



# IntelliJ IDEA Workshop

Welcome to a hands-on IntelliJ IDEA workshop. This session will help you go from just knowing the basics to becoming truly productive with IntelliJ. We'll cover the most important shortcuts, workflows, and hidden features that can seriously improve your speed and code quality.

---



## Initial Setup














To ensure a good learning experience:

- Set your preferred theme (light/dark), find **Theme**
  - Font: **15** (**Font**)
  - Turn on font ligatures (**Font** → **Enable font ligatures**)
  - Zoom in/out using **^** **⌘** **-/=** (**Ctrl+Alt** + **-/=**)
  - Turn on **Presentation Assistant** (**Presentation assistant**)
- 



## 1. Search & Navigation

Action	macOS Shortcut	(Windows/Linux)
Search Everywhere	<b>⇧ ⇧</b>	(Shift Shift)
Go to Class	<b>⌘ O</b>	(Ctrl+N)
Go to File	<b>⌘ ⇧ O</b>	(Ctrl+Shift+N)
Go to Symbol	<b>⌘ ⌘ O</b>	(Ctrl+Alt+Shift+N)
Find Action	<b>⌘ ⇧ A</b>	(Ctrl+Shift+A)
Go to Endpoint (Spring)	<b>⌘ ⇧ \</b>	(Ctrl+Shift+\)

Navigate Between Tabs	  [ /   ]	(Ctrl+Shift+[ / ])
Navigate to Test	  T	(Ctrl+Shift+T)
Jump to Line	 L	(Ctrl+G)
Recent Files	 E	(Ctrl+E)
Recent Locations	  E	(Ctrl+Shift+E)
Back/Forward	 [ /   ]	(Ctrl+Alt+Left/Right)

## Exercises:

- Navigate to class *StringUtils* with **Go to class**
- Navigate to line 1500 in a class *StringUtils*
  - a. Use **Go to class** & **Jump to line**
  - b. Use only with **Go to class**
- Open a file with *Readme.MD* with **Go to file**
- Use **Find action** to search for **Set Background Image** and choose a file from the *wallpapers* folder
- Use **Go to symbol** to find symbol *SPACE*
- Use **Go to endpoint** to find endpoints with *offers* in path
- Find methods *isEmpty(...)* inside *StringUtils* class by using:
  - a. **Go to symbol**
  - b. **Go to class** & **File Structure**
- Find methods *isEmpty(...)* with **Go to symbol**
- Use **Recent files** to open:
  - a. *Terminal*
  - b. *Gradle* tab
  - c. any of recent files
- Go from/to *OffersService* and its test class with **Go to test (subject)**

- Use **Search everywhere** to calculate  $2+2*2$



## 2. Tool Windows

Action	macOS Shortcut	(Windows/Linux)
Show Tool Window	<b>⌘ 1, ⌘ 4, ⌘ 5</b>	(Alt+1, Alt+4...)
Hide/Restore Tool Windows	<b>⌘ ⇧ F12</b>	(Ctrl+Shift+F12)
Maximize/Unmaximize Tool Window	<b>⌘ ⇧ ' (apostrophe)</b>	(Ctrl+Shift+F12)
Enter Distraction Free Mode		
Select File in Project View		
press ESCAPE		
Stretch (tool window) to Left / Right	<b>⌘ ⇧ ⌵ / ⌶</b>	(Ctrl+Alt+Shift + < / >)



### Exercises:

- Assigning a new shortcut
  - a. Go to class **OffersService**
  - b. Find this file in a **Project** view
  - c. Find action **Select File in Project View**
  - d. Check the shortcut near the action name and use it
  - e. Find again action **Select File in Project View**
  - f. Press **⇧ ENTER** and assign new shortcut, for example **⌘ ⇧ 1**
- Check the shortcut for **Gradle** tab in **Recent files**
  - a. Find action **Gradle**

- b. Check if it opens and closes the *Gradle* tab
    - c. Assign again new shortcut
    - d. Check if it works correctly
    - e. Check again **Recent files** and assigned shortcut
    - f. Run tests from the *Gradle* tab
  - Find **Editor tabs** in the settings
    - a. Set to *None*
    - b. Enjoy!
  - Find **Keymap**
- Find action name searching by shortcut
- Open **Project view** with **⌘ 1** (ALT 1)
    - a. Use **Stretch (tool window) to Left / Right**
    - b. Do the same for any other tool window (**Recent Files**, **Find Action**, **Run** etc)

If stretching doesn't work, go to *Keymap* and check:

- **Stretch to Left/Right**
- **Increment/decrement width/height**
  - c. use *ESCAPE* to jump back to editor from the tool window
- Search for **Run anything**
  - a. Use shortcut for **Run anything**
  - b. Run *AppRunner*
  - c. Stop application
  - d. Run again with **Run**



### 3. Bookmarks



Action	macOS Shortcut	(Windows/Linux)
Toggle Bookmark	<b>F3</b>	(F11)
Toggle Bookmark	<b>^ ⌘ 1</b> <b>(1–9)</b>	(Ctrl+Shift+1)
Show Bookmarks	<b>⌘ F3</b>	(Shift+F11)

## Exercise:

- Exploring bookmarks
    - a. Find the constant `MAX` with the Go to symbol
    - b. Add a bookmark with the `Toggle bookmark`
    - c. `Go to class` `StringU:146`
    - d. Toggle Bookmark with `Toogle bookmark 1`
    - e. Open bookmarks
      - from `Recent Files`
      - by using a shortcut
    - f. Go to bookmark 1 with a shortcut
- 



## 4. Scratch Files

Action	macOS Shortcut	(Windows/Linux)
Create Scratch File	  <code>N</code>	(Ctrl+Shift+Alt+Insert)



## Exercises:

- Write a quick main method in a Java file
  - a. `Create Scratch File` in `Java`
  - b. add new `System.out.println()` with by typing `sout`
  - c. paste inside `new Date().getTime()`
  - d. `Run` the scratch file
  - e. Check where the file is located by using `Select file in a project view`
  - f. Open `Terminal`:
    - `Find action`
    - from `Recent Files`
  - g. Check if `GIT` tracks the new file




- Curl and *.http* scratch
    - Go to the *curl.txt* file
    - Copy content
    - Create Scratch file with *.http* extension
    - Paste curl
    - Make some modifications
    - Export to curl and paste in curl.txt
    - Check if your modifications are present
- 

## 5. Editing

Action	macOS Shortcut	(Windows/Linux)
Extend Selection	 ↑	(Ctrl+W)
Shrink Selection	 ↓	(Ctrl+Shift+W)
Paste from History	 ⇧ V	(Ctrl+Shift+V)
Delete Line	 ⌫	(Ctrl+Y)
Duplicate Line	 D	(Ctrl+D)
Move Line Up/Down	⌘ ⇧ ↑ / ↓	(Shift+Alt+↑ / ↓)
Move Statement Up/Down	 ⇧ ↑ / ↓	(Ctrl+Shift+↑ / ↓)
Complete Current Statement	 ⇧ ↵	(Ctrl+Shift+Enter)
Join Lines	^ ⇧ J	(Ctrl+Shift+J)
Select All Occurrences	^  G	(Ctrl+Alt+Shift+J)
Next/Previous Occurrence	^ G / ⇧ ^ G	(Ctrl+Alt+Right/Left)
Comment Line	 /	(Ctrl+ <i>/</i> )

Comment Block	  /	(Ctrl+Shift+/,)
---------------	---	-----------------

## Exercises:

- Use   V (Ctrl+Shift+V) to paste older values
- Select all variable uses with ^  G (Ctrl+Alt+Shift+J)
- Duplicate, delete lines
- Move lines down and up
- Move whole statements up and down
- Comment lines, blocks
- **Next Highlighted Error**
  - a. Go to class *StringUtils*
  - b. Go to import (first line of the file) with Go to line
  - c. Remove one of the imports. Code won't compile now

### What is the fastest way to find where is the error?

- Scrolling
- Red line in the scrolling area
- Problems tab
- **Next Highlighted Error**

- Searching for actions

Go to file *StringUtils*:

- a. Render doc comments
  - b. Fold/expand and expand all comments
  - c. Fold/expand any if statement
  - d. Fold/expand any method
  - e. Fold/expand everything
-



## 6. Testing

Action	macOS Shortcut	(Windows/Linux)
Run Test	⌘ R /  ⌥ R	(Ctrl+Shift+F10)
Debug Test	⌘ D /  ⌥ D	(Ctrl+Shift+F9)
Create/Navigate to Test	⌘ ⌥ T	(Ctrl+Shift+T)



### Exercises:

- Basic test navigation and run
  - Go to *OffersService*
  - Navigate to Test
  - Run a whole test suite
  - Jump to Run tab and go to failed test
  - Run a single test that failed
- Multi-line cursor
  - Go to class *OffersServiceTest*
  - In the section where, add additional pipe (|) between input parameters (offset and limit) and expected parameters *expectedOfferIds* with multi-line cursor:

Press ⌘ twice, and then without releasing it, press up or down arrow keys.

- Run this single test
- Duplicate last line in the *where* section
- Set *offset*, edit *expectedOfferIds*
- Run the test again
- Reformat code

<https://www.jetbrains.com/help/idea/multicursor.html#select-multiple-words-or-text-ranges>

- Remove all annotations @Unroll

### Hints

- Use Replace in files - ⌘ ⌥ R (Ctrl + Shift + R)
- Enable regex
- Enable/disable regex and matching case in Replace in files with shortcuts



The code may be formatted improperly after replacing.

- **Reformat code** in the whole directory
- a. **Select file in a project view**
  - b. **Reformat code** on a whole directory, for example test
  - c. Use mnemonics by holding **⌘** (Alt) to pressing characters select **Optimize imports**

<https://plugins.jetbrains.com/docs/intellij/mnemonics.html>

---

## 7. Debugging

Action	macOS Shortcut	(Windows/Linux)
Toggle Breakpoint	⌘ F8	(Ctrl+F8)
View Breakpoints	⌘ ⇧ F8	(Ctrl+Shift+F8)
Step Over	F8	(F8)
Step Into	F7	(F7)
Smart Step Into	⇧ F7	(Shift+F7)
Step Out	⇧ F8	(Shift+F8)
Run to Cursor	⌘ F9	(Alt+F9)
Evaluate Expression	⌘ F8	(Alt+F8)
Resume Program	⌘ ⌘ R	(F9)
Show Execution Point	⌘ ⇧ F10	(Alt+F10)

## Exercise:

- Go to *OffersServiceTest*
  - a. Run all tests
  - b. Run a single failing test
  - c. Jump to the tested method
  - d. Add breakpoint
  - e. Go further and add more breakpoint
  - f. Run in debug mode
  - g. When you stop near this expression:















*if (limit != null && limit <= -2 ...)*

Evaluate expression: `limit != null && limit <= -2`




- h. Fix error
- i. Run tests again
- j. Disable all breakpoint (View breakpoints) with mnemonics
- k. Run test in Debug

---

## 8. Refactoring





Action	macOS Shortcut	(Windows/Linux)
Refactor This	 T	(Ctrl+Alt+Shift+T)
Rename	 F6	(Shift+F6)
Change Signature	 F6	(Ctrl+F6)
Extract Method	  M	(Ctrl+Alt+M)
Extract Constant	  C	(Ctrl+Alt+C)
Extract Variable	  V	(Ctrl+Alt+V)
Inline	  N	(Ctrl+Alt+N)
Convert Java to Kotlin	   K	(Ctrl+Alt+Shift+K)

## Exercise:

- Rename a method using  F6 (Shift+F6)
  - Check Refactor this
  - Refactoring OffersService
    - a. Extract constant in method `add(Offer offer, String accountId)`  
`List.of(AccountStatus.TO_ACTIVATE, AccountStatus.BLOCKED)`
    - b. Extract method with if condition  
`if (EXTRACT_METHOD)`
    - c. Extract variable with `accountStatus`
    - d. Inline all changes back
  - Move exception to separate file
  - Check options for refactoring for Java file (*OffersService*) in the top menu
  - Convert Java class to Kotlin with   K (Ctrl+Alt+Shift+K)
  - Check options for refactoring for Kotlin file (*OffersService*)
- 



## 9. Show usages



Action	macOS Shortcut	(Windows/Linux)
Show Implementations	  B	(Ctrl+Alt+B)
Show Usages	  F7	(Ctrl+Alt+F7)

## 10. Intent Actions, Live Templates, Postfix Completion, Surround With

## a. Intent Actions (**Alt + Enter**)

IntelliJ often suggests actions, like fixing imports, simplifying code, or generating code.

### Shortcut:

-   (Mac)
- **Alt + Enter** (Windows)


### Examples:

- Add missing import
  - Implement interface
  - Surround with try/catch
  - Simplify expression
  - Convert to **stream()** / **forEach**
- 

## b. Live Templates

Live templates are code snippets triggered by short keywords.

### Shortcut to open live templates settings:

-  , → **Live Templates** (Mac)
- **Ctrl + Alt + S** → **Editor > Live Templates** (Windows)

### Examples:

- **psvm** → **public static void main(...)**
- **sout** → **System.out.println(...)**
- **ifn** → **if (x != null) {...}**
- **spgwt** → Spock test structure (**given-when-then**)

Use `^ Space` (Mac) / `Ctrl + Space` (Windows) to trigger basic code completion, which includes templates.

---

### c. Postfix Completion

Transform an existing expression with postfix templates.

#### Examples:

- `offerId.nn` → `if (offerId != null) {}`
- `list.for` → `for (item : list) {...}`
- `str.sout` → `System.out.println(str);`
- `bool.not` → `!bool`

To view or customize:

- Preferences → `Editor > General > Postfix Completion`

---

### d. Surround With

Surround code block with common structures like `if`, `try/catch`.

#### Shortcut:

- `⌘ ⇧ T` (Mac)
- `Ctrl + Alt + T` (Windows)

#### Examples:

- Surround with `if`
  - Surround with `try/catch`
-

## Exercises

- Add [Live Template](#) called “lognn” for

```
if (limit != null) {  
    System.out.println(limit);  
}
```

- Add [Postfix Completion](#) template called “log” like

*variable.log*

which results in

*System.out.println(variable)*

- Write simple [Live Template](#) for *Spock* tests “spgwt” (Spock-Given-When-Then)

```
def "()" {  
  
    given:  
  
    when:  
  
    then:  
  
}
```

## Live Templates examples:

- [sout](#), [soutv](#), [soutm](#)
- [ifn-if \(object == null\)](#)
- Create your own templates in: [Settings](#) → [Editor](#) → [Live Templates](#)

## 10. Additionally

- Search Structurally / Replace structurally
  - Project View Options
  - Help > Search
  - Help > Keymap Shortcuts PDF
  - Help > IntelliJ IDEA on YouTube
  - Help > My Productivity
  - Help > Change memory settings
- 






## 10. Plugins Worth Trying

- **Presentation Assistant** – show shortcuts during actions
  - **Rainbow Brackets** – matching brackets
  - **Key Promoter X** – teaches shortcuts while you click
  - **String Manipulation** – conversions, encode/decode
  - **HTTP Client CLI** – better than Postman
  - **.env support**
- 



## Wrap-up

### Suggested Next Steps:

- Practice   
- Customize your keymap for even faster access

## Additional resources

<https://blog.jetbrains.com/idea/category/tips-tricks/>

<https://www.youtube.com/user/intellijideavideo>

<https://blog.jetbrains.com/kotlin/2021/06/simple-steps-for-improving-your-ide-performance/>

<https://blog.jetbrains.com/idea/2021/08/10-places-you-don-t-need-to-use-the-mouse-in-intellij-idea>

<https://blog.jetbrains.com/idea/2022/10/top-underrated-shortcuts/>

<http://courses.ics.hawaii.edu/ics314s17/morea/development-environments/reading-intellij-macos-configuration.html>

<https://itectec.com/askdifferent/mac-os-how-to-disable-the-minimize-command-m-shortcut-in-mavericks>

[https://www.youtube.com/watch?v=NUndgK7f1\\_Q](https://www.youtube.com/watch?v=NUndgK7f1_Q)