherwise specified. All trademarks and registered trademarks are the property of their respective company owners.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com
Chapter 1: Getting started with intellij-idea

Remarks

intellij-idea is an IDE made as the spiritual successor for the widely-adopted Eclipse IDE used for Java development. Eclipse, although extremely powerful, is often criticized for being extremely clunky and difficult to use.

IntelliJ IDEA attempts to build an IDE with similar power to Eclipse, but with a finishing polish on top. Developers would be at an advantage using IDEA because of the many tools and hooks it has to save time on all projects. Smart code completion, native unit test integration and native Gradle management are just a few of the highlights of Jetbrain's Java IDE

Examples

Installation or Setup

There are two main versions of IntelliJ IDEA: the Community edition and the Ultimate edition. The Community edition is free and is not lacking for features in terms of Java SE development.

Windows & Linux

Download IntelliJ IDEA from the JetBrains website, and follow installation procedures. If the Java Development Kit (JDK) is not installed, download and install the JDK. Note that you need the JDK, only having the Java Runtime Environment (JRE) is not enough.

Once IntelliJ IDEA has been downloaded:

1. Run the installer
2. Press next
3. Choose a folder to install IntelliJ IDEA to (In most cases, leave this as the default)
4. Choose a start menu folder to crete IntelliJ IDEA shortcuts (In most cases, leave this as the default)
5. Choose whether to create a desktop shortcut, and choose whether to associate various Java files with IntelliJ IDEA
6. Press next, and wait for it to install

OS X / macOS

Download IntelliJ IDEA from the JetBrains website, open the disk image (*.dmg) file downloaded, and drag and drop the application to the alias to your /Applications folder.
Arch Linux

IntelliJ IDEA can be installed on Arch Linux using its package manager, `pacman`. Open a terminal and enter the following command.

```
sudo pacman -S intellij-idea-community-edition
```

Using `sudo` is not required if you're running as the root user.

```
@ArchLinux ~ sudo pacman -S intellij-idea-community-edition
[sudo] password for user:
Resolving dependencies...
:: There are 2 providers available for java-environment:
:: Repository extra
  1) jdk7-openjdk  2) jdk8-openjdk
Enter a number (default=1): 2
looking for conflicting packages...

Total Download Size: 251.91 MiB
Total Installed Size: 593.61 MiB
```

Ubuntu

https://riptutorial.com/
(1) Install ubuntu-make package.

For ubuntu 16.04 and later,

```
sudo apt install ubuntu-make
```

For previous versions of ubuntu,

```
sudo add-apt-repository ppa:ubuntu-desktop/ubuntu-make
sudo apt-get update
sudo apt-get install ubuntu-make
```

(2) After installing Ubuntu Make, do a

```
umake ide idea
```

Default installation path: /home/current-user/.local/share/umake/ide/idea

Follow hello_world project listed above.

Follow ubuntu-make page to change default installation and to install other IDEs.

**Other**

Further installation details can be found here:

**Hello, World!**

This will teach you how to make your first project using IDEA.

Launch IDEA, and click Create New Project from the startup screen:
Click Next on the next screen. We're creating a simple Java project, so we don't need any addons or extras to this project.
Use the next screen to create the Java Hello World template project:
Finally, name your project and select a location on disk, and click Finish.
You should end up with a window that looks something like this:
```java
public class Main {
    public static void main(String[] args) {
        System.out.println("Hello World!");
    }
}
```
At this point, the project is all ready to go, simply click the Run button or go to Run -> Run 'Main'.
public class Main {
    public static void main(String[] args) {
        System.out.println("Hello World!");
    }
}
And you're done! The console will automatically pop up, giving it's salutations to the globe!
public class Main {
    public static void main(String[] args) {
        System.out.println("Hello World!");
    }
}

Hello World!

Process finished with exit code 0
Migrating from Eclipse

IntelliJ IDEA attempts to appeal to the wide Java fanbase which uses Eclipse for their development by allowing developers to migrate their Eclipse projects over to an IDEA structure with a few simple clicks!

First, start IDEA and click **Import Project** from the startup window:

![IntelliJ IDEA startup window](https://riptutorial.com/)

Then, select your Eclipse project using the explorer window.
Intellij will prompt you for the model you are importing from, make sure Eclipse is selected before clicking Next.
The next screen will show a confirmation of the path you want to import, simply click Next.
Next, select the modules you want created. In the particular example project, only the Alice and the BuggyRos projects mattered when working in Eclipse.
Finally, make sure the correct version of the JDK is selected before finishing.
And the Eclipse project has been fully migrated to IntelliJ! The project will still open in both IDEs, and will be fully functional in both.

Read Getting started with IntelliJ IDEA online: https://riptutorial.com/intelliJ-idea/topic/1734/getting-started-with-intelliJ-idea
Chapter 2: Database Tools

Examples

Creating new data source

1. Open "Database Tool Window" if you don't have it opened:

   This what it looks like:

   ![Database Tool Window Image]

   ![Database Tool Window Code]

https://riptutorial.com/
2. Create new data source:
   • By clicking on "+" icon:
   • Or by opening "Data Sources and Drivers" dialog
And adding new data source by clicking "+" there

3. If you have not already downloaded JDBC driver, you will be prompted to do it:
4. Enter connection parameters:
5. Configure SSH tunnel if you need it:

6. Configure SSL if you need it:
7. Check that configuration successful, by pressing "Test Connection":

Database: MySQL (ver: 5.7.7-rc)
Identifier case sensitivity: exact (plain), exact (delimited)
Driver (JDBC4.0): MySQL Connector java (ver. mysql-connector-java-5.1.35 (Revision: 5fb9c5849535c13917c2cf9b0aeece6ef9693ef27))

Read Database Tools online: https://riptutorial.com/intellij-idea/topic/6275/database-tools
Chapter 3: Exporting

Examples

Building a .jar

Eventually, when you're ready to release a version of your code to production, you'll need a .jar file to distribute. IntelliJ makes building JARs quite easy.

First, navigate to File -> Project Structure and click on Artifacts.
Click on the + button, and select JAR -> From modules with dependencies:

Select your module from the dropdown list, and the main file (this is the file that contains your...
public static void main() method):

Click **OK**, verify that all the information regarding dependencies is correct, and click **OK** to finish setting up the artifact.
We're not done yet! We've only told Intellij how to build the artifact, now we actually need to make

Simply click **Build -> Build Artifacts**, and click **Build** on the popup menu:

The **jar** will be found in **build -> classes -> artifacts**

Read Exporting online: [https://riptutorial.com/intellij-idea/topic/4807/exporting](https://riptutorial.com/intellij-idea/topic/4807/exporting)
Chapter 4: Hibernate HQL console and inspections

Introduction

IntelliJ IDEA supports HQL auto completion and running HQL queries on console. This is how you enable that support.

Examples

Configuring HQL Inspections

1. Go to File -> Project Structure -> Modules.
2. Add new Hibernate module.
3. Right click on the desired module -> Add -> Hibernate.
4. Select the newly created Hibernate configuration option, and click the (+) sign in the right pane to create hibernate.cfg.xml file.
5. Go to File -> Project Structure -> Facets, and add new JPA.
6. Select the newly created JPA configuration option, and click the (+) sign in the right pane to assign it your Hibernate configuration file.
7. Open Persistence window, there you should see the list of your project modules.
8. Expand the module name, and assign your data source to the hibernate.cfg.xml file.

Now you can write queries on hibernate console and get HQL auto completion.

Read Hibernate HQL console and inspections online: https://riptutorial.com/intellij-idea/topic/9617/hibernate-hql-console-and-inspections
Chapter 5: How to Install Plugins

Introduction

Plugins help us to do things easier. IntelliJ provides a vast range of plugins for various technologies/languages. There are three ways to install plugin in IntelliJ.

Examples

To download and install a repository plugin

1. Go to File --> Settings (e.g. Ctrl+Alt+S).
2. In the left-hand pane, select Plugins.
3. On the Plugins window, click "Install JetBrains plugin" or the "Browse repositories button".

To install a plugin from the disk

1. Go to File --> Settings (e.g. Ctrl+Alt+S).
2. In the left-hand pane, select Plugins.
3. On the Plugins window, click "Install plugin from disk button".
4. Select the desired plugin from your local machine.
   Click Apply button of the Settings/Preferences dialog.
Or you can directly add plugins from the other vendors by directly searching the repositories.

1. Go to "Browse Repositories"

2. Select the category (on the top of the window) that you need to search (or just search by the name if you know it).

3. Install it.
Read How to Install Plugins online: https://riptutorial.com/intellij-idea/topic/8069/how-to-install-plugins
Chapter 6: IdeaVim

Remarks

IdeaVim is a plugin for IDEA products that aims in providing Vim functionality in editor views

Examples

Showing line numbers

As of IntelliJ IDEA version 2016.2, and IdeaVim version 0.46, IntelliJ's native option for showing line numbers is ineffective. When clicking Show line numbers, the line numbers immediately show and disappear.

This problem is caused by a bug in the IdeaVim plugin, which can be resolved by using the Vim command for showing line numbers:

```vim
:set number
```

and

```vim
:set nonumber
```

to hide.

These commands can also be used as the shorthand :set nu and :set nonu.

If you wish to activate the feature which shows relative line numbers instead you can use

```vim
:set relativenumber
```

or a shorthand :set rnu. Remember that you can mix set relativenumber with set number.

Allocating conflicting keystrokes to IdeaVim

By default, some keystrokes that are useful in Vim contradict with the keystrokes of IntelliJ.

For example, ^R in Vim is ‘redo’, but in IntelliJ it's the shortcut for Run

To decide which program interprets the keystroke, go to Preferences -> Other Settings -> Vim Emulation and choose which keystrokes to use with IdeaVim and which to use with IntelliJ:
Read IdeaVim online:
https://riptutorial.com/intellij-idea/topic/4210/ideavim

https://riptutorial.com/
Chapter 7: Inspections

Introduction

Intellij IDEA provides lots of code inspections, that can significantly simplify writing code.

Inspections parameters can be found in Preferences | Editor | Inspections section. By default, IDEA has lots of them enabled. And lots of inspections support auto-fixing options, that can be seen on pressing Alt + Enter.

To run inspections for your whole project (or some custom scope), you need to select Analyze | Inspect code.

Examples

@NotNull / @Nullable inspections

These inspections are extremely useful for preventing NullPointerExceptions. By default they are disabled. You can find these inspections in Inspections preferences: Java | Probable bugs | Constant conditions & exceptions and @NotNull/@Nullable problems. There you can also configure your annotations. You can use this manual to add JetBrains annotations into your project.

For example, consider this methods:

```java
public int getLength() {
    return getString().length();
}

public String getString() {
    return "not null string";
}
```

If `getString` can't possibly return null, everything is fine. But if we enable our inspections and in some cases it can return null, we will immediately will see inspection triggered:

```java
public int getLength() {
    return getString().length();
}

public String getString() {
    if (condition()) {
        return null;
    }
    return "not null string";
}
```

which says 'null' is returned by the method which is not declared as @Nullable. And if we hit Alt + Enter, there will be an option Annotate method as '@Nullable'. If we hit Enter again, our code will look like that:
with inspection triggered on `length()` method saying Method invocation 'length' may produce 'java.lang.NullPointerException'. And if we go further and introduce the result of `getString()` method as a variable, after hitting Alt + Enter IDEA will suggest a few ways to fix this inspection:

This way you can inspect your code on-the-fly, and fix all potential `NullPointerException`s. If you want to check your whole project (or some random scope), you can use Analyze | Inspect code. Just make sure that your selected inspections profile has all necessary inspections enabled.

Read Inspections online: https://riptutorial.com/intellij-idea/topic/9361/inspections
Chapter 8: Little-known features

Examples

Language injection

If you want to write strings containing other languages (JSON, regexes), it's hard to keep up with escaping symbols, and it would be nice to get some code assist.

1. Put your cursor inside an empty string
2. ALT + ENTER
3. Pick "Inect language or reference"

4. Pick the desirable language (RegExp in my case) from the pop-up

5. Again use ALT + ENTER and pick Edit regex fragment

6. In the new tool window enter the regex - note how it's automatically mapped to a properly escaped Java string. Similarly for JSON the indents will be placed properly.

Quick Preview

Check screencast video on YouTube about this feature

https://riptutorial.com/
IntelliJ provides a quick-preview feature called **Viewing Definition**. Using this feature allows a user to quickly see the contents of a method/class without navigating into the class itself:

- OS X: `(1+y)` or `(1+Space)`
- Unix / Windows: `Ctrl+Shift+I`

Example: Looking into `Arrays.copyOfRange()`:

```java
public static char[] copyOfRange(char[] original, int newLength = to - from;
    if (newLength < 0)
        throw new IllegalArgumentException(from
    char[] copy = new char[newLength];
    System.arraycopy(original, from, copy, 0,
        Math.min(original.length -
    return copy;
```

Verifying if you've selected the right class in **Search**.

https://riptutorial.com/
public final class String

    implements java.io.Serializable, Comparable<String>

    /** The value is used for character storage. */
    private final char value[];

    /** Cache the hash code for the string. */
    private int hash; // Default to 0

    /** use serialVersionUID from JDK 1.0.2 for interoperability */
    private static final long serialVersionUID =

    /****
    * Class String is special cased within the String
    // Note: offset must be >= 0
    if (offset >= length) throw new StringIndexOutOfBoundsException()
    }
Chapter 9: Live Templates

Examples

Add a test method easily

```java
@org.junit.Test
public void should_$name$() {
    $END$
}
```

Make sure to check the **Shorted FQ names** box when creating this template.

<table>
<thead>
<tr>
<th>Abbreviation:</th>
<th>should</th>
</tr>
</thead>
<tbody>
<tr>
<td>Template text:</td>
<td></td>
</tr>
</tbody>
</table>
| ```java
@org.junit.Test
public void should_$name$() {
    $END$
}
``` |

Applicable in Java: declaration.  Change

When you type "should" (the abbreviation), this will add the necessary `import org.junit.Test;` statement at the top of the file, and this code:

```java
@Test
public void should_() {
    
}
```

It is thanks to the **Shorten FQ names** option that `@org.junit.Test` is reduced to simply `@Test`.

The `$name$` variable is irrelevant, it could be named something else. The purpose of that variable is that when the template is inserted in the class, the cursor will be placed in the position of `$name$`, asking you to enter something.

https://riptutorial.com/
After you entered a value for $name$ (effectively the name of the test method), the cursor will finally jump to $END$, a built-in variable, so that you can carry on and implement the test case.

**Insert the name of the current class**

Consider the *utility class pattern*: a class with only static methods and no fields. It's recommended to prevent instantiation of such classes by adding a private constructor.

This live template example makes it easy to add a private constructor to an existing class, using the name of the enclosing class.

```java
private $className$() {
    throw new AssertionError("utility class, forbidden constructor");
}
```

Applicable in Java: declaration scope.

**Abbreviation:** utility_class  
**Template text:**

```java
private $className$() {
    throw new AssertionError("utility class, forbidden constructor");
}
```

Click **Edit variables** to define the className variable as the built-in className() expression, and check the **Skip if defined** box to avoid prompting for a custom name, which is unnecessary in this example.
For example, inside a class like this:

```java
class ListUtils {
    // ...
}
```

When you type "utility_class" (the abbreviation), this will insert a constructor like this:

```java
class ListUtils {
    private ListUtils() {
        throw new AssertionError("utility class, forbidden constructor");
    }
    // ...
}
```

Read Live Templates online: https://riptutorial.com/intellij-idea/topic/2703/live-templates
Chapter 10: Optimization

Examples

Customizing the VM Options

You can override the default `vmoptions` with your own personal settings by choosing Help > Edit Custom VM Options from the toolbar. This will create a local copy of the file which you are free to edit.

For example, if you double the value set for `Xmx`, the maximum size of the memory allocation pool will be doubled after you restart the IDE. On many machines this will lead to faster performance.

See the options here for a description of each parameter.

Read Optimization online: https://riptutorial.com/intellij-idea/topic/5305/optimization
Chapter 11: Useful Shortcuts

Examples

Compile and Run

Make project (compile modified and dependent)

Windows: Ctrl + F9

OS X / macOS: Cmd + F9

Compile selected file, package or module

This is useful to know, as when debugging this shortcut can be used to quickly reload / hotswap classes.

Windows: Ctrl + Shift + F9

OS X / macOS: Cmd + Shift + F9

Select configuration and run

Windows: Alt + Shift + F10

OS X / macOS: Option + Shift + F10

Select configuration and debug

Windows: Alt + Shift + F9

OS X / macOS: Option + Shift + F9
Run

Shift + F10

Debug

Shift + F9

Run context configuration from editor

Windows: Ctrl + Shift + F10

OS X / macOS: Cmd + Shift + F10

Code Completion

Basic code completion (the name of any class, method or variable)

Windows: Ctrl + Space

OS X / macOS: Cmd + Space

Smart code completion (filters the list of methods and variables by expected type)

Windows: Ctrl + Shift + Space

OS X / macOS: Cmd + Shift + Space

Overwriting code with a suggestion

Tab
Adding code from a completion suggestion

Enter

Search/Replace

Search everywhere

Double Shift

Find

Windows / Linux: Ctrl + F
OS X / macOS: Cmd + F

Find next
F3

Find previous
Shift + F3

Replace

Windows / Linux: Ctrl + R
OS X / macOS: Cmd + R

Find in path

Windows / Linux: Ctrl + Shift + F
OS X / macOS: Cmd + Shift + F
Replace in path

Windows / Linux: Ctrl + Shift + R

OS X / macOS: Cmd + Shift + R

Refactoring

Copy
F5

Move
F6

Safe delete

Windows / Linux: Alt + Delete

OS X / macOS: Cmd + Delete

Note that the Delete key on OS X / macOS is the equivalent of the Backspace key on other operating systems.

Rename

Shift + F6

Extract Method

Windows / Linux: Ctrl + Alt + M

OS X / macOS: Cmd + Option + M

Extract Field
Windows / Linux: Ctrl + Alt + F

OS X / macOS: Cmd + Option + F

Extract Variable

Windows / Linux: Ctrl + Alt + V

OS X / macOS: Cmd + Option + V

Extract Constant

Windows / Linux: Ctrl + Alt + C

OS X / macOS: Cmd + Option + C

Extract Parameter

Windows / Linux: Ctrl + Alt + P

OS X / macOS: Cmd + Option + P

Other

Surround with

Surrounds a code block with an if, for, <editor-fold ...> and more.

Windows / Linux: Ctrl + Alt + T

OS X / macOS: Cmd + Option + T

Basic navigation

Go to editor (from tool window)
Switching focus to corresponding tool window

Windows: Alt + <tool window number>

OS X / macOS: Cmd + <tool window number>

For example switching focus to the project window

Windows: Alt + 1

OS X / macOS: Cmd + 1

Recent files popup

Windows: Ctrl + E

OS X / macOS: Cmd + E

Find Action

Windows: Ctrl + Shift + A

OS X / macOS: Cmd + Shift + A

Navigate to

File:

Windows: Ctrl + Shift + N
OS X / macOS: Cmd + Shift + N

Class:

- Windows: Ctrl + N
- OS X / macOS: Cmd + N

Symbol (class/method/variable/constant name):

- Windows: Ctrl + Alt + Shift + N
- OS X / macOS: Cmd + Option + Shift + N

Note that you can use class name to narrow down the method/variable/constant search, for example to find symbol `usersCollection` in class `UserDAO` type:

```
UserDAO.usersCollection
```

Everywhere:

- Windows: Shift + Shift
- OS X / macOS: Shift + Shift

To search for something that has multiple words, e.g., `InetAddressCachePolicy` you can just type `InAddCacPo` or something similar that contains parts of words in the whole name.

Go to line number

Windows: Ctrl + G
OS X / macOS: Cmd + L
Go back to last edit location

**Windows:** Ctrl + Shift + Backspace

**OS X / macOS:** Cmd + Shift + Backspace

Usage Search

Find usages / Find usages in file

**Windows / Linux:** Alt + F7 / Ctrl + F7

**OS X / macOS:** Option + F7 / Ctrl + F7

Highlight usages in file

**Windows / Linux:** Ctrl + Shift + F7

**OS X / macOS:** Cmd + Shift + F7

Show usages

**Windows / Linux:** Ctrl + Alt + F7

**OS X / macOS:** Cmd + Option + F7

Show Method Parameters

**Windows / Linux:** Ctrl + P

**OS X / macOS:** Cmd + P

Shows what parameters a method and all of its overloads accepts.
Selection

Selection with increasing scope

This comes handy when you want to select a block to extract a variable / method etc, no need to do a precise bracket matching, just put the caret somewhere in the statement and keep doing this

Windows: Ctrl + W

OS X / macOS: Cmd + W

Selection with decreasing scope

Windows: Ctrl + Shift + W

OS X / macOS: Cmd + Shift + W

This feature is also very useful when editing / playing with json documents in your IDE.

Vertical selection

Press and hold

Windows: Alt

OS X / macOS: Opt

and select normally using mouse / trackpad (the way you select a word in a row etc)

This is how it should look like

https://riptutorial.com/
Multiple carets

Press and hold

**Windows:** \texttt{Alt + Shift}

**OS X / macOS:** \texttt{Opt + Shift}

and click where all you want to put a caret. You can choose to put multiple carets in a single line or across lines at different positions.

Now you can perform all operations that you would have been able to perform on a single selected word (hold \texttt{Ctrl} (windows) or \texttt{option} (mac OS) and use \texttt{Left} or \texttt{Right} keys to jump across words) and all those will affect all caret positions.

You can even cut / paste multiple selections from one place to another.

Having multiple carets is very useful when you want to change the structure of text across many lines / many positions in same line.

**Selecting duplicate occurrences**

Select some text and press

**Windows:** \texttt{Alt + J}

**OS X / macOS:** \texttt{ctrl + G}

to select the next occurrence of the same text.

You get one caret at each of the selected occurrence that could be used to change each occurrence simultaneously.

E.g., I've tried to put an example in this gif, hope it helps

https://riptutorial.com/
Read Useful Shortcuts online: https://riptutorial.com/intellij-idea/topic/3085/useful-shortcuts
## Credits

<table>
<thead>
<tr>
<th>S. No</th>
<th>Chapters</th>
<th>Contributors</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Getting started with IntelliJ-idea</td>
<td>Community, CraftedCart, foxtrot9, Harmelodic, Jeeter</td>
</tr>
<tr>
<td>2</td>
<td>Database Tools</td>
<td>kassak</td>
</tr>
<tr>
<td>3</td>
<td>Exporting</td>
<td>Jeeter</td>
</tr>
<tr>
<td>4</td>
<td>Hibernate HQL console and inspections</td>
<td>Shai Givati</td>
</tr>
<tr>
<td>5</td>
<td>How to Install Plugins</td>
<td>Priya, RamenChef, Sudeepa Nadeeshan</td>
</tr>
<tr>
<td>6</td>
<td>IdeaVim</td>
<td>Jeeter, Mateusz Piotrowski, mnoronha, sevenforce</td>
</tr>
<tr>
<td>7</td>
<td>Inspections</td>
<td>esin88</td>
</tr>
<tr>
<td>8</td>
<td>Little-known features</td>
<td>Anton Dozortsev, Jeeter, mszymborski, Prateek</td>
</tr>
<tr>
<td>9</td>
<td>Live Templates</td>
<td>janos</td>
</tr>
<tr>
<td>10</td>
<td>Optimization</td>
<td>Kevin Cooper</td>
</tr>
<tr>
<td>11</td>
<td>Useful Shortcuts</td>
<td>CraftedCart, karel, mszymborski, Nadim Bahadoor, Prateek</td>
</tr>
</tbody>
</table>