The University of Nottingham

UNITED KINGDOM · CHINA · MALAYSIA

Lecture 02B

Horia A. Maior and Marjahan Begum

Lab Reflection + IntelliJ + Java Releases + jShell + Basic ZooApp Maintenance

COMP2013: Developing Maintainable Software
Week 3 – 1:00pm Thursday – 12 October 2023

valid for 65 minutes from 12:55pm
generated 2023-10-10 03:14

valid for 65 minutes from 12:55pm
generated 2023-10-10 03:14

COMP2013

Developing Maintainable Software

Lecture 02B

Lab Reflection + IntelliJ + Java Releases + jShell + Basic ZooApp Maintenance

Horia A. Maior and Marjahan Begum

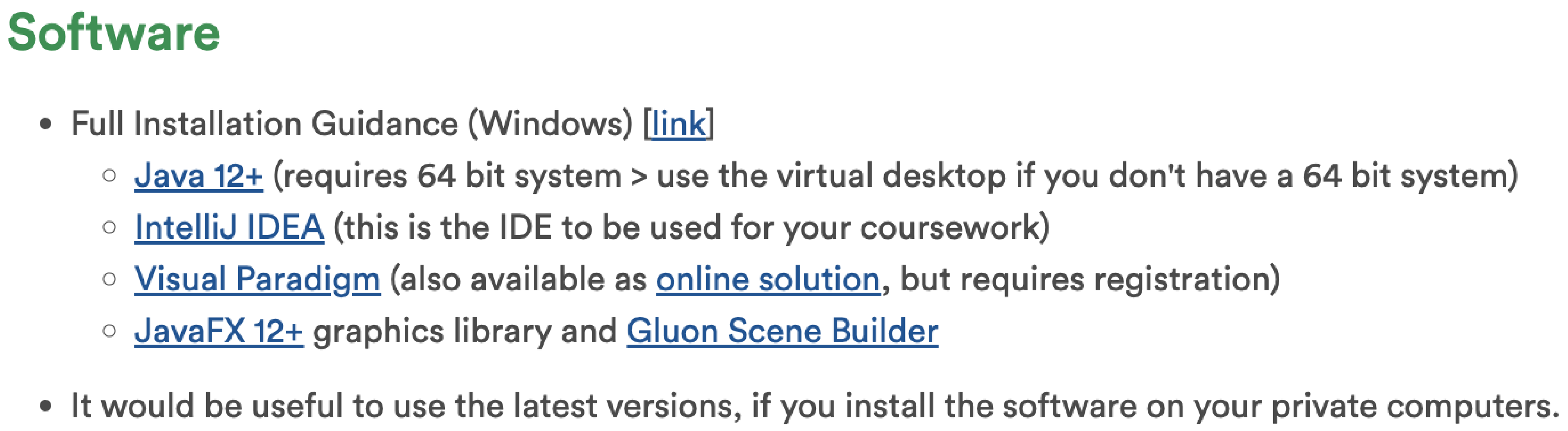
University of Nottingham
UK | CHINA | MALAYSIA

Topics for this Week

valid for 65 minutes from 12:55pm
generated 2023-10-10 03:14

- Lecture 02A:
  - OO and Java Refresher (2/2)
    - The missing bits from last lecture
    - Java Collections framework
    - Implementation of object oriented concepts in Java
- Lab 02:
  - Implementing the ZooApp
- Lecture 02B:
  - Lab reflection
  - IDEs + Java Releases
  - jShell
  - Maintaining the ZooApp (basic maintenance)

Software that we recommended you to install



Software

- Full Installation Guidance (Windows) [link]
  - Java 12+ (requires 64 bit system > use the virtual desktop if you don't have a 64 bit system)
  - IntelliJ IDEA (this is the IDE to be used for your coursework)
  - Visual Paradigm (also available as online solution, but requires registration)
  - JavaFX 12+ graphics library and Gluon Scene Builder
- It would be useful to use the latest versions, if you install the software on your private computers.

valid for 65 minutes from 12:55pm
generated 2023-10-10 03:14

</>

Labsheet 2

COMP2013: Developing Maintainable Software
Week 3 – 1:00pm Thursday – 12 October 2023

valid for 65 minutes from 12:55pm
generated 2023-10-10 03:14

LAB 2: BUILDING THE ZOO EXAMPLE

Aims:

- Implement some of the object-oriented examples we saw in lecture 2A (to remind yourself, go back to lecture 2A slides)
- Gain more experience of object oriented programming
  - Remember that familiarity with this will not only help you write better code, but will aid understanding of existing projects

Coverage:

- In Part 1 (first half of this worksheet) we cover:
  - setters, getters, static variables, ArrayLists, and IDE shortcuts
- In Part 2 (second half of this worksheet) we cover:
  - Aggregation vs Composition
  - Writing the Employee classes for the Zoo example, including an abstract class and an interface

We release the labsheets on Mondays, for you to start working on this right away. This work goes beyond the lab session time, so we encourage you to finish this work, and use the lab time for support.

Questions channels on Teams. Tag one of our teaching support group in the questions channel. Ask generic questions on the peer support channel. We aim to respond as quickly as possible.

intellij

Useful Material to Learn More about IntelliJ

- Check out the "IntelliJ IDEA by JetBrains" channel for more
  - https://www.youtube.com/channel/UC4ogdcPcIAOOMJktgBMhQnQ
- IntelliJ IDEA Full Course 2020 (video 2h 35 min; skip first 23 min)
  - https://www.youtube.com/watch?v=yefmcX57Eyg
- TutorialsPoint: IntelliJ (text-based)
  - https://www.tutorialspoint.com/intellij_idea/index.htm

Code Completion

- Code completion:
  - Start typing and use {TAB} key to use choice offered or {CTRL+.} to use the choice offered followed by a "."
  - Type shortcuts and hit {TAB} key
    - For example: "sout" + {TAB} > "System.out.println();"
    - Hit {CTRL+J} to see all available shortcuts
  - Use clipboard viewer to find and reuse copied content, by pressing {CTRL+SHIFT+V}

Useful Plugins

- Productivity Bundle Plugins: https://plugins.jetbrains.com/bundles/4-productivity-bundle
  - Key Promoter X: https://plugins.jetbrains.com/plugin/9792-key-promoter-x
  - IDE Feature Trainer: https://plugins.jetbrains.com/plugin/8554-ide-features-trainer
- Presentation Assistant (show keys pressed): https://plugins.jetbrains.com/plugin/7345-presentation-assistant (requires MacOS Keymap plugin https://plugins.jetbrains.com/plugin/13258-macos-keymap)

</>

what's new in Java (highlights)

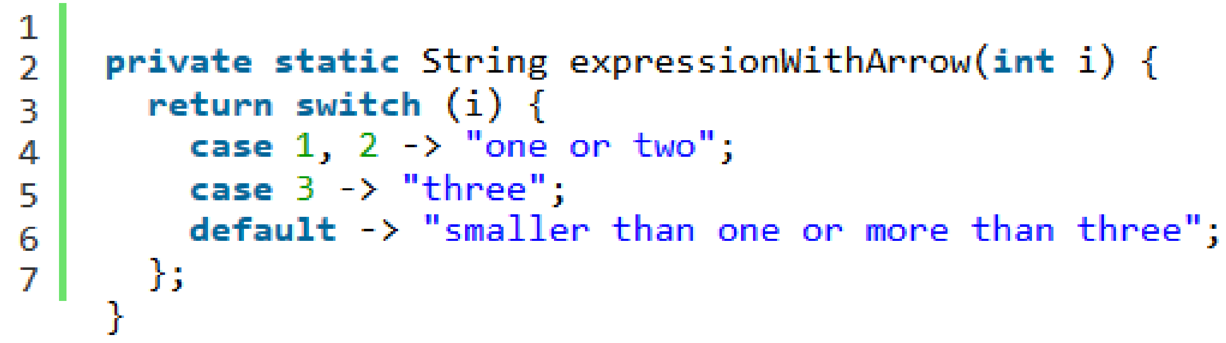
What's new in Java (highlights)

https://www.codejava.net/java-se/java-se-versions-history

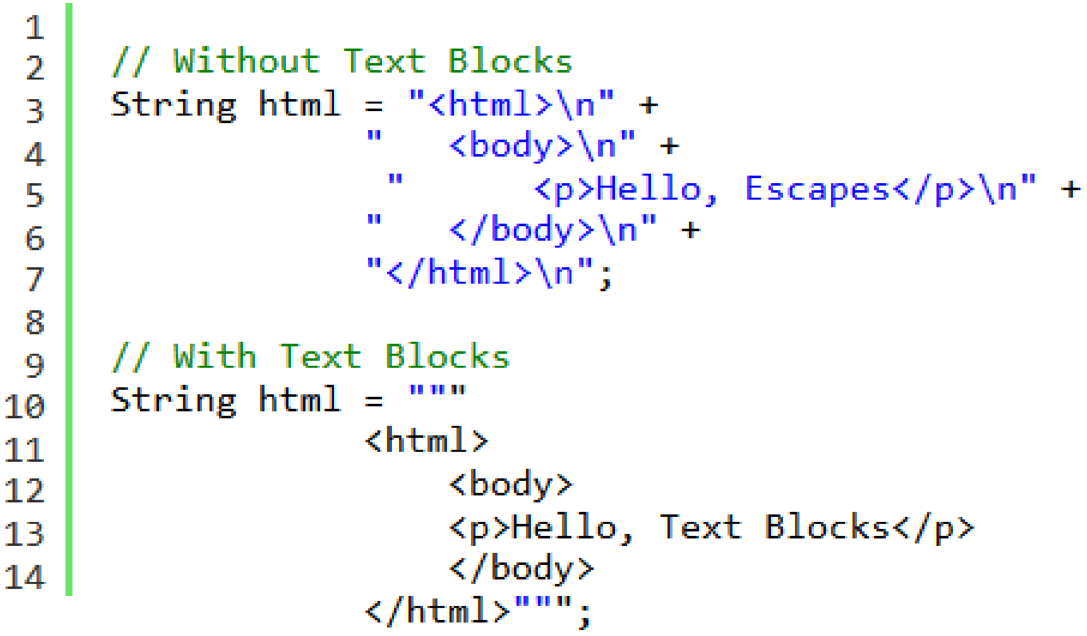
- Java 9
  - Modules
  - jShell (REPL)
- Java 10
  - Local-Variable Type Inference (using var)
  - Enhancements for garbage collection and compilation
- Java 11
  - Removal of JavaFX (now an independent module)

What's new in Java (highlights)

- Java 12
  - New garbage collector
  - New switch expression (preview)
- Java 13
  - Switch expressions (preview)
  - Text Blocks (preview)



```java

private static String expressionWithArrow(int i) {
  return switch (i) {
    case 1, 2 -> "one or two";
    case 3 -> "three";
    default -> "smaller than one or more than three";
  };
}
```



```java

// Without Text Blocks
String html = "<html>\n" +
              "   <body>\n" +
               "      <p>Hello, Escapes</p>\n" +
              "   </body>\n" +
              "</html>\n";

// With Text Blocks
String html = """
              <html>
                  <body>
                  <p>Hello, Text Blocks</p>
                  </body>
              </html>""";
```

What's new in Java (highlights)

</>

- Java 14
  - JDK Flight Recorder event streaming
  - Pattern matching
  - Switch expressions
- Java 15
  - Text blocks, hidden classes, a foreign-memory access API, the Z Garbage Collector
  - Previews of sealed classes, pattern matching, and records

What's new in Java (highlights)

</>

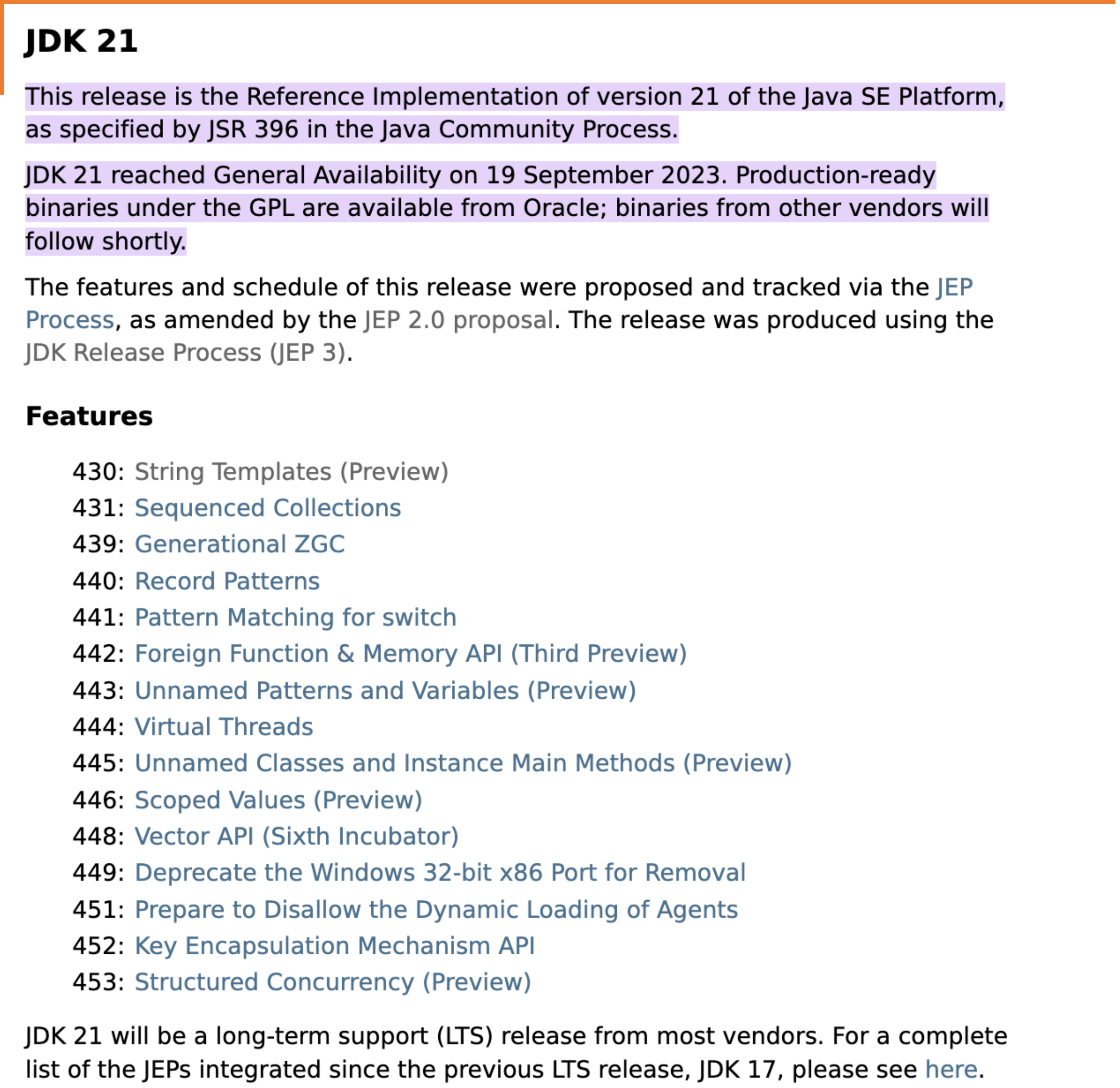
- Java 16
  - Concurrent thread-stack processing for garbage collection (JVM)
  - Support for C++ 14 language features (for the C++ JDK)
  - "Elastic Metaspace" capability to more quickly return unused class metadata memory (JVM)
  - Pattern matching for instanceof
  - Records (classes that act as transparent carriers of immutable data)
- Java 17
  - Standardises sealed classes and interfaces
  - Pattern matching for switch statements (Preview)
  - Deprecates the Security Manager and Applet APIs

What's new in Java (highlights)

- Java 18
  - Simple Web Server
  - Code Snippets in Java API Documentation
  - Vector API (Third Incubator) and Foreign Function & Memory API (Second Incubator)
  - Pattern Matching for switch (Second Preview)
- Java 19
  - Structured Concurrency (Incubator)
  - Pattern Matching for switch (Third Preview)
  - Vector API (Fourth Incubator) and Foreign Function & Memory API (Preview)
  - Virtual Threads (Preview)
  - Record Patterns (Preview)

- Java 20
  - virtual threads,
  - a vector API proposal,
  - structured concurrency,
  - a foreign function and memory API,
  - record patterns,
  - pattern matching for switch statements and expressions

https://openjdk.org/projects/jdk/21



JDK 21

This release is the Reference Implementation of version 21 of the Java SE Platform, as specified by JSR 396 in the Java Community Process.

JDK 21 reached General Availability on 19 September 2023. Production-ready binaries under the GPL are available from Oracle; binaries from other vendors will follow shortly.

The features and schedule of this release were proposed and tracked via the JEP Process, as amended by the JEP 2.0 proposal. The release was produced using the JDK Release Process (JEP 3).

Features

430: String Templates (Preview)
431: Sequenced Collections
439: Generational ZGC
440: Record Patterns
441: Pattern Matching for switch
442: Foreign Function & Memory API (Third Preview)
443: Unnamed Patterns and Variables (Preview)
444: Virtual Threads
445: Unnamed Classes and Instance Main Methods (Preview)
446: Scoped Values (Preview)
448: Vector API (Sixth Incubator)
449: Deprecate the Windows 32-bit x86 Port for Removal
451: Prepare to Disallow the Dynamic Loading of Agents
452: Key Encapsulation Mechanism API
453: Structured Concurrency (Preview)

JDK 21 will be a long-term support (LTS) release from most vendors. For a complete list of the JEPs integrated since the previous LTS release, JDK 17, please see here.

What's new in Java (highlights)

- For full details: Tech Geek Next
  - https://www.techgeeknext.com/java/java11-features

    …

    …
  - https://www.techgeeknext.com/java/java19-features

    …

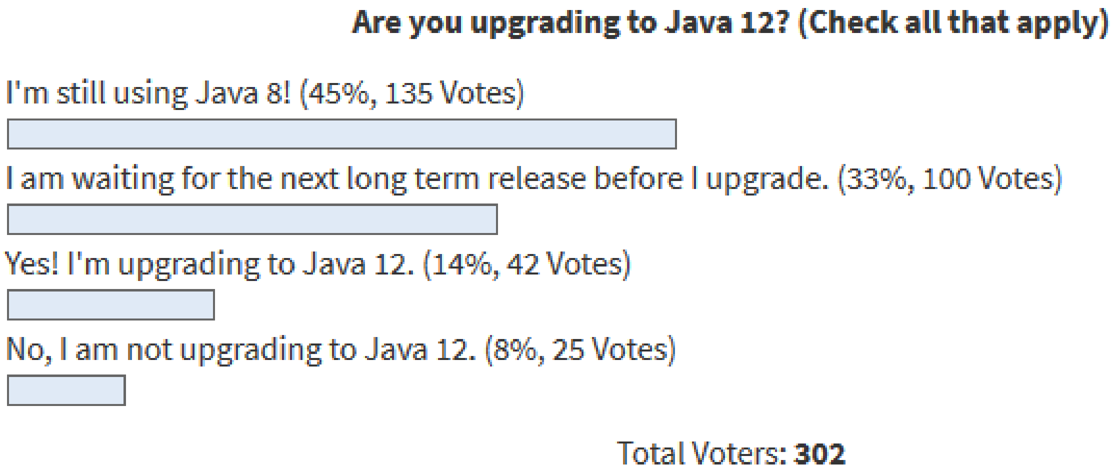
    …
  - https://developers.redhat.com/articles/2023/09/21/whats-new-developers-jdk-21

- For full details: Wikipedia
  - https://en.wikipedia.org/wiki/Java_version_history

Adoption Poll

- Which Version of Java Should You Use?
  - It's important to note that even in 2022, many applications continue to run on Java 8 and Java 11. While Java 17 offers LTS, it's certainly not unusual if your application is using 11 or even 8.

https://www.stackchief.com/blog/Which%20Version%20of%20Java%20Should%20You%20Use%3F



Are you upgrading to Java 12? (Check all that apply)

I'm still using Java 8! (45%, 135 Votes)

I am waiting for the next long term release before I upgrade. (33%, 100 Votes)

Yes! I'm upgrading to Java 12. (14%, 42 Votes)

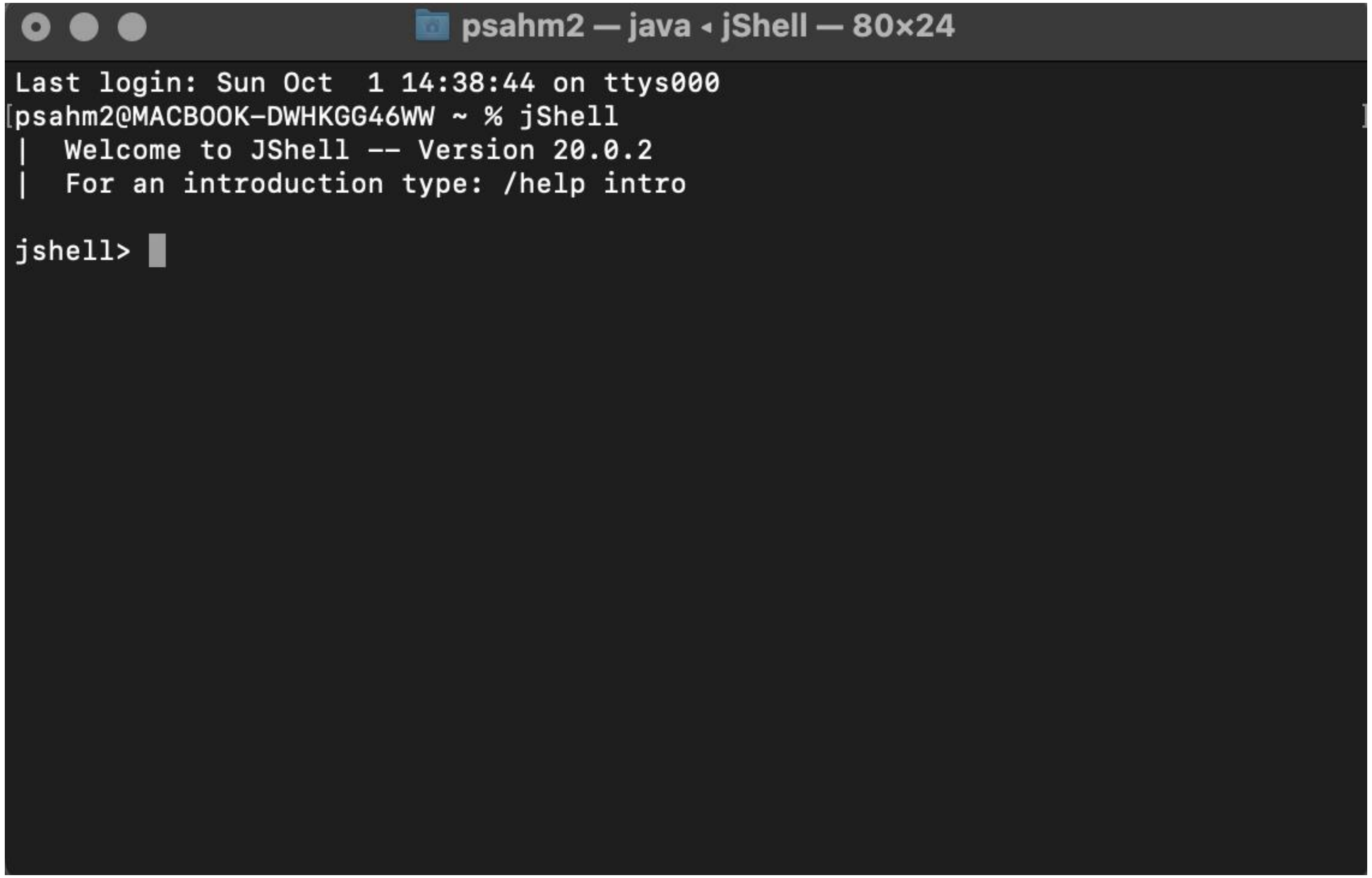
No, I am not upgrading to Java 12. (8%, 25 Votes)

Total Voters: **302**

https://jaxenter.com/java-12-adoption-poll-157164.html

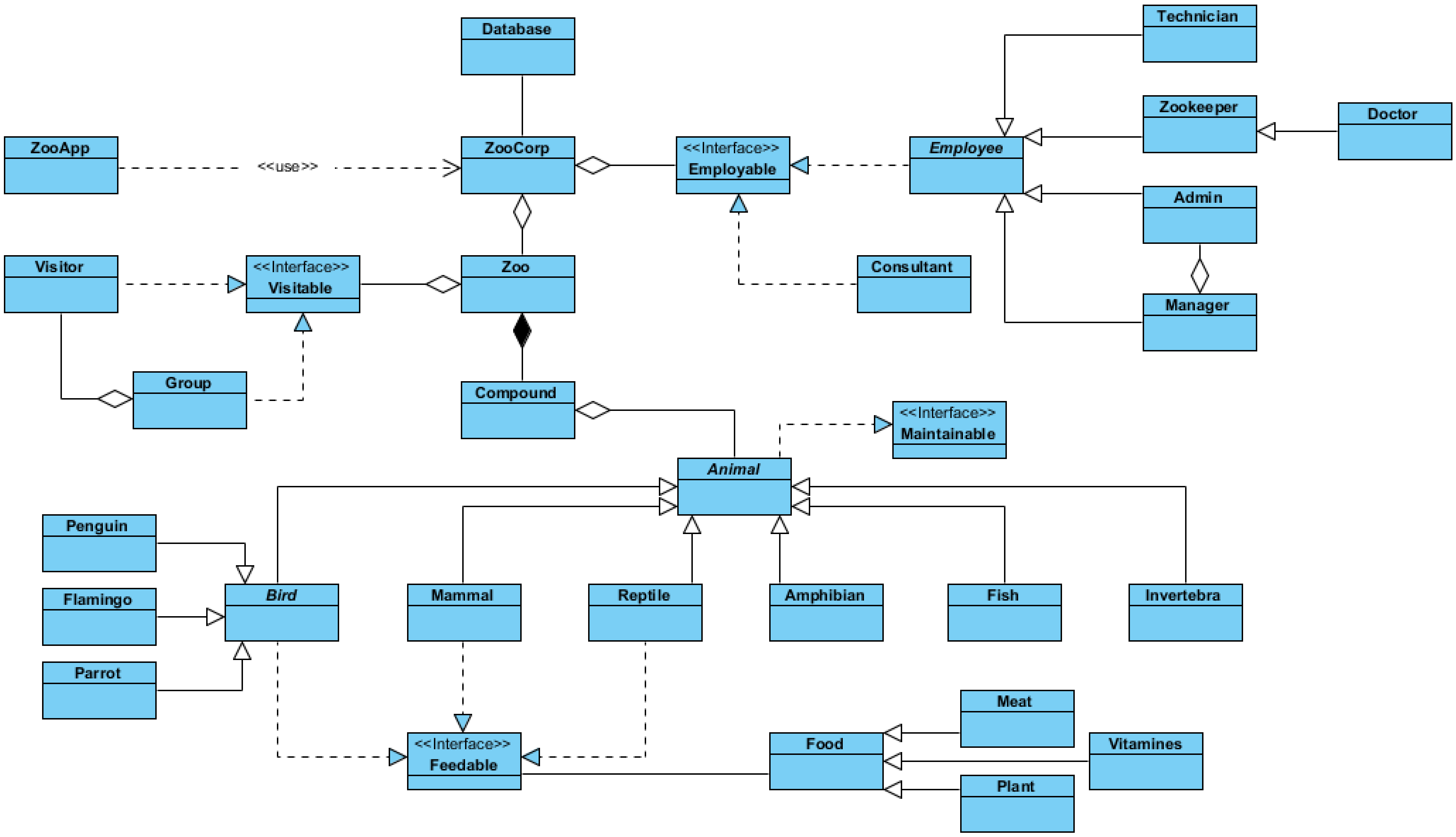
jShell

Live Demo of jShell



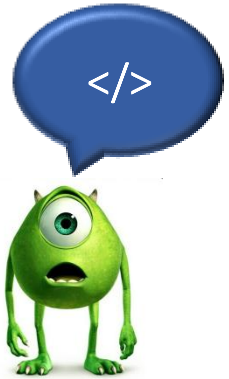
```
Last login: Sun Oct  1 14:38:44 on ttys000
psahm2@MACBOOK-DWHKGG46WW ~ % jShell
|  Welcome to JShell -- Version 20.0.2
|  For an introduction type: /help intro

jshell> 
```

basic ZooApp maintenance


Database
ZooApp
<<use>>
ZooCorp
<<Interface>>
Employable
Employee
Technician
Zookeeper
Doctor
Admin
Manager
Consultant
Visitor
<<Interface>>
Visitable
Zoo
Group
Compound
<<Interface>>
Maintainable
Animal
Penguin
Flamingo
Parrot
Bird
Mammal
Reptile
Amphibian
Fish
Invertebra
<<Interface>>
Feedable
Food
Meat
Vitamines
Plant

</>

Basic Maintenance – 2 min discussion in pairs



- What would you propose how we should get started?

Refactoring code

- Improved Code Quality
- Enhanced Readability
- Reduced Code Duplication
- Maintainability
- Scalability
- Performance Optimisation

Basic Maintenance

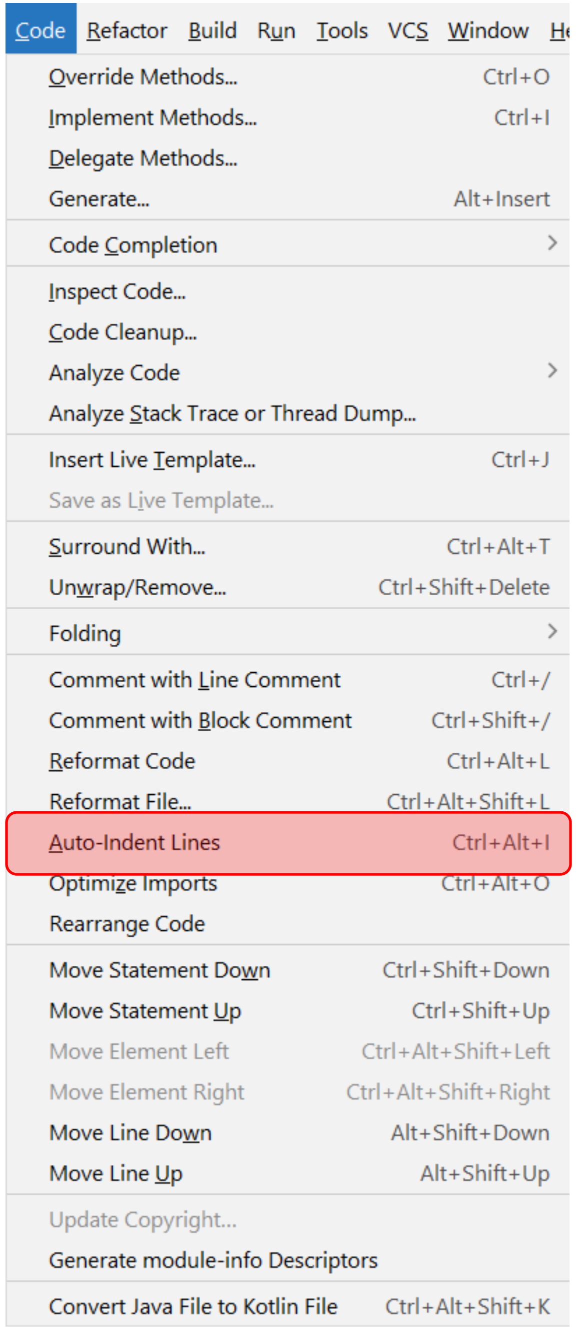
- Goal
  - Improving current code without creating new functionality
  - Keep the code well organised and easy to maintain in the future
- Examples (see also Lecture 1A)
  - Adding unit tests
  - Renaming
  - Packaging
  - Avoiding code duplication
  - Comments and documentation
  - Moving code to where it belongs
  - Using correct access modifiers (encapsulation)

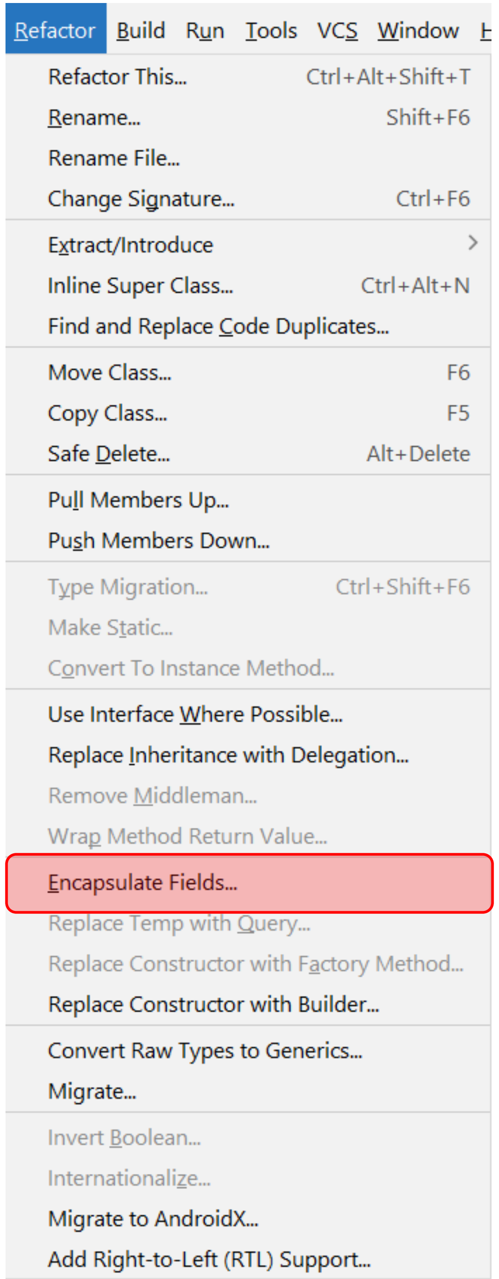
Basic Maintenance

- ZooApp Development
  - Make ArrayLists private
  - Make all methods/classes used by the constructor "final"
  - Remove obsolete "this" keywords
  - Consider what the interface to the outside world should look like
    - Remove redundant setters and getters
  - Check the module-info.java file

Basic Maintenance

- Help from IntelliJ


Code
Refactor
Build
Run
Tools
VCS
Window
He
Override Methods... Ctrl+O
Implement Methods... Ctrl+I
Delegate Methods...
Generate... Alt+Insert
Code Completion
Inspect Code...
Code Cleanup...
Analyze Code
Analyze Stack Trace or Thread Dump...
Insert Live Template... Ctrl+J
Save as Live Template...
Surround With... Ctrl+Alt+T
Unwrap/Remove... Ctrl+Shift+Delete
Folding
Comment with Line Comment Ctrl+/
Comment with Block Comment Ctrl+Shift+/
Reformat Code Ctrl+Alt+L
Reformat File... Ctrl+Alt+Shift+L
Auto-Indent Lines Ctrl+Alt+I
Optimize Imports Ctrl+Alt+O
Rearrange Code
Move Statement Down Ctrl+Shift+Down
Move Statement Up Ctrl+Shift+Up
Move Element Left Ctrl+Alt+Shift+Left
Move Element Right Ctrl+Alt+Shift+Right
Move Line Down Alt+Shift+Down
Move Line Up Alt+Shift+Up
Update Copyright...
Generate module-info Descriptors
Convert Java File to Kotlin File Ctrl+Alt+Shift+K


Refactor
Build
Run
Tools
VCS
Window
H
Refactor This... Ctrl+Alt+Shift+T
Rename... Shift+F6
Rename File...
Change Signature... Ctrl+F6
Extract/Introduce
Inline Super Class... Ctrl+Alt+N
Find and Replace Code Duplicates...
Move Class... F6
Copy Class... F5
Safe Delete... Alt+Delete
Pull Members Up...
Push Members Down...
Type Migration... Ctrl+Shift+F6
Make Static...
Convert To Instance Method...
Use Interface Where Possible...
Replace Inheritance with Delegation...
Remove Middleman...
Wrap Method Return Value...
Encapsulate Fields...
Replace Temp with Query...
Replace Constructor with Factory Method...
Replace Constructor with Builder...
Convert Raw Types to Generics...
Migrate...
Invert Boolean...
Internationalize...
Migrate to AndroidX...
Add Right-to-Left (RTL) Support...

Some final remarks …