IntelliJ Wizardry with Heinz Kabutz

Dr Heinz M. Kabutz

Last Updated 2022-01-19

Javaspecialists.eu java training

Copyright Notice

1: Introduction

Why IntelliJ IDEA?

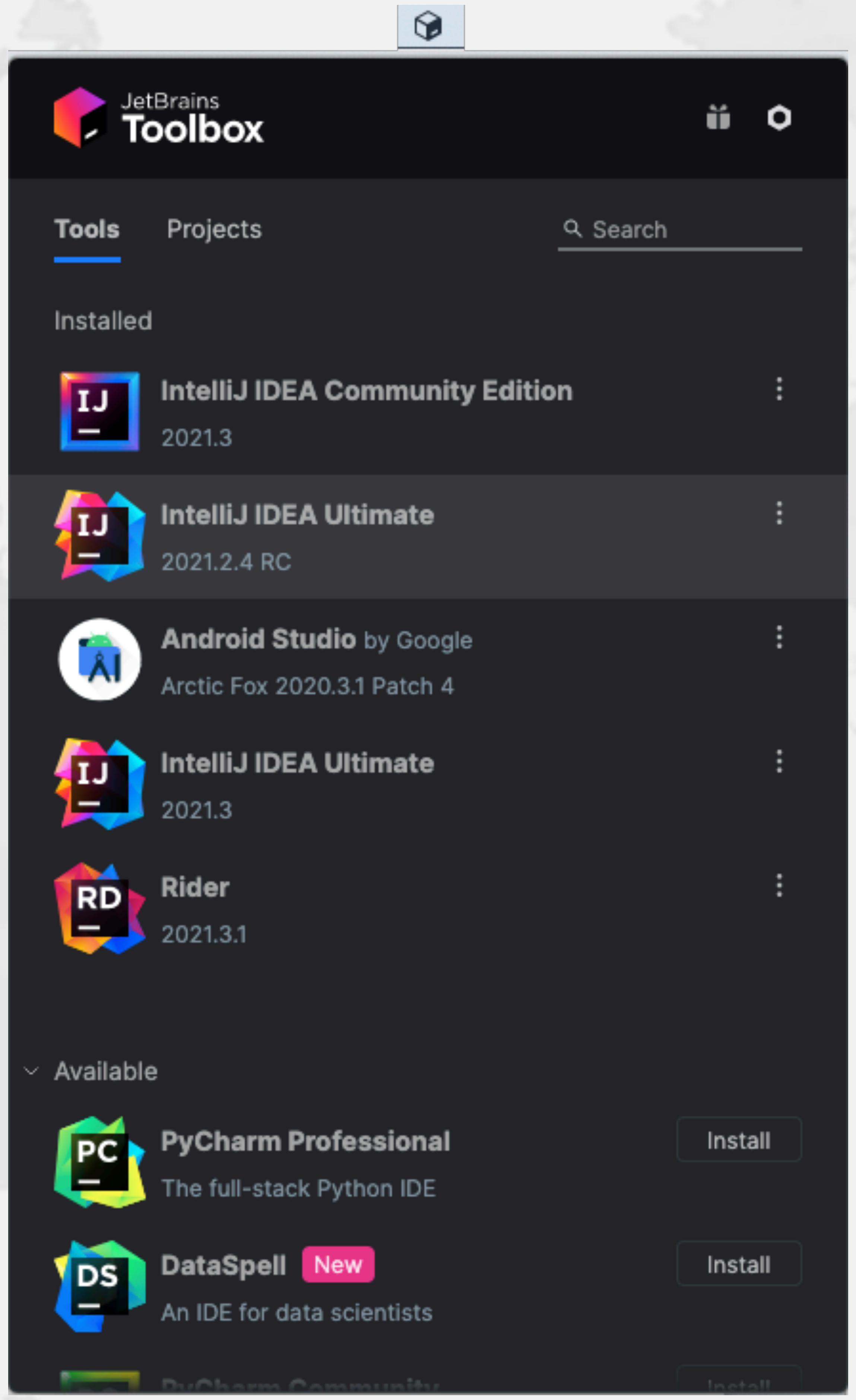
- My story
  - Started with Borland JBuilder
  - Then used Eclipse for a year
  - Needed something to work with horrible messy Java code
    - And create some Swing GUIs at the same time
  - Downloaded IntelliJ IDEA
    - No free version at the time
    - Used it for 30 days
    - Bought it
  - Paid twice
    - Once for the license and then reduced hours worked
      - But work was far more pleasant, less frustration, better life

2: Setting Up IntelliJ IDEA

Javaspecialists.eu
java training

Download JetBrains Toolbox

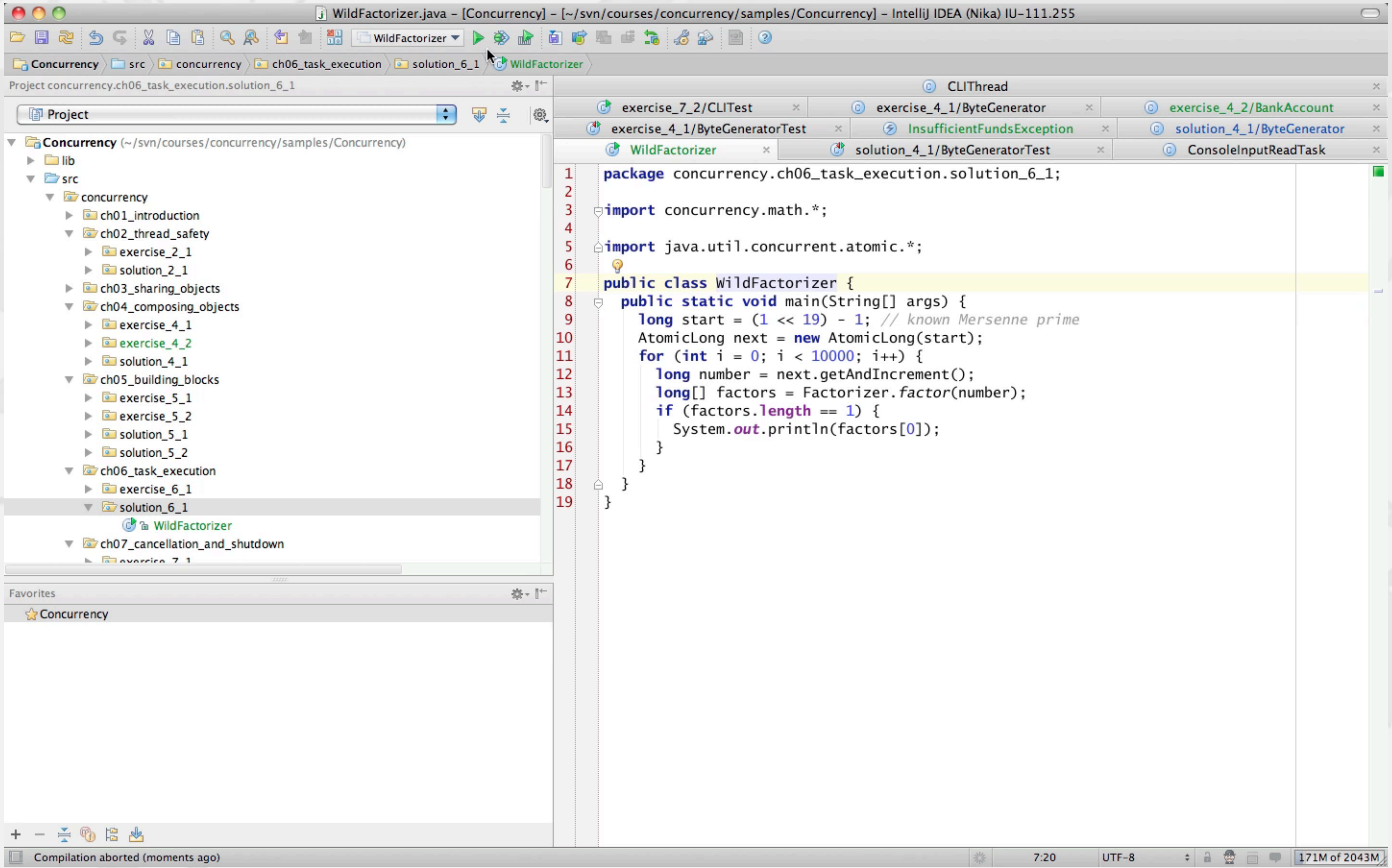
- **Keeps the IDEs up to date**
  - **https://www.jetbrains.com/toolbox-app/**



Setting up IntelliJ IDEA Community Edition

- **Editor →**
  - **General → Appearance: Show method separators**
  - **General → Code folding**
  - **General → Smart Keys: Select Use "CamelHumps" words**
  - **Color Scheme: Darcula Courses better for showing errors**
  - **Live Templates (more about this later)**

Livelock from Corrupt Local Files

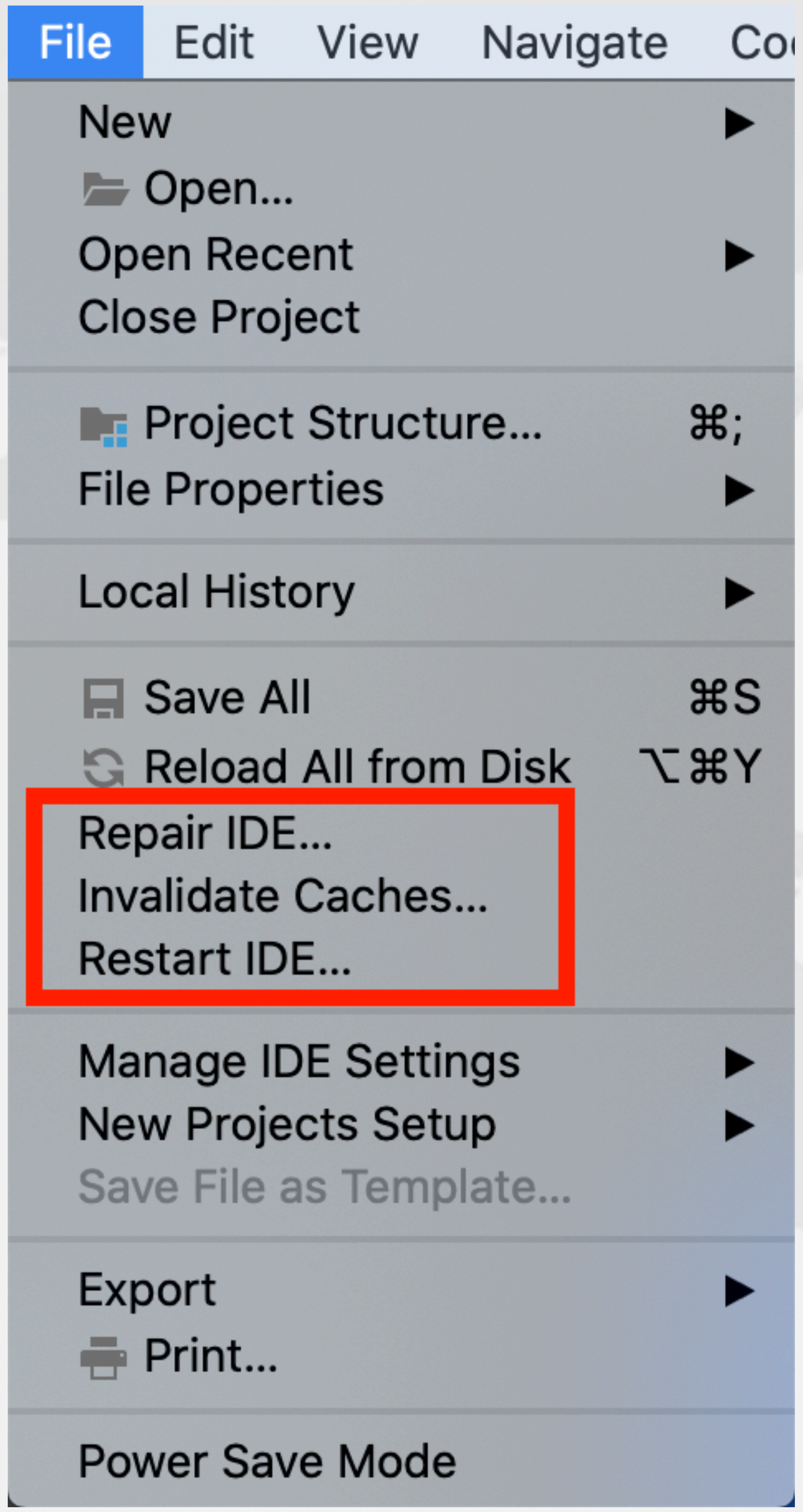

WildFactorizer.java – [Concurrency] – [~/svn/courses/concurrency/samples/Concurrency] – IntelliJ IDEA (Nika) IU-111.255

```
package concurrency.ch06_task_execution.solution_6_1;

import concurrency.math.*;

import java.util.concurrent.atomic.*;

public class WildFactorizer {
  public static void main(String[] args) {
    long start = (1 << 19) - 1; // known Mersenne prime
    AtomicLong next = new AtomicLong(start);
    for (int i = 0; i < 10000; i++) {
      long number = next.getAndIncrement();
      long[] factors = Factorizer.factor(number);
      if (factors.length == 1) {
        System.out.println(factors[0]);
      }
    }
  }
}
```

Compilation aborted (moments ago)

Recovering from Broken Indexes

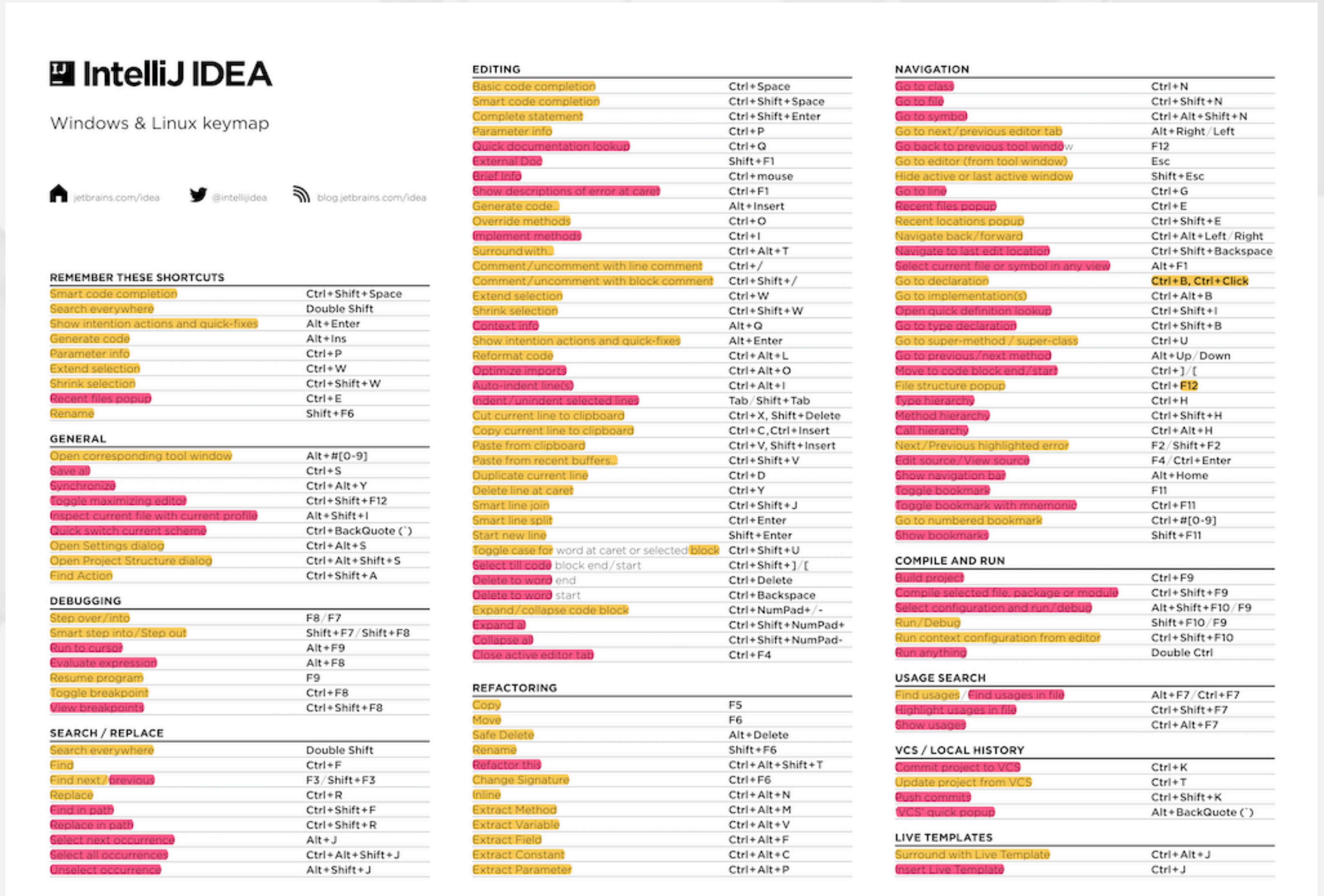
- IntelliJ gets into a bad state if indexes are corrupted
  - File → Repair IDE...
    - Try this first, stepping through until everything works
  - File → Invalidate Caches...
    - Usually takes a while to reindex
      - But solves most of the problems
      - Try to not lose local history



3. IntelliJ IDEA Philosophy

IntelliJ IDEA Philosophy

- **IntelliJ designed to be used mostly without mouse**
- **Hotkeys for almost everything**
  - **Print out resources/IntelliJIDEA_ReferenceCard.pdf**
  - **Memorize one new shortcut per day ≈ 6 months**



IntelliJ IDEA

Windows & Linux keymap

jetbrains.com/idea @intellijidea blog.jetbrains.com/idea

REMEMBER THESE SHORTCUTS

Smart code completion	Ctrl+Shift+Space
Search everywhere	Double Shift
Show intention actions and quick-fixes	Alt+Enter
Generate code	Alt+Ins
Parameter info	Ctrl+P
Extend selection	Ctrl+W
Shrink selection	Ctrl+Shift+W
Recent files popup	Ctrl+E
Rename	Shift+F6

GENERAL

Open corresponding tool window	Alt+#[0-9]
Save all	Ctrl+S
Synchronize	Ctrl+Alt+Y
Toggle maximizing editor	Ctrl+Shift+F12
Inspect current file with current profile	Alt+Shift+I
Quick switch current scheme	Ctrl+BackQuote (`)
Open Settings dialog	Ctrl+Alt+S
Open Project Structure dialog	Ctrl+Alt+Shift+S
Find Action	Ctrl+Shift+A

DEBUGGING

Step over / into	F8 / F7
Smart step into / Step out	Shift+F7 / Shift+F8
Run to cursor	Alt+F9
Evaluate expression	Alt+F8
Resume program	F9
Toggle breakpoint	Ctrl+F8
View breakpoints	Ctrl+Shift+F8

SEARCH / REPLACE

Search everywhere	Double Shift
Find	Ctrl+F
Find next / previous	F3 / Shift+F3
Replace	Ctrl+R
Find in path	Ctrl+Shift+F
Replace in path	Ctrl+Shift+R
Select next occurrence	Alt+J
Select all occurrences	Ctrl+Alt+Shift+J
Unselect occurrence	Alt+Shift+J

EDITING

Basic code completion	Ctrl+Space
Smart code completion	Ctrl+Shift+Space
Complete statement	Ctrl+Shift+Enter
Parameter info	Ctrl+P
Quick documentation lookup	Ctrl+Q
External Doc	Shift+F1
Brief Info	Ctrl+mouse
Show descriptions of error at caret	Ctrl+F1
Generate code...	Alt+Insert
Override methods	Ctrl+O
Implement methods	Ctrl+I
Surround with...	Ctrl+Alt+T
Comment / uncomment with line comment	Ctrl+/
Comment / uncomment with block comment	Ctrl+Shift+/
Extend selection	Ctrl+W
Shrink selection	Ctrl+Shift+W
Context info	Alt+Q
Show intention actions and quick-fixes	Alt+Enter
Reformat code	Ctrl+Alt+L
Optimize imports	Ctrl+Alt+O
Auto-indent line(s)	Ctrl+Alt+I
Indent / unindent selected lines	Tab / Shift+Tab
Cut current line to clipboard	Ctrl+X, Shift+Delete
Copy current line to clipboard	Ctrl+C, Ctrl+Insert
Paste from clipboard	Ctrl+V, Shift+Insert
Paste from recent buffers...	Ctrl+Shift+V
Duplicate current line	Ctrl+D
Delete line at caret	Ctrl+Y
Smart line join	Ctrl+Shift+J
Smart line split	Ctrl+Enter
Start new line	Shift+Enter
Toggle case for word at caret or selected block	Ctrl+Shift+U
Select till code block end / start	Ctrl+Shift+] / [
Delete to word end	Ctrl+Delete
Delete to word start	Ctrl+Backspace
Expand / collapse code block	Ctrl+NumPad+ / -
Expand all	Ctrl+Shift+NumPad+
Collapse all	Ctrl+Shift+NumPad-
Close active editor tab	Ctrl+F4

REFACTORING

Copy	F5
Move	F6
Safe Delete	Alt+Delete
Rename	Shift+F6
Refactor this	Ctrl+Alt+Shift+T
Change Signature	Ctrl+F6
Inline	Ctrl+Alt+N
Extract Method	Ctrl+Alt+M
Extract Variable	Ctrl+Alt+V
Extract Field	Ctrl+Alt+F
Extract Constant	Ctrl+Alt+C
Extract Parameter	Ctrl+Alt+P

NAVIGATION

Go to class	Ctrl+N
Go to file	Ctrl+Shift+N
Go to symbol	Ctrl+Alt+Shift+N
Go to next / previous editor tab	Alt+Right / Left
Go back to previous tool window	F12
Go to editor (from tool window)	Esc
Hide active or last active window	Shift+Esc
Go to line	Ctrl+G
Recent files popup	Ctrl+E
Recent locations popup	Ctrl+Shift+E
Navigate back / forward	Ctrl+Alt+Left / Right
Navigate to last edit location	Ctrl+Shift+Backspace
Select current file or symbol in any view	Alt+F1
Go to declaration	Ctrl+B, Ctrl+Click
Go to implementation(s)	Ctrl+Alt+B
Open quick definition lookup	Ctrl+Shift+I
Go to type declaration	Ctrl+Shift+B
Go to super-method / super-class	Ctrl+U
Go to previous / next method	Alt+Up / Down
Move to code block end / start	Ctrl+] / [
File structure popup	Ctrl+F12
Type hierarchy	Ctrl+H
Method hierarchy	Ctrl+Shift+H
Call hierarchy	Ctrl+Alt+H
Next / Previous highlighted error	F2 / Shift+F2
Edit source / View source	F4 / Ctrl+Enter
Show navigation bar	Alt+Home
Toggle bookmark	F11
Toggle bookmark with mnemonic	Ctrl+F11
Go to numbered bookmark	Ctrl+#[0-9]
Show bookmarks	Shift+F11

COMPILE AND RUN

Build project	Ctrl+F9
Compile selected file, package or module	Ctrl+Shift+F9
Select configuration and run / debug	Alt+Shift+F10 / F9
Run / Debug	Shift+F10 / F9
Run context configuration from editor	Ctrl+Shift+F10
Run anything	Double Ctrl

USAGE SEARCH

Find usages / Find usages in file	Alt+F7 / Ctrl+F7
Highlight usages in file	Ctrl+Shift+F7
Show usages	Ctrl+Alt+F7

VCS / LOCAL HISTORY

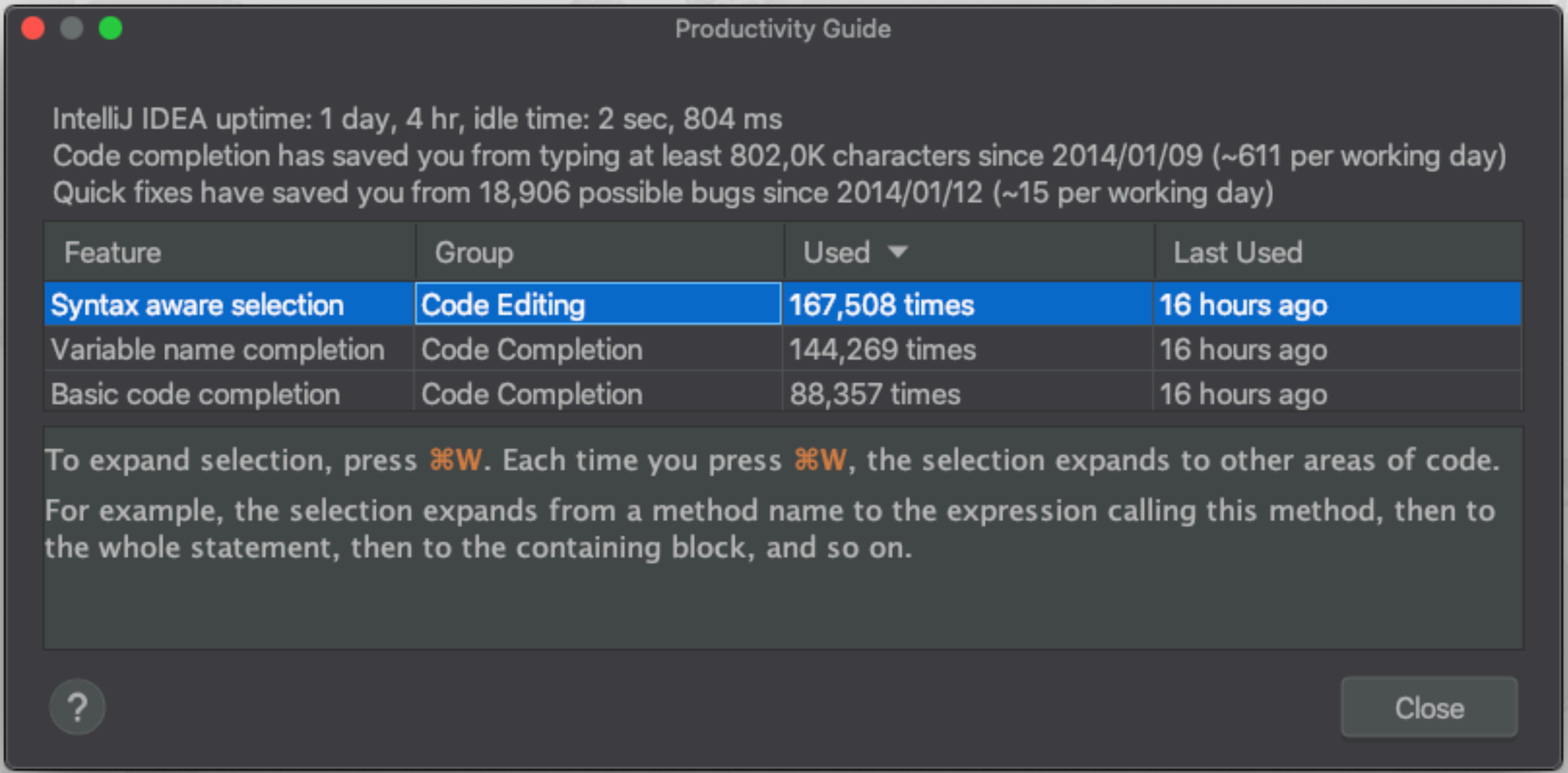
Commit project to VCS	Ctrl+K
Update project from VCS	Ctrl+T
Push commits	Ctrl+Shift+K
'VCS' quick popup	Alt+BackQuote (`)

LIVE TEMPLATES

Surround with Live Template	Ctrl+Alt+J
Insert Live Template	Ctrl+J

Help → My Productivity

- Track progress in how productive you have become



Productivity Guide

IntelliJ IDEA uptime: 1 day, 4 hr, idle time: 2 sec, 804 ms
Code completion has saved you from typing at least 802,0K characters since 2014/01/09 (~611 per working day)
Quick fixes have saved you from 18,906 possible bugs since 2014/01/12 (~15 per working day)

Feature	Group	Used ▾	Last Used
Syntax aware selection	Code Editing	167,508 times	16 hours ago
Variable name completion	Code Completion	144,269 times	16 hours ago
Basic code completion	Code Completion	88,357 times	16 hours ago

To expand selection, press ⌘W. Each time you press ⌘W, the selection expands to other areas of code.

For example, the selection expands from a method name to the expression calling this method, then to the whole statement, then to the containing block, and so on.

Searching

- "Search Everywhere"
  - Windows/Linux: Double Shift
  - Mac OS X: ⇧⇧
- "Search Everywhere and Include non-project items"
  - Windows/Linux: Quadruple Shift
  - Mac OS X: ⇧⇧⇧⇧
- Kept on hitting this by mistake when pressing ↑

Tool Windows

- Project Tool Window
  - Windows/Linux: Alt + 1
  - Mac OS X: ⌘1
- Terminal Tool Window
  - Windows/Linux: Alt + F12
  - Mac OS X (Official): ⌥ F12
  - Mac OS X (Heinz): ⌘2
- Run Tool Window
  - Windows/Linux: Alt + 4
  - Mac OS X: ⌘4

Tool Windows

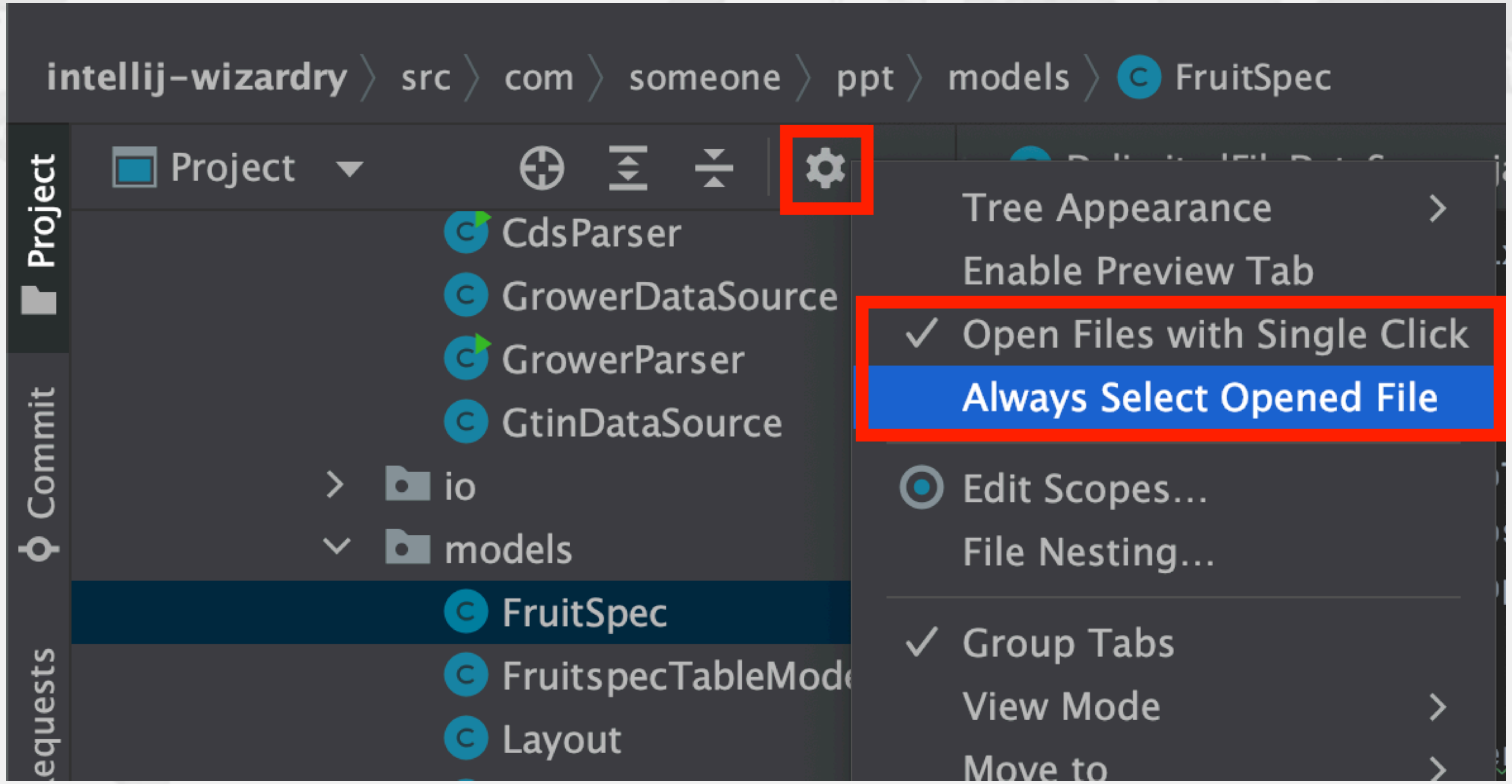
- **Debug Tool Window**
  - **Windows/Linux: Alt + 5**
  - **Mac OS X: ⌘5**
- **Structure Tool Window**
  - **Windows/Linux: Alt + 7 or Ctrl + F12 for popup**
  - **Mac OS X: ⌘7 or ⌘F12 for popup**

Switching Between Editor & Tool Windows

- "Go to editor" (from tool window)
  – Windows/Linux: Esc
  – Mac OS X: ⎋
- "Hide active or last active window"
  - Only the last one, not all of them
  – Windows/Linux: Shift + Esc
  – Mac OS X: ⇧⎋
- "Toggle maximizing editor"
  – Windows/Linux: Ctrl + Shift + F12
  – Mac OS X: ⌘⇧F12

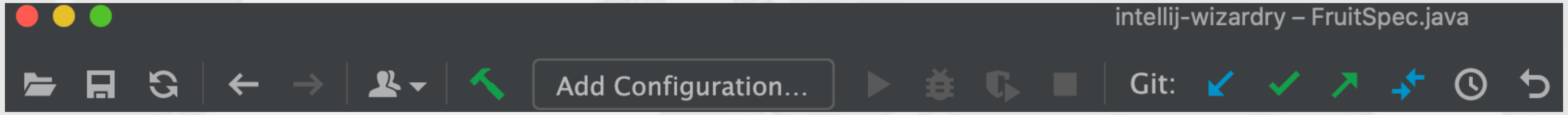
Autoscroll to/from source

- Should be on by default
  - Lots of times saw programmers editing the wrong file
  - ✓ Open Files with Single Click
  - ✓ Always Select Opened File



View → Appearance → Toolbar

- **Shows a useful toolbar at the top**



4. Essential Shortcuts

Superkey for fixing almost anything

- "Show intention actions and quick-fixes"
  - Windows/Linux: Alt + Enter
  - Mac OS X: ⌥↩
- Quick demo fixing com.someone.ppt.cds.CdsGenerator

Generate new code

- "Generate Code"
  - Windows/Linux: Alt + Ins
  - Mac OS X (Official): ⌘N
  - Mac OS X (Heinz): ^↩ or ^N
- Quick demo creating playground.Demo
  - Add final field, show difference between ^↩ and ⌥↩

Live Templates

- **We can generate code quickly with live templates**
  - psvm or main: Main method
  - sout, soutv, soutm, soutp, souf, serr, soutc: Output
  - iter, fori, itco, itar, ritar: Iteration
  - ifn, inn: if == null / if != null
  - prsf, psf, psfi, psfs: private/public final fields

Navigation

- "Go to declaration"
  - Windows/Linux: Ctrl + B or Ctrl + Click
  - Mac OS X: ⌘B or ⌘Click
- "Go to super-method / super-class"
  - Windows/Linux: Ctrl + U
  - Mac OS X: ⌘U
- "Go to implementation(s)"
  - Windows/Linux: Ctrl + Alt + B
  - Mac OS X: ⌘⌥B

Should you throw away your mouse?

- **Everything can be done with keyboard in IDEA**
  - **It is useful to learn to touch type**
  - **I usually have left hand on keyboard and right on mouse**
    - **Easy enough to find the correct keys - index fingers on F & J**
- **For navigating, I find the mouse faster**
  - **Hold down Ctrl or ⌘ and everything becomes a hyperlink**
  - **Scrolling with mouse or touchpad smoother**

Find Usages

- Find Usages
  - Windows/Linux: Alt + F7
  - Mac OS X: ⌥F7
- We can specify the scope of where to search ⚙
  - All places
  - Project files
  - Project and libraries
  - Recently viewed files
  - etc.
- "Open results in new tab" also quite useful

Back and Forth Navigation

- **"Navigate back / forward"**
  - **Windows/Linux: Ctrl + Alt + Left / Right**
  - **Mac OS X: ⌘⌥← / ⌘⌥→**
- **"Recent files/locations popup"**
  - **Windows/Linux: Ctrl + E / Ctrl + Shift + E**
  - **Mac OS X: ⌘E / ⌘⇧E**
- **"Go to next / previous editor tab"**
  - **Windows/Linux: Alt + Right / Left**
  - **Mac OS X: ^← / ^→**
    - **Clashes with Mac OS X Default Keymap for Mission Control**

Bookmarks

- **Quick navigation between locations in project**
- **Set with**
  - **Windows/Linux: Ctrl + Shift + #[0-9]**
  - **Mac OS X: ^⇧0 ... ^⇧9**
- **Navigate with**
  - **Windows/Linux: Ctrl + #[0-9]**
  - **Mac OS X: ^0 ... ^9**

Search and Replace

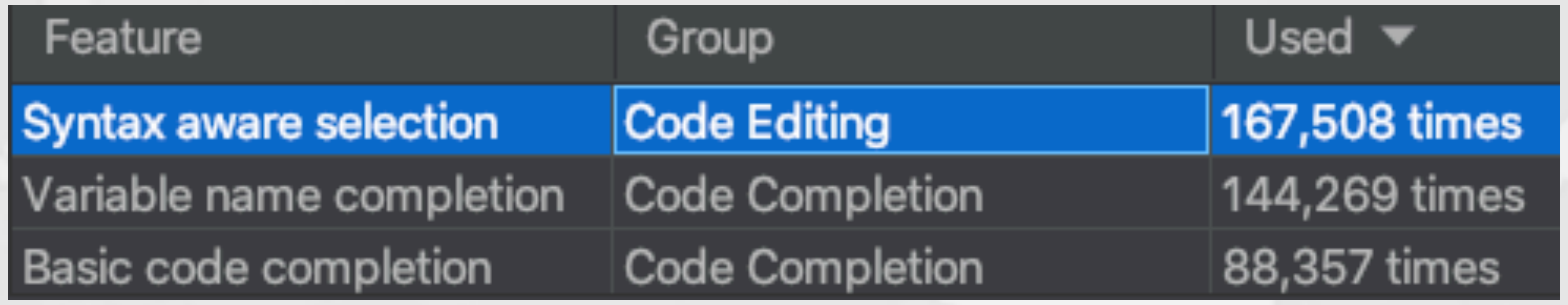
- "Find"
  - Allows regular expressions with .*
  - Windows/Linux: Ctrl + F
  - Mac OS X: ⌘F
- "Find next"
  - Windows/Linux/Mac OS X (Heinz): F3
  - Mac OS X (Official): ⌘G
- "Replace"
  - Windows/Linux: Ctrl + R
  - Mac OS X: ⌘R

Jump out of line onto new line

- "Start New Line"
  - Windows/Linux: Shift + Enter
  - Mac OS X: ⇧↩
- Even in the middle of some code, jump onto new line

Syntax Aware Selection

- "Extend Selection"
  - Windows/Linux: Ctrl + W
  - Mac OS X (Official): ⌥↑
  - Mac OS X (Heinz): ⌘W
    - Closes windows in other Mac OS X programs
    - But my left thumb and middle finger and pinkie do this nicely
    - By FAR my most used shortcut, 167k times since 2014



Feature	Group	Used ▾
Syntax aware selection	Code Editing	167,508 times
Variable name completion	Code Completion	144,269 times
Basic code completion	Code Completion	88,357 times

- "Shrink Selection"
  - Windows/Linux: Ctrl + Shift + W
  - Mac OS X (Official): ⌥↓
  - Mac OS X (Heinz): ⌘⇧W

Move code up / down

- "Move Code Up"
  - Windows/Linux: Ctrl + Shift + Up
  - Mac OS X: ⌘⇧↑
- "Move Code Down"
  - Windows/Linux: Ctrl + Shift + Down
  - Mac OS X: ⌘⇧↓
- Note: If nothing is selected, then we consider the current line to be selected

5. Turbo-boosted Productivity

Surround with ...

- "Surround with ..."
  - Windows/Linux: Ctrl + Alt + T
  - Mac OS X: ⌘⌥T
- Context aware, for example with Java

1. if	6. try / catch
2. if / else	7. try / finally
3. while	8. try / catch / finally
4. do / while	9. synchronized
5. for	0. Runnable

Surround with Live Template

- **"Surround with Live Template"**
  - Windows/Linux: Ctrl + Alt + J
  - Mac OS X: ⌘⌥J
- **Again context aware, for example with Java**

  C. Surround with Callable
  RL. Surround with ReadWriteLock.readLock
  WL. Surround with ReadWriteLock.writeLock
  I. Iterate Iterable or array

Define your own Live Templates

- **Preferences -> Editor -> Live Templates**
- **e.g. Wrap code in System.nanoTime()**
  - **Fantastic for demos, use JMH for serious benchmarks**

```
Abbreviation: snt
Description: System.nanoTime()
Template text:

long $TIME$ = System.nanoTime();
try {
  $SELECTION$
} finally {
  $TIME$ = System.nanoTime() - $TIME$;
  System.out.printf("$TIME$ = %dms%n",
    ($TIME$/1_000_000));
}
```

Error Based Coding

- **Make deliberate mistakes, followed by**
  - **Next highlighted error (F2)**
    - **Previous highlighted error with Shift + F2 or ⇧F2**
  - **Superkey (Alt + Enter or ⌥↩)**
  - **Tab**
- **Quick demo**

Copy & Paste

- An April Fools joke that became real
  - Sorry, you can't have one - they are sold out!

Line Based Editing

- **By default, actions apply to our current line**
  - Windows/Linux:
    - Ctrl + D - Duplicate current line
    - Ctrl + C, Ctrl + Insert - Copy current line to clipboard
    - Ctrl + X, Shift + Delete - Cut current line to clipboard
    - Ctrl + V, Shift + Insert - Paste from clipboard
    - Ctrl + Shift + V - Paste from recent buffers...
    - Ctrl + Y - Delete line
  - Mac OS X:
    - ⌘D - Duplicate
    - ⌘C - Copy
    - ⌘X - Cut
    - ⌘V - Paste
    - ⌘⇧V - Paste from recent buffers
    - ⌘⌫ - Delete line

Column Select Editing

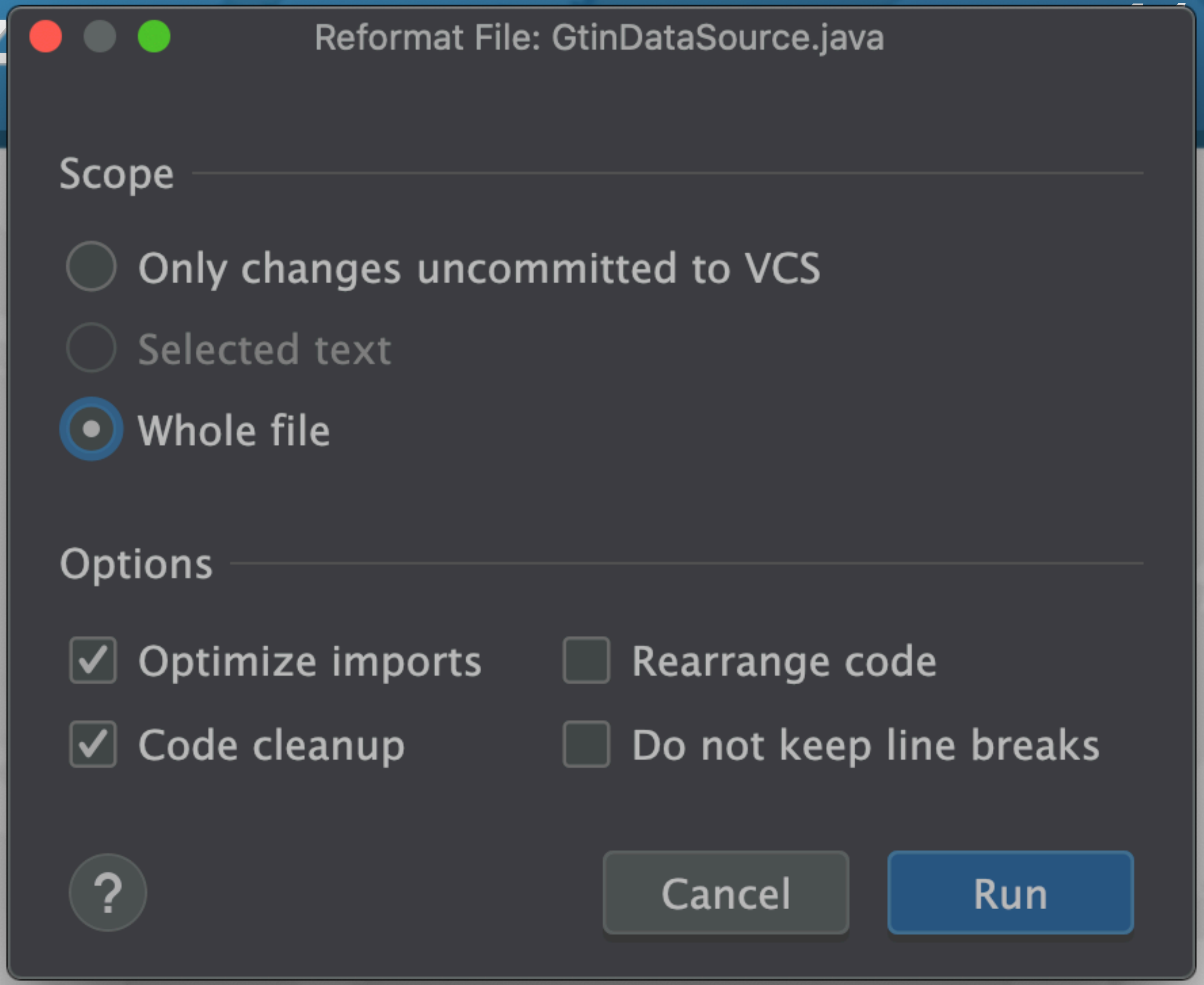
- Let's make all fields in CdsGenerator final
  - One at a time is tedious
- With the mouse
  - Windows/Linux: Alt + Drag Mouse
  - Mac OS X: ⌥+drag mouse
- With keyboard toggle column selection mode
  - Windows/Linux: Alt + Shift + Insert
  - Mac OS X (Official): ⌘⇧8
  - Mac OS X (Heinz): ⌘^⇧C
- Make sure to turn column selection mode off

Toggle Case

- Toggles upper/lower case
  - hello → HELLO → hello
  - Hello → hello
  - helloWorld → helloworld → HELLOWORLD
- Shortcut
  - Windows/Linux: Ctrl + Shift + U
  - Mac OS X: ⌘⇧U

Reformatting Code

- **"Reformat Code"**
  - **Windows/Linux: Ctrl + Alt + L**
  - **Mac OS X: ⌘⌥L**
- **"Reformat File"**
  - **Can be configured for "Optimize Imports" and "Code Cleanup"**
  - **These settings then are also used for "Reformat Code"**
  - **Windows/Linux: Ctrl + Alt + Shift + L**
  - **Mac OS X: ⌘⌥⇧L**


Reformat File: GtinDataSource.java

Scope

( ) Only changes uncommitted to VCS
( ) Selected text
[✓ Whole file]

Options

[✓ Optimize imports]
Rearrange code
[✓ Code cleanup]
Do not keep line breaks

Exercise

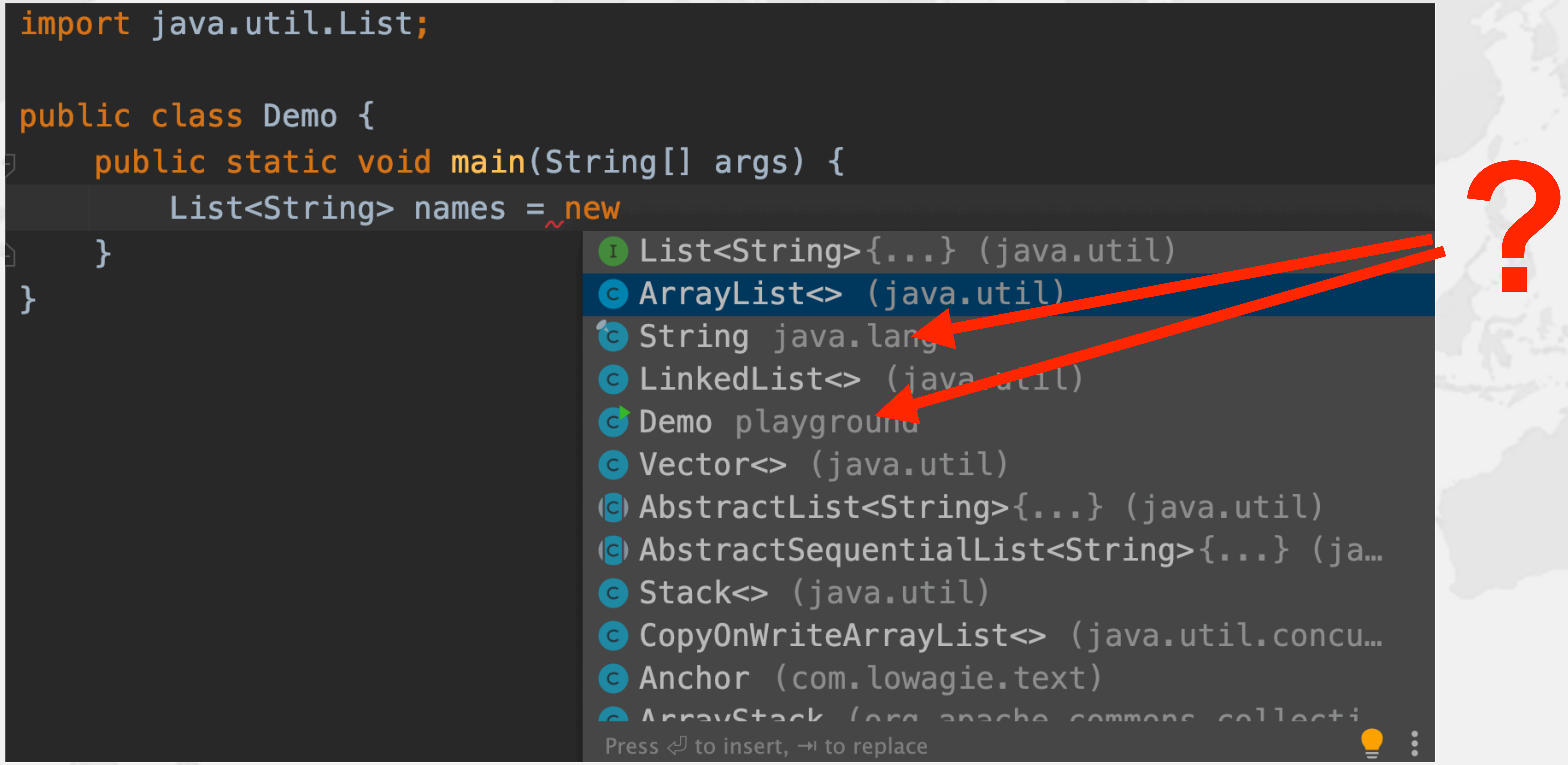
- **Inside the com.someone.ppt.models.FruitSpec, create an enum that holds all the fields**

```
public enum Field {
    BARCODE("BarCode"),
    RUNNUMBER("RunNumber"),
    PUC("PUC"),
    /* etc. ... */
    GTIN("GTIN");
    private final String name;

    Field(String name) {
        this.name = name;
    }
}
```

6. Code Completion

Basic Code Completion

- Code completion traditionally uses Ctrl+Space
  - Windows/Linux: Ctrl + Space
  - Mac OS X: ^Space

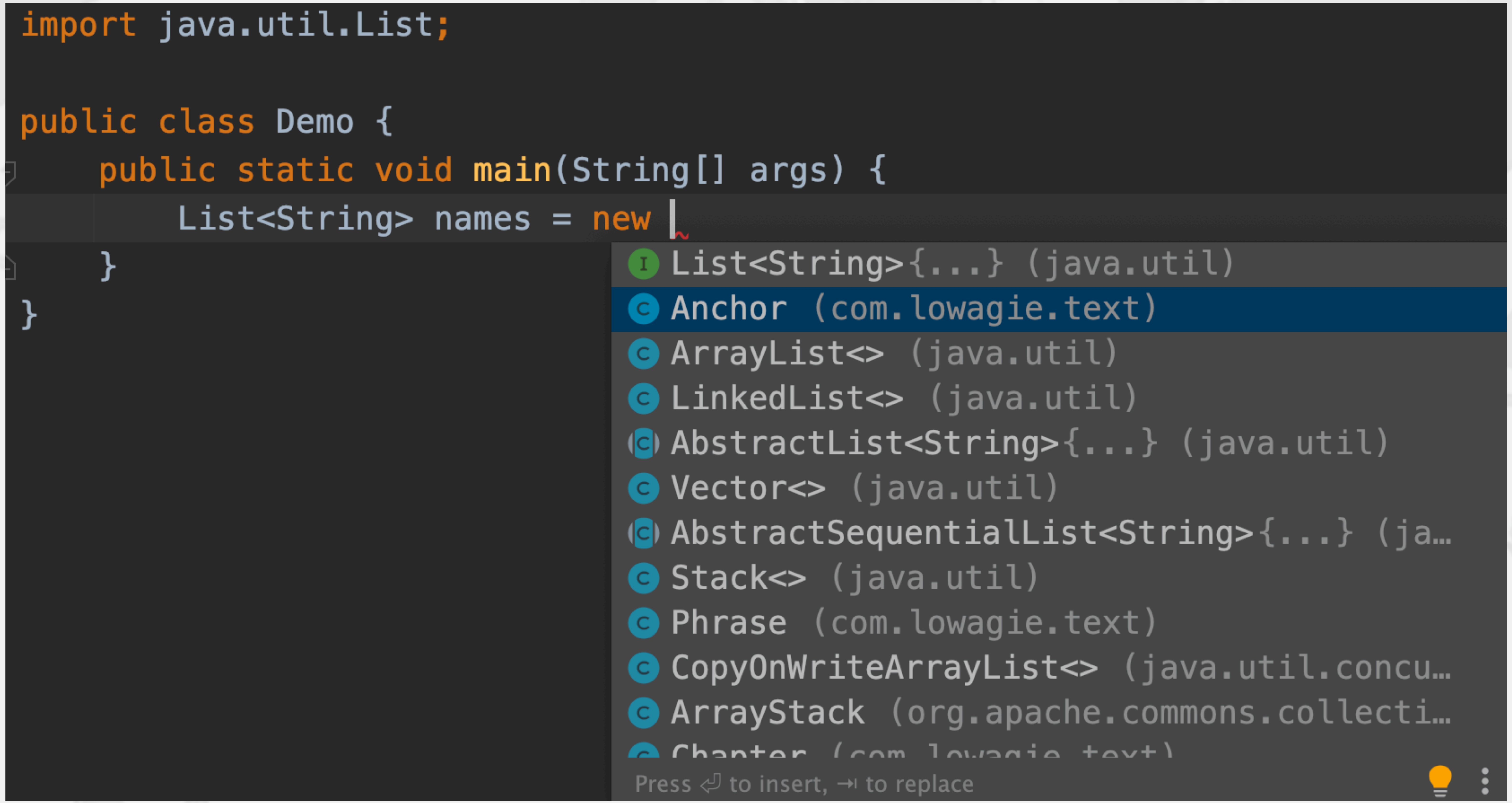


```
import java.util.List;

public class Demo {
    public static void main(String[] args) {
        List<String> names = new
    }
}
```

```
List<String>{...} (java.util)
ArrayList<> (java.util)
String java.lang
LinkedList<> (java.util)
Demo playground
Vector<> (java.util)
AbstractList<String>{...} (java.util)
AbstractSequentialList<String>{...} (ja…
Stack<> (java.util)
CopyOnWriteArrayList<> (java.util.concu…
Anchor (com.lowagie.text)
ArrayStack (org.apache.commons.collecti
Press ⏎ to insert, ⇥ to replace
```

Smart Code Completion

- This gives much better result - I always use this
  - Windows/Linux: Ctrl + Shift + Space
  - Mac OS X: ^⇧Space



```
import java.util.List;

public class Demo {
    public static void main(String[] args) {
        List<String> names = new 
    }
}
```

```
List<String>{...} (java.util)
Anchor (com.lowagie.text)
ArrayList<> (java.util)
LinkedList<> (java.util)
AbstractList<String>{...} (java.util)
Vector<> (java.util)
AbstractSequentialList<String>{...} (ja…
Stack<> (java.util)
Phrase (com.lowagie.text)
CopyOnWriteArrayList<> (java.util.concu…
ArrayStack (org.apache.commons.collecti…
Chapter (com.lowagie.text)
Press ⏎ to insert, ⇥ to replace
```

Code Folding

- Hide non-essential information, such as
  - File headers, imports, Javadocs
  - Method bodies
    - One-line methods, getters/setters, anonymous types
  - Sections of a method
  - etc.
  - Settings → Editor → General → Code Folding
- "Expand / collapse code block"
  - Windows/Linux: Ctrl + NumPad+ / NumPad-
  - Mac OS X: ⌘+ / ⌘-

Override and Implement Methods

- "Override methods"
  - Windows/Linux: Ctrl + O
  - Mac OS X: ⌘O
- "Implement methods"
  - Windows/Linux: Ctrl + I
  - Mac OS X: ⌘I
- We can also do that with "Generate Code"
  - Windows/Linux: Alt + Ins
  - Mac OS X (Official): ⌘N
  - Mac OS X (Heinz): ^↩ or ^N

CamelCase in Code Completion

- **Instead of C**opy**O**n**W**rite**A**rray**L**ist**, use COWAL**
  - **IntelliJ starts searching as we type**
  - **Simply use the capital letters in the CamelCase class**

8. Refactoring

Refactoring

- Pioneered by Martin Fowler
  - Based on research by William Opdyke
- What it is
  - Improving the design of existing code
    - Without adding new functionality
    - Introduce good design patterns
- Unit testing
  - Bad refactorings often introduce bugs

Copy Class

- "Copy"
  - Windows/Linux: F5
  - Mac OS X: F5
  - Turn on function keys for IntelliJ IDEA on MacBook Pro
    - Settings → Keymap → Show F1, F2 etc. keys on the Touch Bar
- Used more often than I thought
  - On average, once a day
  - But that includes a lot of teaching, rather than just new code
- Ok, let's move on ....

Move Class / Field / Method

- "Move"
  - Windows/Linux: F6
  - Mac OS X: F6
- Exercise
  - Move PrtGenerator.calculateFakeGTIN() and calculateFakeGTIN2() to a new class com.someone.ppt.cds.GTINGenerator

Renaming Things

- "Rename ..."
  - Windows/Linux: Shift + F6
  - Mac OS X: ⇧F6
- Applies to almost anything:
  - Classes, methods, parameters, variables, fields
- Another trick:
  - Rename it and then propagate all changes with our superkey
    - Windows/Linux: Alt + Enter
    - Mac OS X: ⌥↩
    - Do it before leaving that particular line
  - Exercise: Rename Tickable to Tickeable

Change Signature

- "Change Signature"
  - Windows/Linux: Ctrl + F6
  - Mac OS X: ⌘F6
- Applies mainly to methods for changing:
  - Return type
  - Parameter types and names
  - Order of parameters
- Can in some cases use the same trick as previous
  - Change method and then propagate with Alt + Enter or ⌥↩

Exercise

- **Remove parameters from the buildQuery() method in TableModelFactory whose values are always null**
  - **Also change "StringBuffer result" to use StringBuilder**

Extract Variable

- Local variables tend to increase method length
- "Extract Variable"
  - Windows/Linux: Ctrl + Alt + V
  - Mac OS X: ⌘⌥V
- Break up long chain of method calls into variables
  - e.g. Java 8 streams
- Can also help to reduce duplicate code
- Exercise
  - generatePcdTemplate() in PcdGenerator
    - "" + eventID

Extract Method

- Select a block of code and "Extract Method"
  - Windows/Linux: Ctrl + Alt + M
  - Mac OS X: ⌘⌥M
- Some restrictions
  - Cannot have more than one return value
  - Block must represent a set of statements or expressions
- Additional benefits
  - Extracting a method can discover other, similar, code

Exercise

- **Extract snippets from generatePcdRemarks() method in PcdGenerator into separate method**

```
String remark = resultSet.getString("Remark1");
if (!"".equals(remark)) {
    remarks.put(remark, remark);
}

remark = resultSet.getString("Remark2");
if (!"".equals(remark)) {
    remarks.put(remark, remark);
}

remark = resultSet.getString("Remark3");
if (!"".equals(remark)) {
    remarks.put(remark, remark);
}
```

Inline Code

- Applies to methods, fields, local variables
- "Inline"
  - Windows/Linux: Ctrl + Alt + N
  - Mac OS X: ⌘⌥N
- Conveniently close to "Extract Method" shortcut

Extract Field

- "Extract Field" applies to expressions and variables
  - Windows/Linux: Ctrl + Alt + F
  - Mac OS X: ⌘⌥F
- Not that useful
  - Extracted fields would typically not be final, set in methods
  - May introduce race conditions

Extract Constant

- "Extract Constant" uses Java naming convention
  - Windows/Linux: Ctrl + Alt + C
  - Mac OS X: ⌘⌥C

Extract Parameter

- "Extract Parameter"
  - Windows/Linux: Ctrl + Alt + P
  - Mac OS X: ⌘⌥P

Safe Delete

- "Safe Delete" - searches whether the code is used
  - Can also search in comments and Strings
  - – Windows/Linux: Alt + Delete
  - – Mac OS X: ⌘⌦
    - ⌦ is fn⌫ on laptop keyboard
- We can also search for unused code with analyzer
  - – More in next section

Postfix Completion

- **Write a postfix after your expression and press tab**
  - **! - Negates a boolean expression**
  - **nn - Adds a check verifying that an expression is not null**
  - **null - Adds a check verifying that an expression is null**
  - **try - Inserts a statement in a try-catch block**
  - **var - Introduces a local variable for an expression**
- **Demo**

11. Conclusion

Conclusion to IntelliJ Wizardry

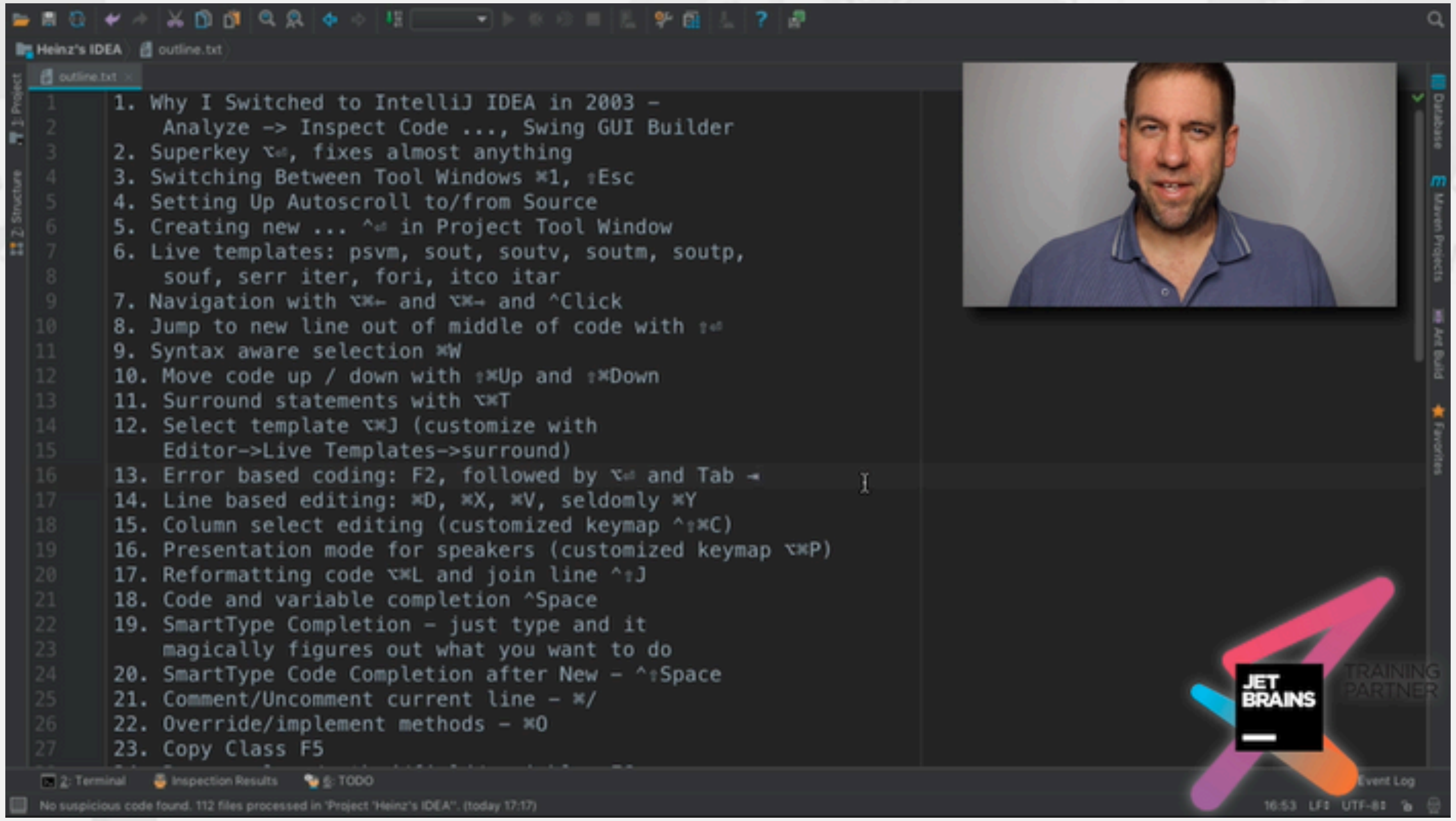
- **Many more keystrokes and features to learn**
- **One new one per day**
- **Happy coding!**

The Java Specialists' Newsletter

- **Make sure to subscribe**
  - **www.javaspecialists.eu/archive/subscribe/**
- **Readers in 150+ countries**
- **Over 21 years of newsletters on advanced Java**
  - **All previous newsletters available on www.javaspecialists.eu**
  - **Courses, additional training, etc.**

IntelliJ Wizardry with Younger Heinz

- **Get it here for $7 plus taxes**
  - **https://javaspecialists.teachable.com/p/intellij-wizardry**
  - **Or wait for the recording of this course (free)**



```
1. Why I Switched to IntelliJ IDEA in 2003 -
    Analyze -> Inspect Code ..., Swing GUI Builder
2. Superkey ⌥⏎, fixes almost anything
3. Switching Between Tool Windows ⌘1, ⇧Esc
4. Setting Up Autoscroll to/from Source
5. Creating new ... ^⏎ in Project Tool Window
6. Live templates: psvm, sout, soutv, soutm, soutp,
    souf, serr iter, fori, itco itar
7. Navigation with ⌥⌘← and ⌥⌘→ and ^Click
8. Jump to new line out of middle of code with ⇧⏎
9. Syntax aware selection ⌘W
10. Move code up / down with ⇧⌘Up and ⇧⌘Down
11. Surround statements with ⌥⌘T
12. Select template ⌥⌘J (customize with
    Editor->Live Templates->surround)
13. Error based coding: F2, followed by ⌥⏎ and Tab ⇥
14. Line based editing: ⌘D, ⌘X, ⌘V, seldomly ⌘Y
15. Column select editing (customized keymap ^⇧⌘C)
16. Presentation mode for speakers (customized keymap ⌥⌘P)
17. Reformatting code ⌥⌘L and join line ^⇧J
18. Code and variable completion ^Space
19. SmartType Completion - just type and it
    magically figures out what you want to do
20. SmartType Code Completion after New - ^⇧Space
21. Comment/Uncomment current line - ⌘/
22. Override/implement methods - ⌘O
23. Copy Class F5
```

No suspicious code found. 112 files processed in 'Project 'Heinz's IDEA''. (today 17:17)