

# IntelliJ IDEA Dojo

Become a more effective JVM developer  
Daniel Hinojosa





# About this course

- This course is a thorough introduction to IntelliJ IDEA
- We will cover items like
  - Navigation and Tool Windows
  - Effective Test Driven Development
  - Maneuvering Efficiently
  - Multicursors
  - Templates
  - Version Control Management
  - Artificial Intelligence in IntelliJ and other JetBrains products
- We will have some labs to get those fingers working

O'REILLY®

# IntelliJ Purpose, Tool Windows, and Files





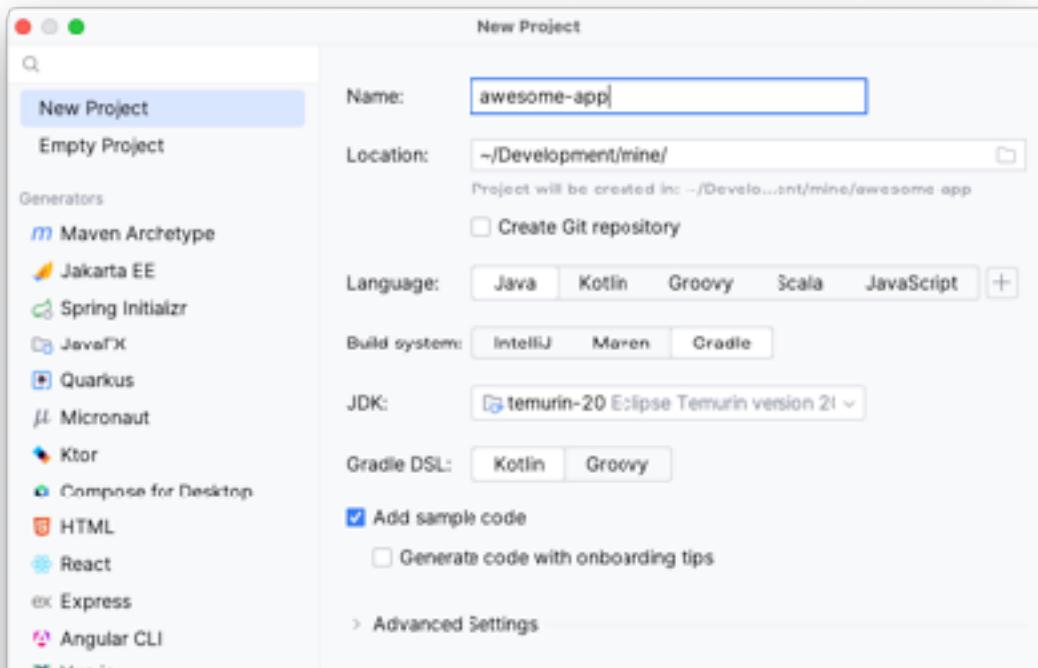


# New Projects



# New Projects

- Typically, projects are created outside the IDE.
- Created and maintained by a build tool, i.e., Maven, Gradle, etc.
- But, IntelliJ offers some good choices without resorting to “IntelliJ-only” type projects





# Opening Projects



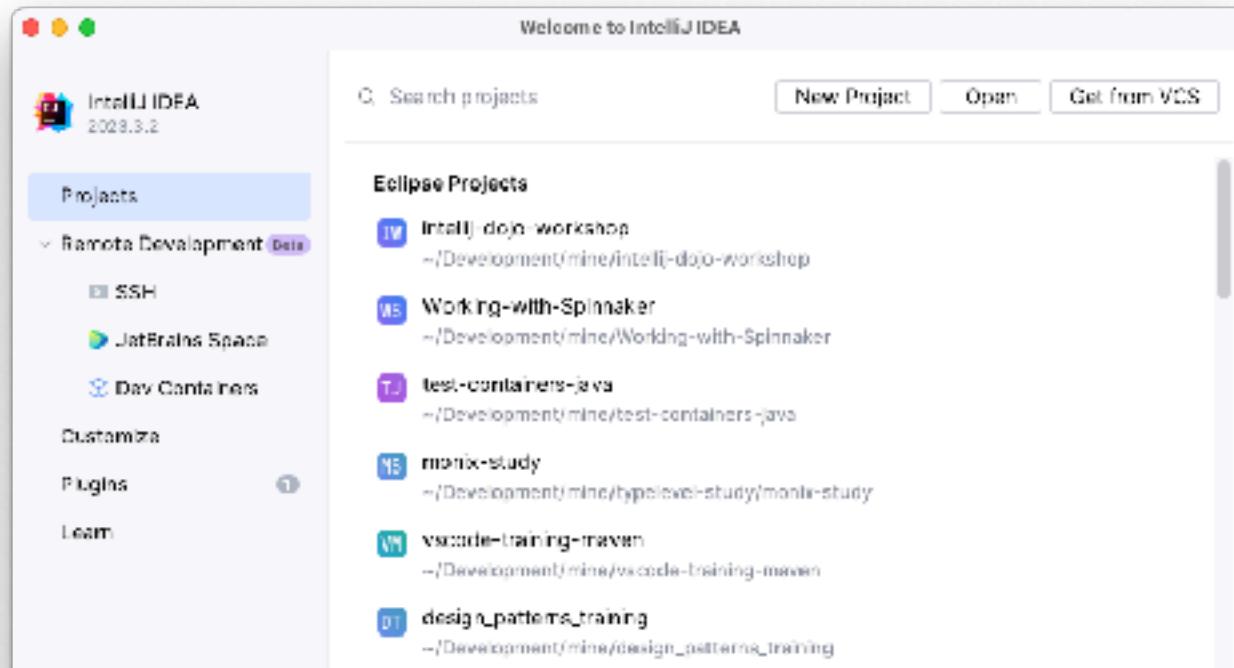
# Opening Projects

- Since projects are created outside the IDE with a build tool. i.e., Maven, Gradle, etc.
- All that needs to be done is open the project.
- This can be done with **[File > Open]** or **[File > Recent Projects]**



# Opening Projects

- When you start IntelliJ for the first time, or when you close out of currently open projects, you can open your projects in the Welcome to IntelliJ window.





# Command Line



# Opening Projects using the Command Line

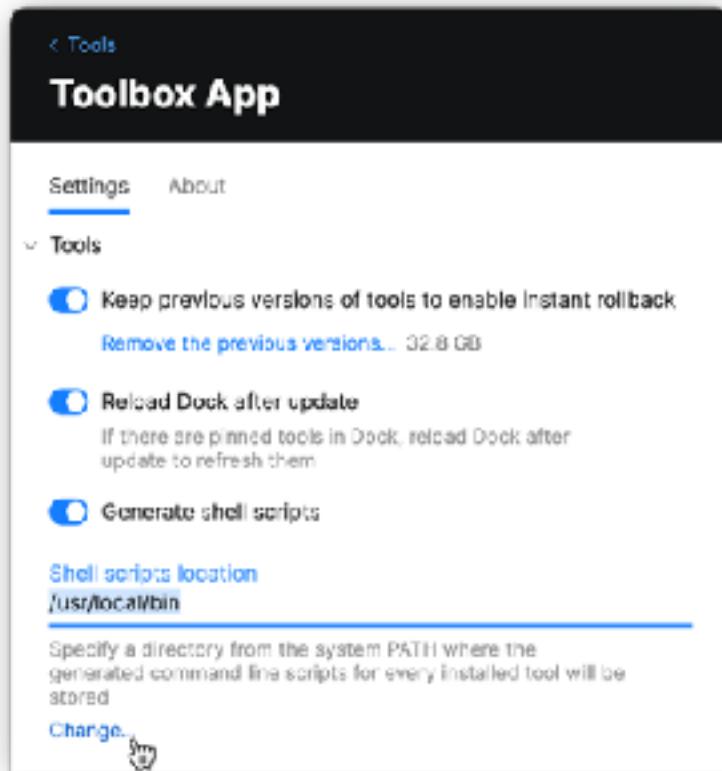
- If you want to open a project into IntelliJ from command line you can do so from command line by typing `idea .`
- After some setup...

A screenshot of a macOS terminal window. The window title is "danno@MacBook-Pro:~/Development/mine/java-tdd". The terminal prompt shows "(base) -----". Below the prompt, the command `~/Development/mine/java-tdd (answers*) » idea .` is typed. The cursor is positioned at the end of the command. The background of the terminal window is light gray, and the text is black. The window has the standard OS X title bar with red, yellow, and green buttons.



# Generating Scripts

- In your JetBrains toolbox, click on the bolt.
- Click on **Settings**
- Toggle on “Generate Shell Scripts” and either keep or override the location of the scripts
- For Windows users, put this location on your PATH





# Tool Windows



# Tool Windows & Navigation

**ALT-0...ALT-9**

Open Corresponding Tool Window

**CTRL-E**

View Recent Files, Includes Tool Windows

**CTRL-SHIFT+E**

View Recent Edited Locations

**ESC**

Return to the Editor

**CTRL+SHIFT+F12**

Maximize the Editor

**SHIFT+ESCAPE**

Close Current Tool Window

**CTRL+ALT+SHIFT+LEFT/ CTRL+ALT+SHIFT+RIGHT**

Shrink or Expand Tool Window





# Tool Windows & Navigation

**COMMAND(⌘) + 0...COMMAND(⌘) + 9**

Open Corresponding Tool Window

**COMMAND(⌘)-E**

View Recent Files, Includes Tool Windows

**COMMAND(⌘)-SHIFT+E**

View Recent Edited Locations

**ESCAPE(⌚)**

Return to the Editor

**COMMAND(⌘)+SHIFT(⇧)+F12**

Maximize the Editor

**SHIFT(⇧)+ESCAPE(⌚)**

Close Current Tool Window

**CONTROL(^) + OPTION(⌥) + LEFT(← )/**

**CONTROL(^) + OPTION(⌥) + RIGHT(→ )**

Shrink or Expand Tool Window





# Find Action



# Action Keys

**CTRL-SHIFT-A** Find Action

**CTRL-CTRL** Run Anything

**SHIFT-SHIFT** Search Everywhere





# Action Keys

**SHIFT(⇧)+COMMAND(⌘) + A** Find Action

**CONTROL(^) + CONTROL(^)** Run Anything

**SHIFT(⇧)+ SHIFT(⇧)** Search Everywhere





# Tab Management



# Tab Management Keys

**ALT+RIGHT/LEFT** Go to Next or Previous Tab

**CTRL+F4** Close Current Tab

**[Action Only]** Close All Tabs





# Tab Management Keys

**CONTROL(⌃) + RIGHT(→ )/LEFT(←)** Go to Next or Previous Tab

**COMMAND(⌘) + W** Close Current Tab

**COMMAND(⌘) +SHIFT(⇧) + W** Close All Tabs





# Finding Items



# Finding Items

**CTRL-N** Find Class

**CTRL-SHIFT-N** Find File

**CTRL-ALT-SHIFT-N** Find Symbol





# Finding Items

**COMMAND(⌘) + O** Find Class

**COMMAND(⌘) + SHIFT(⇧) + O** Find File

**COMMAND(⌘) + OPTION(⌥) + O** Find Symbol





# Dojo: Tool Windows





# Q&A



O'REILLY®

# Test Driven Development





# Test Driven Development



# Test Driven Development

1. Quickly add a test.
2. Run all tests and see the new one fail.
3. Make a little change.
4. Run all tests and see them all succeed.
5. Refactor to remove duplication.

Kent Beck – Test Driven Development By Example 2003



# Test Driven Development Benefits

- Promotes design decisions up front
- Allows you and your team to understand your code
- Model the API the way you want it to look
- Means of communicating an API before implementation
- Avoids technical debt
- Can be used with any programming language



# Prefer Tabs Over Enter



## Prefer Tab over Enter

```
@Test  
void testMath() {  
    var result = Math.abs(30);  
}  
new *
```

Tab or Enter ?

```
@Test  
void testMath() {  
    var result = Math.dabs(30);  
}
```

① **decrementExact**(int a)  
② **decrementExact**(long a)

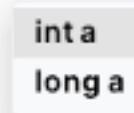
int

long



# Results with Enter

```
new *  
@Test  
void testMath() {  
    var result = Math.decrementExact(|)abs(30);  
}
```



The code completion dropdown menu shows two suggestions:  
int a  
long a



## Results with Tab

```
new *  
@Test  
void testMath() {  
    var result = Math.decrementExact(30);  
}
```



# Generating Files and Code



# Generating Files and Code

**ALT+INSERT** Generate File or Code





# Generating Files and Code

**COMMAND(⌘) + N or CONTROL(^) + RETURN(↵)**

Generate File or Code





# Navigating Errors



# Navigating Errors

**F2**

Move to the Next Highlighted Issue

**ALT+ENTER**

Propose a Fix





# Navigating Errors

F2

Move to the Next Highlighted Issue

**OPTION(⌥) + RETURN(↵)**

Propose a Fix





# Testing



# Navigating Errors

**CTRL + SHIFT + T**

Toggle Between Test and Production





# Navigating Errors

**COMMAND(⌘) + SHIFT (⇧) + T**

Toggle Between Test and Production





# Code Completion



# Code Completion

**CTRL + SPACE**

Basic Code Completion

**CTRL + ALT + SPACE**

Code Completion that shows inaccessible members

**CTRL + SHIFT + SPACE**

Type Sensitive Code Completion

**CTRL + SHIFT + ENTER**

Complete Statement

**CTRL + P**

Parameter Info

**CTRL + SHIFT+P**

Type Info





# Code Completion

**CONTROL(^) + SPACE**

Basic Code Completion

**CONTROL(^) + OPTION (⌥) + SPACE**

Code Completion that shows inaccessible members

**CONTROL(^) + SHIFT(⇧) + SPACE**

Type Sensitive Code Completion

**COMMAND(⌘) + SHIFT(⇧) + RETURN(↵)**

Complete Statement

**COMMAND(⌘) + P**

Parameter Info

**CONTROL(^) + SHIFT(⇧) + P**

Type Info





# Running



# Running

All runs are contextual, this includes main methods, tests, test folders, HTTP files, and more.

**CTRL + SHIFT + F10**

Run and mark your context configuration

**SHIFT + F10**

Run the last context configuration

**ALT+SHIFT+F10**

Select context configuration and run





# Running

All runs are contextual, this includes main methods, tests, test folders, HTTP files, and more.

**CONTROL(^) + SHIFT (^) + R**

Run and mark your context configuration

**CONTROL(^) + R**

Run the last context configuration

**CONTROL(^) + OPTION(\") + R**

Select context configuration and run





# Dojo: Test Driven Development





# Q&A





# Break

O'REILLY®

# Essential Maneuvers





# Formatting Code



# Formatting Code

**CTRL + ALT + L**

Format All Code (includes imports)

**CTRL + ALT + O**

Optimize Imports

**CTRL + ALT + I**

Format Line by Line





# Formatting Code

**COMMAND(⌘) + OPTION(⌥) + L**

Format All Code (includes imports)

**CONTROL(^) + OPTION(⌥) + O**

Optimize Imports

**CONTROL(^) + OPTION(⌥) + I**

Format Line by Line





# Selecting Code Blocks



IRVIN S. TEAWORTH, JR.'s

# THE BLOB

INDESCRIBABLE...  
INDESTRUCTIBLE...  
NOTHING CAN STOP IT!

TONY LYN presents

## THE BLOB

starring

STEVE MCQUEEN,  
ANETA CORSAUT, STEPHEN CHASE,  
EARL ROWE, OLIN HOWLIN,  
JOHN BENSON & GEORGE KARAS

12



# Selecting Code Blocks

**CTRL + W**

Increase Code Block

**CTRL + SHIFT + W**

Decrease Code Block





# Selecting Code Blocks

**OPTION(⌥) + UP(↑)**

Increase Code Block

**OPTION(⌥) + DOWN(↓)**

Decrease Code Block





# Editing Basics



# Editing Basics

**CTRL+Y**

Delete Line at Caret

**CTRL+D**

Duplicate Line at Caret

**CTRL+X**

Cut Line at Caret

**CTRL+C**

Copy Line at Caret

**CTRL+V**

Paste Line at Caret

**CTRL+SHIFT+V**

Paste From History





# Editing Basics

**COMMAND(⌘)+DELETE(⌫)**

Delete Line at Caret

**COMMAND(⌘)+D**

Duplicate Line at Caret

**COMMAND(⌘)+X**

Cut Line at Caret

**COMMAND(⌘)+C**

Copy Line at Caret

**COMMAND(⌘)+V**

Paste Line at Caret

**COMMAND(⌘)+SHIFT(⇧)+V**

Paste From History





# Surround Blocks and Selections



# Surround Blocks and Selections

**CTRL + ALT + T**

Surround Block or Selection





# Surround Blocks and Selections

**COMMAND(⌘) + OPTION(⌥) + T**

Surround Block or Selection





# Refactoring



# Refactoring (Part 1)

**CTRL + ALT+ SHIFT + T**

Refactor This...

**F5**

Copy item

**F6**

Move item

**SHIFT+F6**

Rename globally

**ALT+DELETE**

Safe Delete





# Refactoring (Part 2)

**Ctrl + Alt + N**

Inline

**Ctrl + Alt + M**

Extract Method

**Ctrl + Alt + V**

Extract Variable

**Ctrl + Alt + F**

Extract Field

**Ctrl + Alt + C**

Extract Constant

**Ctrl + Alt + P**

Extract Parameter





# Refactoring (Part 1)

**CONTROL(⌃) + T**

Refactor This

**F5**

Copy item

**F6**

Move item

**SHIFT(⇧)+F6**

Rename globally

**COMMAND(⌘)+DELETE**

Safe Delete





# Refactoring (Part 2)

**COMMAND(⌘) + OPTION(⌥) + N**

Inline

**COMMAND(⌘) + OPTION(⌥) + M**

Extract Method

**COMMAND(⌘) + OPTION(⌥) + V**

Extract Variable

**COMMAND(⌘) + OPTION(⌥) + F**

Extract Field

**COMMAND(⌘) + OPTION(⌥) + C**

Extract Constant

**COMMAND(⌘) + OPTION(⌥) + P**

Extract Parameter





# Dojo: Editing and Refactoring



O'REILLY®

# Navigating Code





# Usages



# Usages

**CTRL + ALT + B**

Go to Implementation

**CTRL + B**

Go to Declaration

**ALT + F7**

Find Usages

**CTRL + F7**

Find in Files

**CTRL + ALT + LEFT / RIGHT**

Navigate back / forward





# Usages

**COMMAND(⌘) + OPTION(⌥) + B**

Go to Implementation

**COMMAND(⌘) + B**

Go to Declaration

**OPTION (⌥) + F7**

Find Usages

**COMMAND(⌘) + SHIFT(⇧) + F7**

Find In Files

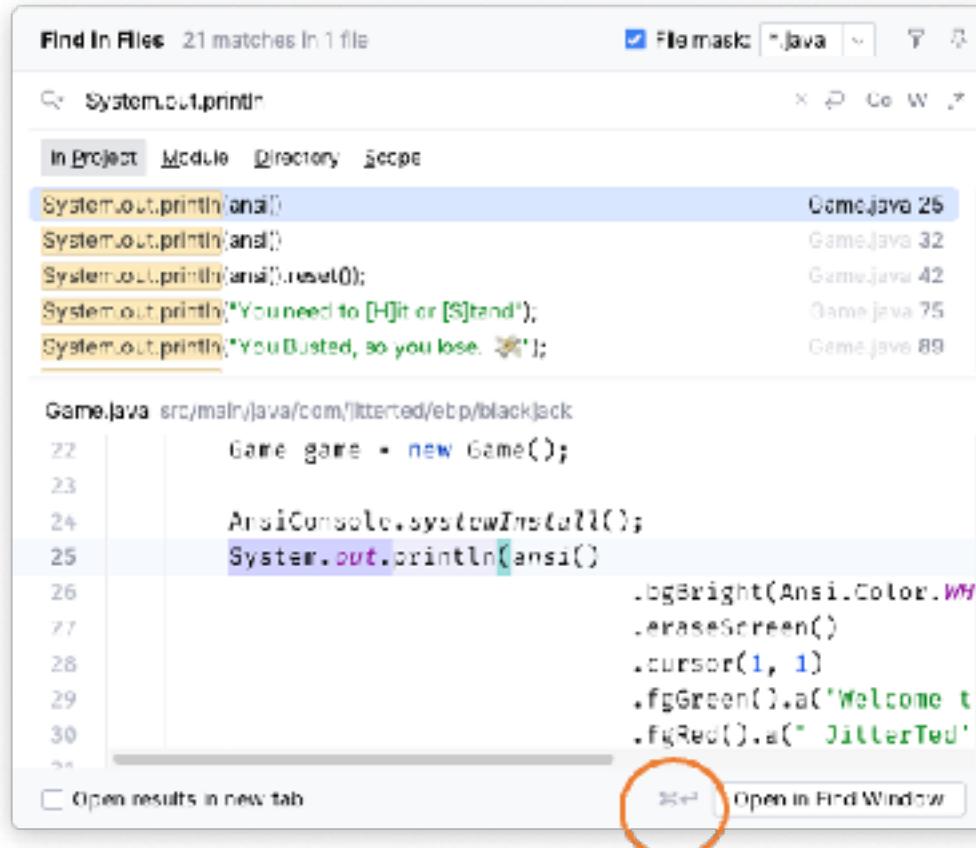
**COMMAND(⌘) + OPTION(⌥) + LEFT(←) / RIGHT(→)**

Navigate back / forward





# Route “Find in Files” to Find Window





# Moving Code



# Moving Code

**CTRL + SHIFT + UP/DOWN**

Move either line or method, depending on caret location





# Moving Code

**SHIFT (⇧) + COMMAND(⌘) + UP(↑)/DOWN(↓)**

Move either line or method, depending on caret location





# Code Documentation and Help



# Code Documentation and Help

**CTRL + Q**

Quick Documentation Lookup

**CTRL + SHIFT + I**

Quick Definition Lookup

**ALT+Q**

Context Info

**CONTROL(^) + SHIFT (^) + P**

Type Info (Twice to show more)





# Code Documentation and Help

**CONTROL(^) + J**

Quick Documentation

**OPTION(\u2190) + SPACE or COMMAND (⌘) + Y**

Quick Definition Lookup

**CONTROL(^) + SHIFT (^) + Q**

Context Info

**CONTROL(^) + SHIFT (^) + P**

Type Info (Twice to show more)





# Hierarchies



# Hierarchies

## **CTRL+H**

Type Hierarchy - Show Parent and Child Classes

## **CTRL+SHIFT+H**

Method Hierarchy – Show subclasses where the method overrides the selected method as well as what is overridden

## **CTRL+ALT+H**

Call Hierarchy – Callers (Supertypes) or Callees (Subtypes) of a method, if on a field it will show the methods that interact with it





# Hierarchies

## **CONTROL(^) + H**

Type Hierarchy - Show Parent and Child Classes

## **COMMAND (⌘) + SHIFT(⇧) + H**

Method Hierarchy – Show subclasses where the method overrides the selected method as well as what is overridden

## **CONTROL(^) + OPTION(⌥) + H**

Call Hierarchy – Callers (Supertypes) or Callees (Subtypes) of a method, if on a field it will show the methods that interact with it





# Structure



# Structure

**ALT + 7**

View the Structure of the Class

**CTRL+F12**

File Structure Popup





# Structure

**COMMAND(⌘) + 7**

View the Structure of the Class

**COMMAND(⌘) +F12**

File Structure Popup





# Dojo: Navigating

# Additional Cool Stuff





# Debugging



# Debugging

## **CTRL+F8**

Toggle Breakpoint

## **CTRL + SHIFT+ F8**

View Breakpoints

## **CTRL+SHIFT+F9**

Debug and Mark your Context Configuration

## **SHIFT+F9**

Debug the Last Context Configuration

## **ALT+SHIFT+F9**

Select Context Configuration and Debug





# Debugging

**COMMAND(⌘) + F8**

Toggle Breakpoint

**COMMAND(⌘) + SHIFT (⇧) + F8**

View Breakpoints

**CONTROL(^) + SHIFT (⇧) + D**

Debug and Mark your Context Configuration

**CONTROL(^) + D**

Debug the Last Context Configuration

**CONTROL(^) + OPTION(⌥) + R**

Select Context Configuration and Debug





# Live Templates



# Live Templates

- Use live templates to insert common constructs into your code, such as:
  - Loops
  - Conditions
  - Various Declarations
  - Print Statements
- Configure Live Templates by going to **[Settings > Editor > Live Templates]**
- You can find **Live Templates** by using **Find Action**





# Live Template Variables

- \$END\$ indicates the position of the caret when the code snippet is complete, and you can no longer press Tab to jump to the next variable.
- \$SELECTION\$ is used in surround templates and denotes the code fragment to be wrapped. After the template expands, it wraps the selected text as specified in the template.





# Live Template Functions

- IntelliJ has wide selection of [Live Template Functions](#) available
- The functions are used with your templates to refine what to do with live template variable. Functions include:
  - enum
  - time
  - suggestIndexName
  - snakeCase
  - more..
- You can use **Edit Template Variables** to add these custom expressions.





# File Templates



# File Templates

- Instead of code snippets that can be created in a live template, a *File Template*
- Provides initial code as a file, this can be anything from:
  - Corporate Templates
  - Favorite Test File Structure
  - Model, View, Controller Templates
  - Common Constructs from your favorite framework
- Configure Live Templates by going to going to  
**[Settings > Editor > File and Code Templates]**
- You can find **File Templates** by using **Find Action**





# File Template Variables

- Variables, are also available for file templates
- This can be used to substitute pertinent information into your template at the time of creation
- Variables include
  - \${DATE}
  - \${PACKAGE\_NAME}
  - \${FILE\_NAME}
  - more..





# Multicursors



# Multicursors

## **ALT+SHIFT+ [Click]**

Start Multicursors

Delete Single Multicursor

Note: Can be used in selections (blobs) as well

## **ALT+J**

Select Next Word Occurrence

## **ALT+SHIFT+J**

Select All Word Occurrences

Can be used with Find...

## **ALT + SHIFT + G**

Add Carets At The End of Lines in Selected Block





# Multicursors

**OPTION(⌥) + SHIFT(⇧) + [Click]**

Start Multicursors

Delete Single Multicursor

Note: Can be used with selections (blobs) as well

**CONTROL(^) + G**

Select Next Word Occurrence

**CONTROL(^) + COMMAND(⌘) + G**

Select All Word Occurrences

Can be used with Find...

**OPTION(⌥) + SHIFT(⇧) + G**

Add Carets At The End of Lines in Selected Block





# Column Selection Mode



# Column Selection Mode

**ALT+SHIFT+INSERT**

Column Selection Mode

**CTRL+CTRL**

Column Selection Mode (Alternate)





# Column Selection Mode

**SHIFT(⇧) + COMMAND(⌘) + 8**

Column Selection Mode

**OPTION(⌥) + OPTION(⌥)**

Column Selection Mode (Alternate)





# HTML/XML Editing



# HTML/XML Editing with Emmet

- <http://docs.emmet.io/abbreviations/syntax/>
- <https://docs.emmet.io/cheat-sheet/>



# Version Control



# Version Control

**ALT + 0**

Version Control Window

**CTRL + K**

Commit

**CTRL+T**

Update Project

**CTRL + SHIFT + K**

Push

**ALT + ` (Backquote)**

Version Control Quick Help





# Version Control

**COMMAND(⌘) + 0**

Version Control Window

**COMMAND(⌘) + K**

Commit

**COMMAND(⌘) + T**

Update Project

**COMMAND(⌘) + SHIFT(⇧) + K**

Push

**CONTROL(^) + V**

Version Control Quick Help





# Version Control Conflicts



# Version Control Conflicts

Merge Branches from PRs with Conflicting Changes					
PR #	Author	Commit Message	Branch	Conflicts	Actions
1	John Doe	Create initial commit	main	-	Accept
2	Jane Smith	Add feature A	feature-A	-	Accept
3	John Doe	Add feature B	feature-B	-	Accept
4	John Doe	Update feature A	feature-A	-	Accept
5	Jane Smith	Update feature A	feature-A	-	Accept
6	John Doe	Update feature B	feature-B	-	Accept
7	Jane Smith	Update feature B	feature-B	-	Accept
8	John Doe	Introduce conflict	feature-C	-	Accept
9	Jane Smith	Introduce conflict	feature-C	-	Accept
10	John Doe	Fix conflict in feature-C	feature-C	-	Accept
11	Jane Smith	Fix conflict in feature-C	feature-C	-	Accept
12	John Doe	Push changes to main	main	-	Accept
13	Jane Smith	Push changes to main	main	-	Accept
14	John Doe	Rebase feature-A onto main	feature-A	-	Accept
15	Jane Smith	Rebase feature-B onto main	feature-B	-	Accept
16	John Doe	Create PR for feature-C	feature-C	-	Accept
17	John Doe	Push changes to feature-C	feature-C	-	Accept
18	Jane Smith	Push changes to feature-C	feature-C	-	Accept
19	John Doe	Rebase feature-C onto main	main	-	Accept
20	Jane Smith	Rebase feature-C onto main	main	-	Accept
21	John Doe	Push changes to main	main	-	Accept
22	Jane Smith	Push changes to main	main	-	Accept
23	John Doe	Rebase feature-A onto main	main	-	Accept
24	Jane Smith	Rebase feature-B onto main	main	-	Accept
25	John Doe	Rebase feature-C onto main	main	-	Accept
26	Jane Smith	Push changes to main	main	-	Accept
27	John Doe	Rebase feature-A onto main	main	-	Accept
28	Jane Smith	Rebase feature-B onto main	main	-	Accept
29	John Doe	Rebase feature-C onto main	main	-	Accept
30	Jane Smith	Push changes to main	main	-	Accept
31	John Doe	Rebase feature-C onto main	main	-	Accept
32	Jane Smith	Push changes to main	main	-	Accept
33	John Doe	Rebase feature-C onto main	main	-	Accept
34	Jane Smith	Push changes to main	main	-	Accept



# Accept or Merge

Conflicts

Merging branch **Discover\_IDEA** into branch **check\_lost\_content**

Name	Yours (check...)	Theirs (Discov...	
current.help.version ~/IdeaProjects/help-sources	Modified	Modified	<b>Accept Yours</b>
Discover_IntelliJ_IDEA.xml ~/IdeaProjects/help-sources/intellij-pl	Modified	Modified	<b>Accept Theirs</b>
.tree ~/IdeaProjects/help-sources/intellij-platform	Modified	Modified	
Manage_branches.xml ~/IdeaProjects/help-sources/intellij-platf	Modified	Modified	
project.ihp ~/IdeaProjects/help-sources/intellij-platform	Modified	Modified	<b>Merge...</b>

Group files by directory

**Close**



# Color Meanings

Merge Revisions for /Users/anna.gasparyan/ideaProjects/help-sources/intellij-platform/topics/Big\_Data\_Tools.Configuration.xml

Apply non-conflicting changes: >< Left >< All << Right / | Don't ignore | Highlight words | ? | 8 changes. 1 conflict.

Your version		Result		Changes from server (revision ad835f2ba...)	
①	<tab title="Zeppelin"	47	48	Mandatory parameters:</p>	49
	<list>	X	49	<list>	58
	<li><control>URL</c	50	51	<li><control>URL</cont	51
	<li><control>Login a	51	52	<li><control>Login</c>	52
	</list>	52	53	</list>	53
	<p> Optionally, you can s	X	53	<p> Optionally, you can set	54
	<list>	54	55	<list>	55
	<li><control>Name</c	55	56	<li><control>Name</cont	56
	<li><control>Login as	56	57	<li><control>Login as a	57
	<li><control>Enable co	X	57	<li><control>Per project	58
	By default, all	58	59	Deselect it if you	59
	<li><control>Per pro	59	60	<li><control>Enable con	60
	Deselect it if y	60	61	By default, the new	61
	<li><control>Scala V	X	61	<li><control>Scala Vers	62
	plugin bundles.	62	63	plugin bundles. If	63
	<li><control>Enable	63	64	<li><control>Enable HTT	64
	connection with	64	65	connection with the	65

Deleted line

Modified line

Newly added line

Conflicting changes

Accept Left | Accept Right | Cancel | Apply

# Accepting Conflicts

```
    46    47      public re
    47    48      }
    48    49
  X 49  50      public
  Accept 51      re
  52    53      }
  53    54      public
  middleName, 54  55      re
  55    56      }
  56    57
  57    58      @Override
  58    59 *t  public
```

```
  hDay, LocalDa 32
                  33
                  34
String middl 35 @
  , birthDay, l 36 ✕X
  Accept
  39
  40
  41
  42      }
```



# Ignoring Conflicts

```
    45    46    47    pub
    46    47    48    }
    47    48    49    }
    48    49    50    < Ignore 0  pub
    50    51    51    }
    51    52    52    }
    52    53    53    > pub
    53    54    54    }
    54    55    55    }
    55    56    56    }
    56    57    57    }
    57    58    58    @Override
    58    59    59    pub
    59    60    60    if (this == null)
    60    61    61    return null;
    61    62    62    else
    62    63    63    if (o == null)
    63    64    64    return null;
    64    65    65    else
    65    66    66    if (this.getClass() != o.getClass())
    66    67    67    return false;
    67    68    68    for (Field field : getClass().getDeclaredFields())
    68    69    69    if (!field.getName().equals("name"))
    69    70    70    if (!field.get(this).equals(field.get(o)))
    70    71    71    return false;
    71    72    72    return true;
    72    73    73    }
```

```
    47
    48    public Optional<String> getName() {
    49    return Optional.ofNullable(name);
    50    }
    51
    52    public void setName(String name) {
    53        this.name = name;
    54    }
    55    public String getMiddleName() {
    56        return middleName;
    57    }
    58    public void setMiddleName(String middleName) {
    59        this.middleName = middleName;
    60    }
    61    public String getLastName() {
    62        return lastName;
    63    }
    64    public void setLastName(String lastName) {
    65        this.lastName = lastName;
    66    }
    67    public Date getBirthDate() {
    68        return birthDate;
    69    }
    70    public void setBirthDate(Date birthDate) {
    71        this.birthDate = birthDate;
    72    }
    73    public String toString() {
    74        return "Person{" +
    75            "name=" + name +
    76            ", middleName=" + middleName +
    77            ", lastName=" + lastName +
    78            ", birthDate=" + birthDate +
    79            '}';
    80    }
    81}
```

sDate.get() 53 << X

Ignore  
Ctrl+click to resolve conflict



# Version Control Conflicts

F7

Next Difference

**SHIFT+F7**

Previous Difference





# Version Control Conflicts

F7

Next Difference

SHIFT(↑) + F7

Previous Difference





# Dojo: Multicursors and Templates

# Conclusion





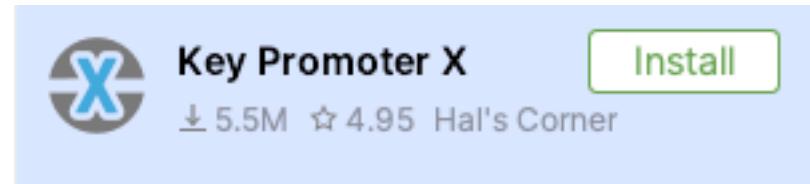
# Continual Learning



# You don't have to grok all this in one day

- There is a lot to take in when learning a new editor and becoming efficient
- This presentation is to
  - Show all the possibilities. Even if you don't have it all memorized
  - Serve as a reference so you can know what to demand from other editors/IDEs
- When using IntelliJ after today, some things to consider:
  - Keep a personal cheat sheet handy, but only the items that you don't have memorized
  - Keep in the back of your mind that there must be a key binding, and look it up using **Find Action**
  - Try **Key Promoter X**
  - For Presentations try **Presentation Assistant**
- **Subscribe to the JetBrains IntelliJ channel on YouTube:**

<https://www.youtube.com/@intellijidea>





# AI and IntelliJ IDEA



# AI Assistant

- JetBrains now offers AI solutions as either a trial or as an ultimate package
- AI Assistant Services include:
  - Chat Question and Answers
  - Assistance in writing tests, commit messages, and documents
  - Research common bugs
  - Ask for solutions to common problems
  - Provide naming assistance
  - Provide refactoring help
- Levels
  - Trial - Equal to the Pro Plan, but with limitations
  - Pro - \$8.33 per month/\$100.00 per year
  - Enterprise - Multi seat pro plan and customization





## Poll Question

What are some of the favorite things  
that you've learned?



# Q&A



O'REILLY®