

IntelliJ Super Productivity in 45 Minutes

Dr Heinz M. Kabutz

Last Updated 2022-05-03



Why IntelliJ IDEA?

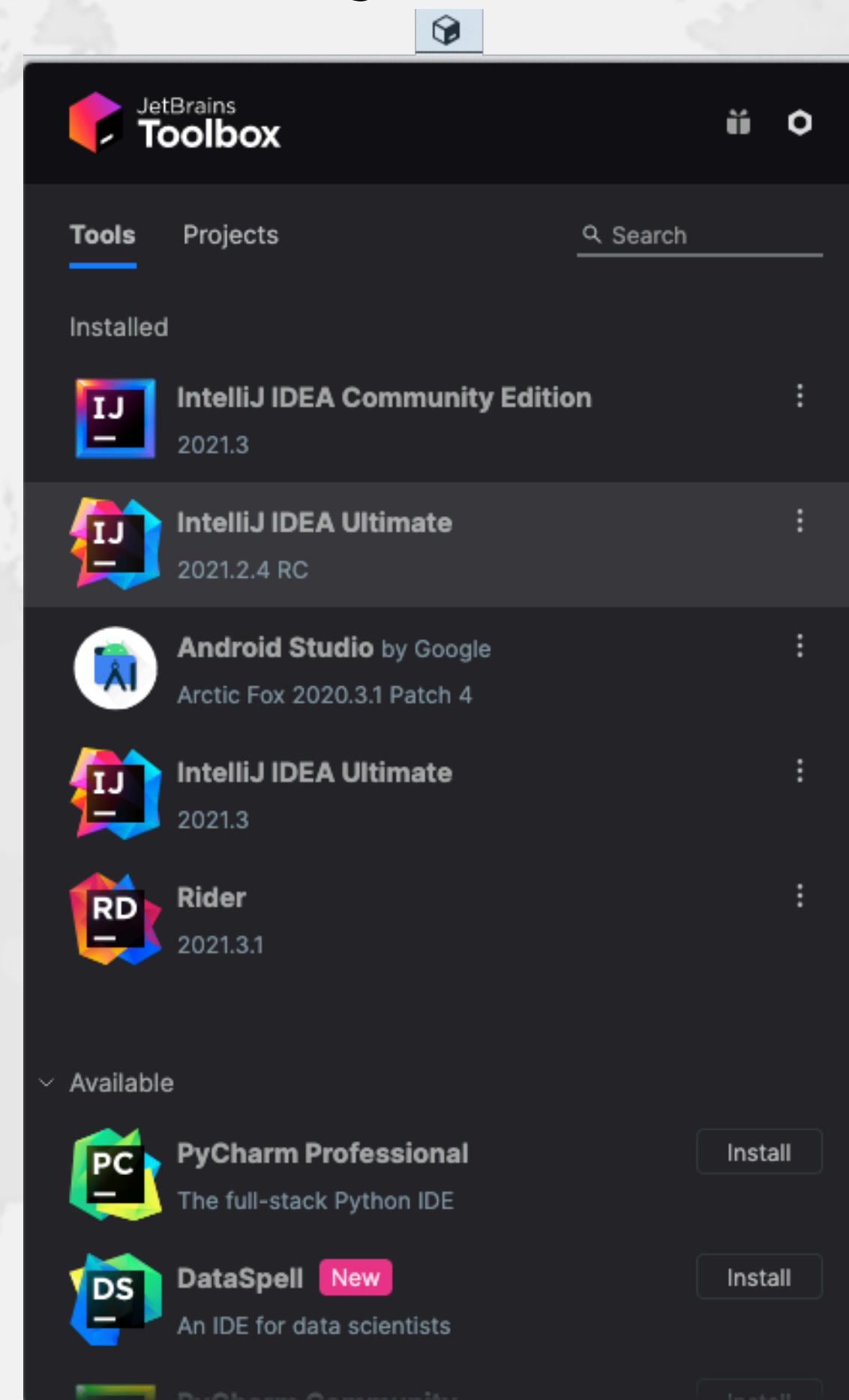
- **My story**

- **Started with Borland JBuilder, then Eclipse for a year**
- **Needed to work on some messy Java code**
 - **Heard about IntelliJ IDEA's amazing code analysis**
- **Downloaded evaluation copy**
 - **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**

Download JetBrains Toolbox

- **Keeps the IDEs up to date**

– <https://www.jetbrains.com/toolbox-app/>



IntelliJ Settings

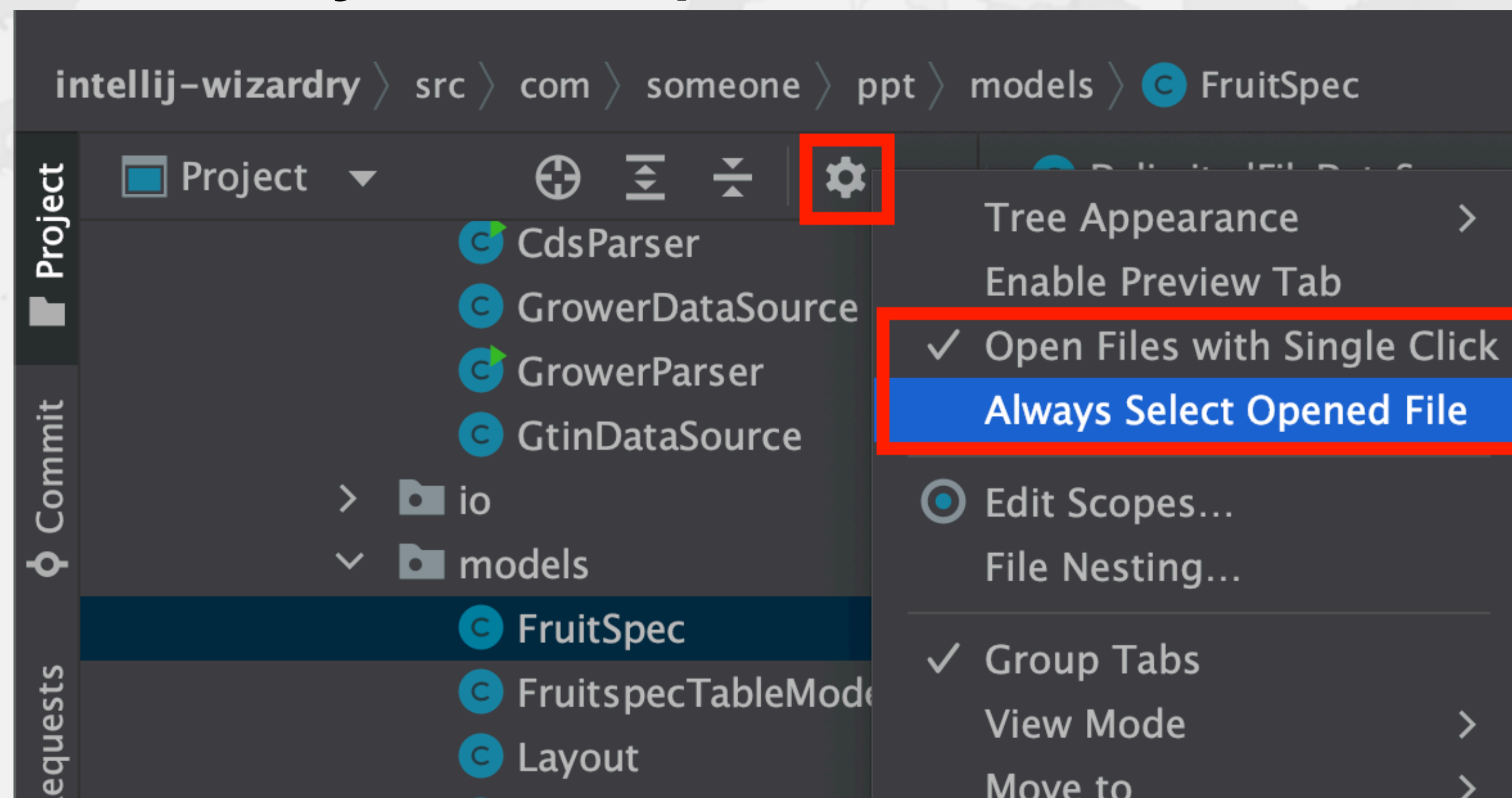
- **Editor →**

- **General → Appearance: Show method separators**
- **General → Smart Keys: Select Use "CamelHumps" words**
- **Color Scheme: Darcula but with more prominent errors**

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**



IntelliJ IDEA Philosophy

- IntelliJ designed to be used mostly without mouse
- Hotkeys for almost everything
 - Help → Keyboard Shortcuts 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
Open 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
Resume program	Alt + F8
Toggle breakpoint	F9
New breakpoints	Ctrl + FB
	Ctrl + Shift + FB

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
Deselect 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 Docs	Shift + F1
Ctrl + Ins	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
Autoscroll lines	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 all code block end / start	Ctrl + Shift + J / [
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

REFACTORIZING

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 method	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 line 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 + J / [
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
Alt source / view source	F4 / Shift + Enter
Show navigation bar	Alt + Home
Toggle bookmarks	F11
Toggle bookmarks with mnemonic	Ctrl + F11
Go to numbered bookmark	Ctrl + #[0-9]
Show bookmarks	Shift + F11

COMPILER 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

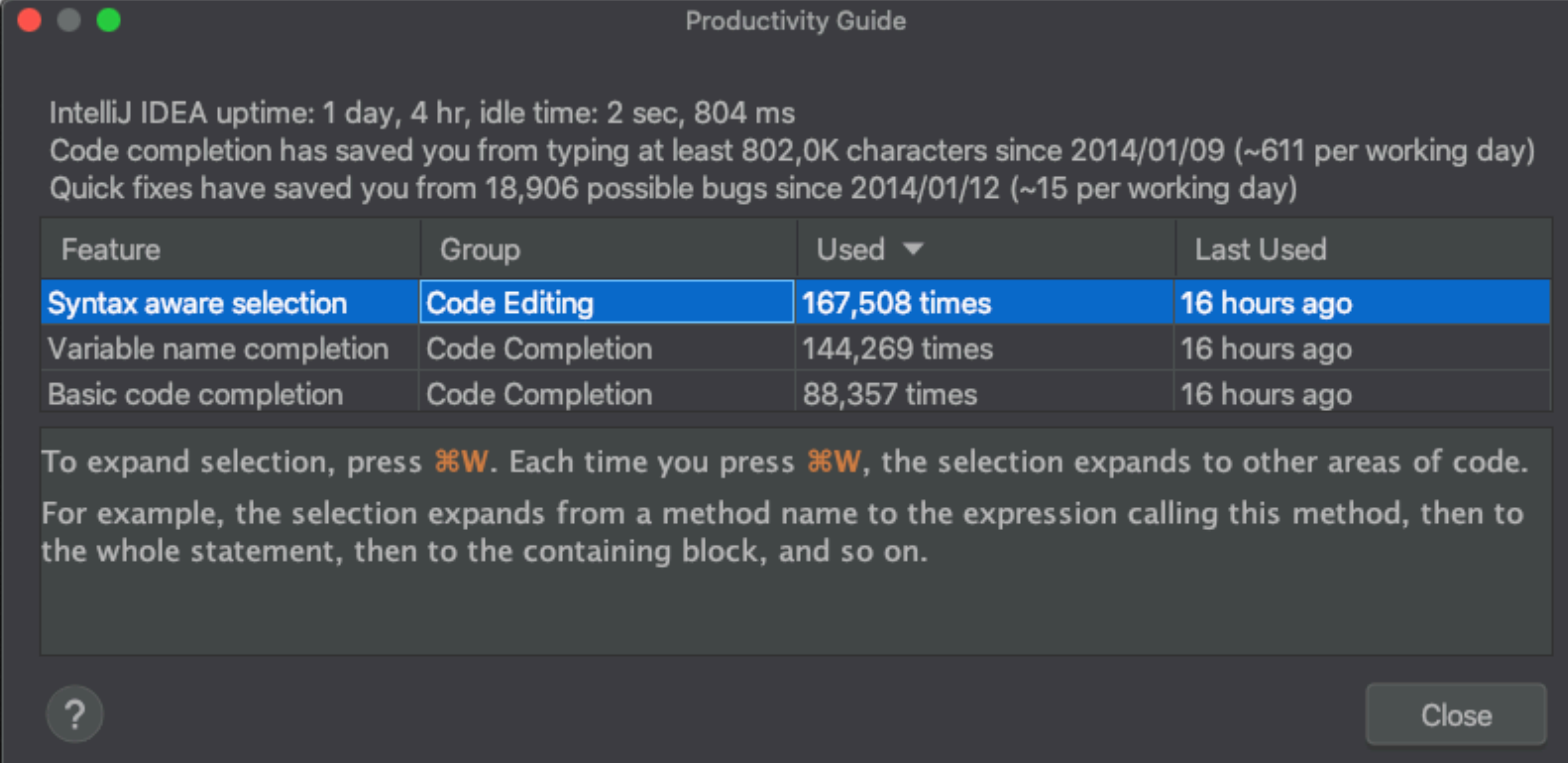
Commit project to VCS	Ctrl + K
Update project from VCS	Ctrl + T
Push commit	Ctrl + Shift + K
VCS quick popup	Alt + BackQuote (`)

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



The screenshot shows a 'Productivity Guide' dialog box with a title bar containing three window control buttons (red, grey, green) and the text 'Productivity Guide'. The main content area contains the following text:

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.

At the bottom left, there is a question mark icon in a circle. At the bottom right, there is a 'Close' button.

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 ⇧**



Superkey for fixing almost anything

- **"Show intention actions and quick-fixes"**

- Windows/Linux: Alt + Enter

- Mac OS X: $\text{⌘} \leftarrow$

- **Quick demo fixing**

- `com.someone.ppt.cds.CdsGenerator`**

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**

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

Syntax Aware Selection

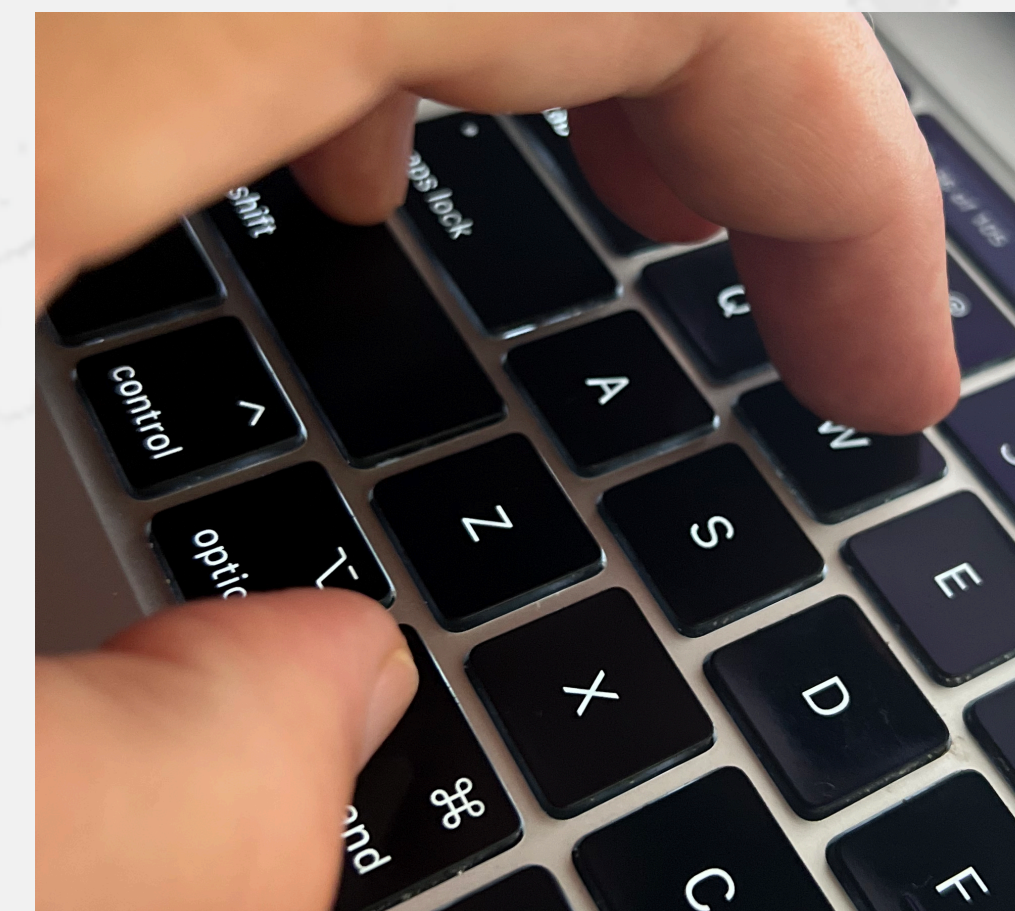
- "Extend Selection"

- 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**



Surround with ...

- "Surround with ..."

- Windows/Linux: Ctrl + Alt + T

- Mac OS X: ⌘⇧T

- Context aware, for example with Java

1. if
2. if / else
3. while
4. do / while
5. for

6. try / catch
7. try / finally
8. try / catch / finally
9. synchronized
0. Runnable

Define your own Templates

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

Abbreviation: `nanotime`

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));  
}
```

Before we continue ...

- **Get our Data Structures in Java Course here**

- <https://tinyurl.com/jax2022>

- **Strongly recommended to tick**



I agree to receive promotional and instructional emails from JavaSpecialists

- **Coupon expires at 13:15 Berlin Time**

- **But lifetime access to course**



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: $\text{⌘} + \text{drag mouse}$
- **With keyboard toggle column selection mode**
 - Windows/Linux: Alt + Shift + Insert
 - Mac OS X (Official): $\text{⌘} \uparrow 8$
 - Mac OS X (Heinz): $\text{⌘} \wedge \uparrow C$
- **Make sure to turn column selection mode off**

tinyurl.com/jax2022



Joining and Splitting Lines

- "Smart line join"

- Windows/Linux: **Ctrl + Shift + J**

- Mac OS X: **⌘⇧J**

- e.g.

```
String[][] data = null;  
data = source.getData();
```



Smart line join

```
String[][] data = source.getData();
```

- "Smart line split"

- Not sure what is "smart" about it - keeps cursor on current line

- Windows/Linux: **Ctrl + Enter**

- Mac OS X: **⌘↵**

tinyurl.com/jax2022



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  
    }  
}
```

```
1 List<String>{...} (java.util)  
2 ArrayList<> (java.util)  
3 String java.lang  
4 LinkedList<> (java.util)  
5 Demo playground  
6 Vector<> (java.util)  
7 AbstractList<String>{...} (java.util)  
8 AbstractSequentialList<String>{...} (ja...  
9 Stack<> (java.util)  
10 CopyOnWriteArrayList<> (java.util.concu...  
11 Anchor (com.lowagie.text)  
12 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 |
```

```
    }
```

```
}
```

```
1 List<String>{...} (java.util)
```

```
2 Anchor (com.lowagie.text)
```

```
3 ArrayList<> (java.util)
```

```
4 LinkedList<> (java.util)
```

```
5 AbstractList<String>{...} (java.util)
```

```
6 Vector<> (java.util)
```

```
7 AbstractSequentialList<String>{...} (ja...
```

```
8 Stack<> (java.util)
```

```
9 Phrase (com.lowagie.text)
```

```
10 CopyOnWriteArrayList<> (java.util.concu...
```

```
11 ArrayStack (org.apache.commons.collecti...
```

```
12 Chapter (com.lowagie.text)
```

Press ↵ to insert, → to replace



Parameter Info

- Should appear automatically when you type "("
- Shows the parameters for a method
 - Windows/Linux: Ctrl + P
 - Mac OS X: ⌘P

@NotNull String url, Properties info

@NotNull String url, @Nullable String user, @Nullable String password

@NotNull String url

DriverManager.getConnection()

CamelCase in Code Completion

- **Instead of CopyOnWriteArrayList, use COWAL**
 - IntelliJ starts searching as we type
 - Simply use the capital letters in the CamelCase class

Imports

- **Imports are managed mostly automatically**
 - **Can be configured in settings for single imports or multiple**
 - **Depends on your company standards**
 - **java.base has**
 - **13679 explicit imports**
 - **876 wildcard imports, mostly java.util, java.security, java.io (6%)**
 - **585 static explicit imports**
 - **180 static wildcard imports (24%)**
 - **I usually fold away the imports and never look at them**

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

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

Demo

- **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**

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**

Analyzer

- **IntelliJ shines with its code analyzer and refactoring**
- **Code → Inspect Code ...**
 - **Whole project**
 - **Or: Right-click in project → Analyze → Inspect Code ...**
 - **Inspection profile: Default IDE**
- **It checks for the most glaring code inconsistencies**

Java Language Migration Aids

- **Exercises**

- Use enhanced switch in

- `CdsGenerator#generateCfg()`

- `PcdGenerator#generatePcdTemplate()`

- Use pattern variable in `ElegantTable#initGui()`

- Use try-with-resource in `LineSocket#receiveFile()`

Run Configuration

- **Templates used for default run configuration**
 - **Run → Edit configurations... → Edit configuration templates...**
 - **Select Application → Modify options → Show the run/debug settings before start**
 - **This always opens up the configuration window before run**
 - **To set a VM Option, use Modify options → Add VM options**

Demo

Edit Configuration

Name: Store as project file

Build and run Modify options ⌘M

Press for field hints

Working directory:

Environment variables:

Separate variables with semicolon: VAR=value; VAR1=value1

Open run/debug tool window when started

Show the run/debug configuration settings before start

Allow multiple instances

Don't forget ...

- **Get our Data Structures in Java Course here**

- <https://tinyurl.com/jax2022>

- **Strongly recommended to tick**



I agree to receive promotional and instructional emails from JavaSpecialists

- **Coupon expires at 13:15 Berlin Time**

- **But lifetime access to course**



Conclusion

- **Many more keystrokes and features to learn**
- **One new one per day**
- **Happy coding!**
- **Twitter: @heinzkabutz**
- **Newsletter: www.javaspecialists.eu**