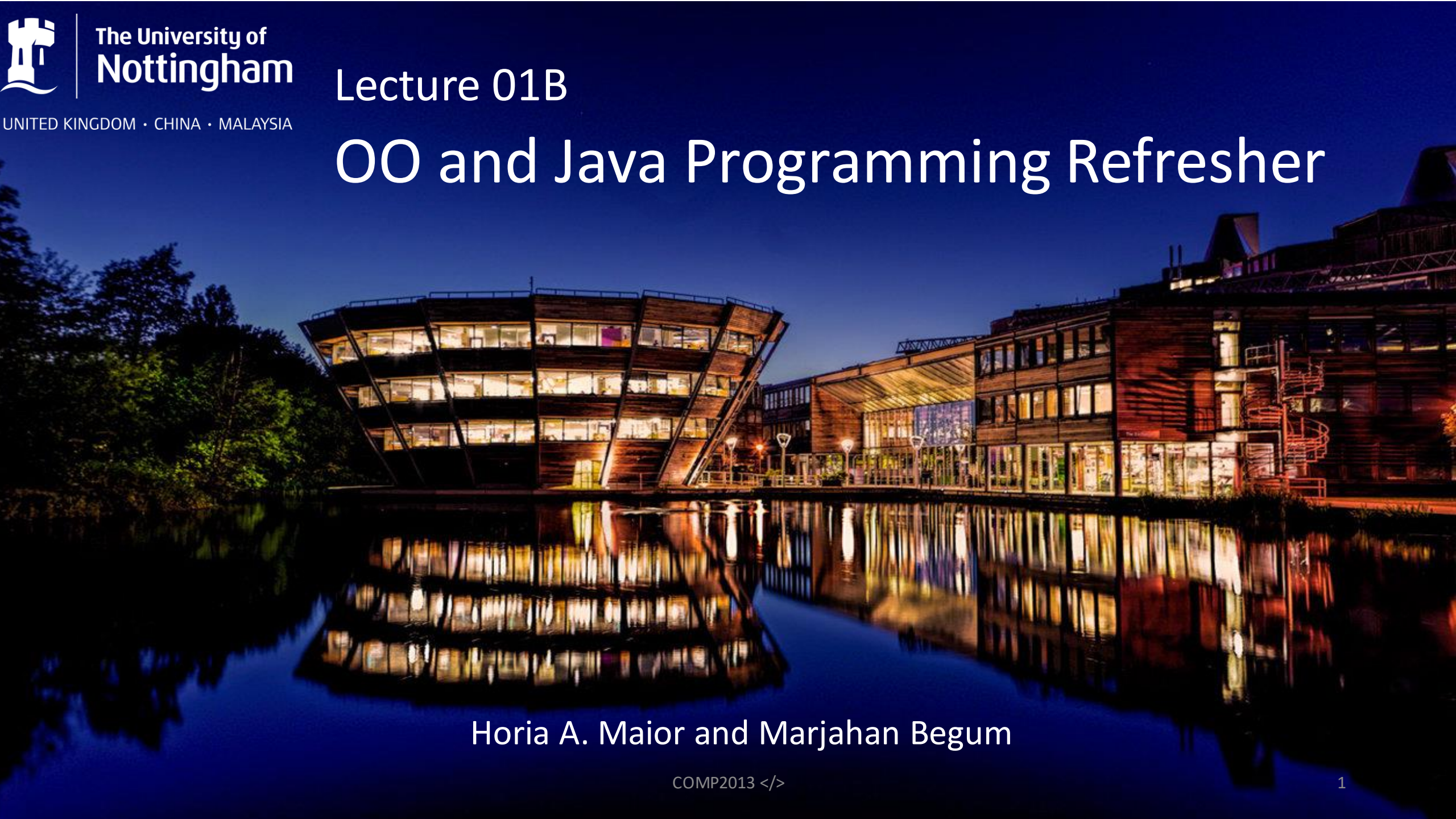


Lecture 01B

OO and Java Programming Refresher

Horia A. Maior and Marjahan Begum

COMP2013

Developing Maintainable Software

Lecture 01B (2023)

OO and Java Programming Refresher (1/2)

Horia A. Maior and Marjahan Begum

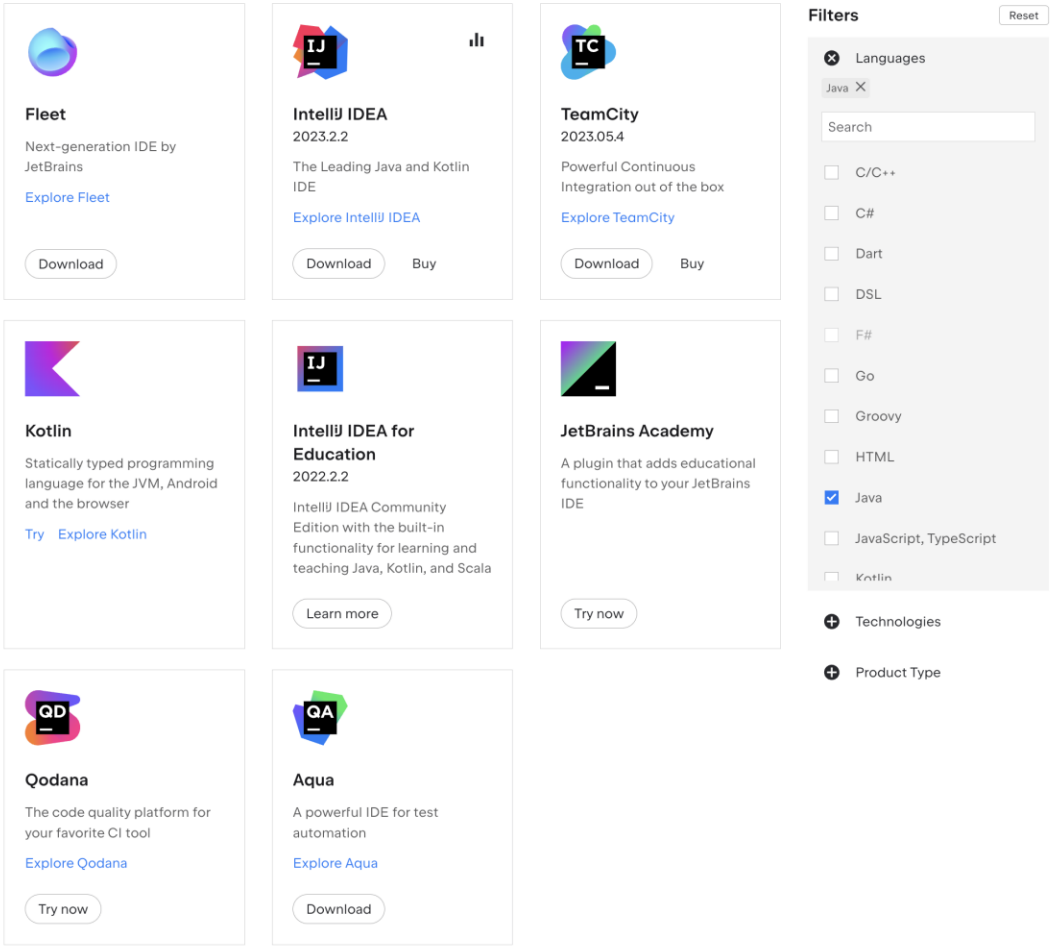
University of Nottingham
UK | CHINA | MALAYSIA

Topics for this Week

- Lecture 01A
  - Welcome to the module
  - Developing maintainable software
  - Why COMP2013?
  - Challenges of software maintenance
- Lecture 01B
  - Labsheet
  - OO concepts & Java programming refresher (we complete this next week)
- Lab 01 – extra support in the lab
  - IntelliJ basics
  - Practicing Java basics
  - Working with existing code

Integrated Development Environment – IDEs

- IntelliJ IDE, JetBrains



Lab sheet

Lab 01

- Fridays from 3-5
- IntelliJ IDE
- Different IntelliJ versions
  - Ultimate: Paid + much more functionality compared to community edition
  - Community: Free + open source
  - Educational: Community + one can practice programming with i
- Using Teams
  - Please use the "Questions" channel for help requests

Lab 01

- Evolution of IDEs and JDKs
  - There are many different ways of doing things and IDEs are evolving every day; no one can stay on top of this, unless perhaps you are a professional software developer
  - In this module we will teach you the ways that we have found work best
    - We are not claiming they are the best ways, but they work
    - Feel free to search the internet and learn some alternative best practices from the professionals

Lab 01 Overview

LAB 01: INTELLIJ, AND "HELLO COMP2013 WORLD"

Content:

1. Preparation
2. Get familiar with IntelliJ
3. Implement a simple object-oriented example
4. Working with existing code

Everyone has different levels of experience with IDEs and OO programming. Jump in at the section, which suits your abilities.

NB: We do not expect everyone to solve all the "**Challenges**" listed below immediately. If you could not solve them now, you might want to consider coming back to them later in the module, once you have a bit more experience with the IDE and Java.

Please note that we have used a Windows environment to prepare this task sheet. If you are using a Linux or Mac environment and get stuck, please ask us, and we will try to help.

PART 1: PREPARATION

To maintain software, you need to be familiar with modern Integrated Development Environments (IDEs). As we said in the first lecture, you should use the IntelliJ IDEA IDE[1] in this module. As for the programming language, everything from Java 12 onwards is suitable, but we recommend that you use on of the latest Java Development Kits.

Here are some tips for installing the JDK and IDE on your private machine:

- Java JDK (not JRE)
  - Download open-jdk GA release (https://jdk.java.net/19/)
    - Difference between GA and Reference release: https://stackoverflow.com/questions/50316397/difference-between-openjdk-10-and-java-se-10-reference-implementation
  - System environment > Set this up in System Variables (bottom) rather than User Variables (top) if it is your own machine
    - JAVA_HOME > Path to JDK folder
    - Path > add "%JAVA_HOME%\bin"
  - Check that java installation works, using "java -version"
    - If it shows Java 8, you can use "where java" to check out why
      - Remove "C:\Program Files (x86)\Common Files\Oracle\Java\javapath\java.exe" or move the java path variable ("%JAVA_HOME%\bin") to the top of the list
- IntelliJ IDEA IDE
  - Download IntelliJ Community Edition: https://www.jetbrains.com/idea/download/
  - Run IntelliJ installer > Choose: 64 bit launcher + associate .java with IntelliJ + update context menu (if you want)
  - Install and then run IntelliJ > need to agree to terms > IntelliJ should open

If you encounter any issues when installing the IDE and JDK on your local machine, let us know, and we are trying to help you to fix those issues.

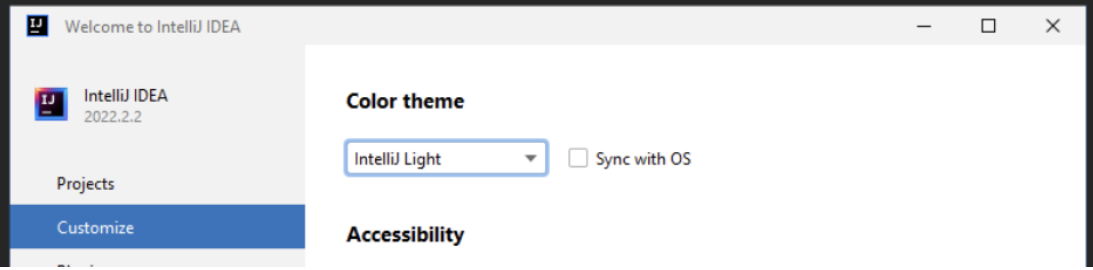
PART 2: GET FAMILIAR WITH INTELLIJ

The following information will guide you through programming a simple "Hello COMP2013 World" program in IntelliJ. Feel free to skip this if you know how to set up a new project in IntelliJ, using Java 12+ and write "Hello COMP2013 World" to the console.

One of the things introduced in Java 9 is formal modularity, in order to create structure inside large software systems. The new hierarchy in a Java project is as follows: Module > Package > Class > Fields and Methods. In one of the later lectures, we will talk about the overall concept in more detail. For now, we only need to be aware that this concept exists, as we should have one additional file (module-info.java) in the new projects we create.

For the following demonstration, I will use IntelliJ IDEA 2022.2.2, which is the version installed on my machine. The university machines and the Virtual Desktop (VD) have IntelliJ IDEA 2022.1.3 installed, but the difference between the two versions is not drastic.

- Start IntelliJ. If you are using the VD, the easiest way to start IntelliJ IDEA is to type "intellij" into the windows search box.
- In the "Welcome to IntelliJ IDEA" screen choose {Customize > All settings ...} and in the coming screen {System Settings > Default project directory} and choose your workspace directory, where you want to save your projects in. Always make sure you are using your personal network storage (I used "C:\Workspace\Java\IntelliJ", but yours will be different).



https://www.oracle.com/uk/corporate/features/understanding-java-9-modules.html

PART 3: IMPLEMENT A SIMPLE OBJECT-ORIENTED EXAMPLE

Create a new Java project, name it BikeProject, and add a new package com.theBestBikeShop to the src folder. Create a module-info.java file and add relevant information (opens com.theBestBikeShop). Download the code Bicycle.java from Moodle and import it into your project. To do this, drag and drop the file in the src\com.theBestBikeShop package, and then click {Refactor} in the appearing popup.

- If you want to check, where the Bicycle.java file is located, right click the file in the Project Explorer and select {Copy Path > Reference...}. In the popup, you can see the absolute path.

You can compile the project using {Build > Build Project} and there should be no errors, but it does not do anything, as it does not have a main() method yet.

Task: Extend the bike project.

- Create a BikeApp class including the main() method which instantiates a Bicycle object. A neat trick to create the main() method is to type main and use {TAB} for auto-complete.
- Write a line of code to change the speed of the bicycle you just created.
- Write a constructor for the Bicycle class, so that you can set the initial gear and speed values. This will create an error. Go to the BikeApp class and change it, so that the new constructor is used for generating the Bicycle object
- Add a method called switchLightStatus to the Bicycle class that turns the light on if it is off, and vice versa. Add a method currentState to the Bicycle class to print out the current speed, gear and light state in the console. Test the functionality of both added methods by adding some code to the BikeApp class

Let's suppose we also want a class for a mountain bike, which has specific features associated with it. It makes sense to create a subclass of a Bicycle class, i.e. inherit from it.

In the Project Explorer, right click your com.theBestBikeShop package and select {New > Java Class}. Name the new class MountainBike and add "... extends Bicycle" to the class signature. Then hoover with the mouse over the appearing red line and pressed {Alt + Shift + Enter} on the keyboard, which automatically created the required constructor.

Hey presto, your subclass is created.

We will cover more about inheriting from classes in the coming lectures. But for now, let's just add two Boolean variables to store whether a bike has a front suspension and a rear suspension.

- Modify the constructor of the MountainBike class to take in value for the two suspension variables, and set them in the constructor. Add a method called isFullSuspension, which returns *true* only if the bike has both front and rear suspension. In the BikeApp class, create objects for two different mountain bikes, one with full suspension, and one without.
- Override the currentState() method in the Bicycle class from within the MountainBike class, providing gear, speed, light status, and suspension state information. Test the functionality of the added method by adding some code to the BikeApp class.

Challenge: Automatically generate getters, setters, and constructors.

The IntelliJ IDE menu {Code} contains a lot of functionality for automating processes such as writing getters and setters and producing constructors. Delete all existing getters, setters, and constructors within your bike source code and use {Code > Generate ...} to get back to how the code looked before.

PART 4: WORKING WITH EXISTING CODE

In this part we will download and explore a larger existing codebase. To do this we will learn some tips for navigating project code. Please note: The following is written for Eclipse, but will work similarly in the IntelliJ IDE.

Challenge: Try to work out how it works in IntelliJ. Some advice is provided in orange.

Let's have a look at some real world source code: a game called "Diamond Hunter". You can find more information about this game here: https://www.youtube.com/watch?v=AA1XpWHhxw0.

First, you need to download the project zip file DiamondHunter.zip from Moodle, unzip it, and import the resulting folder into IntelliJ. In IntelliJ IDEA, just open the project. You might have to define the Project SDK. Use {File > Project Structure > Project Settings > Project > Project SDK} and choose the one you want to use from the pull-down menu] [If you want a real challenge, download DiamondHunterSrc.zip instead and set up your own project in Intellij IDEA.

HOW DO YOU FIND A SPECIFIC CLASS (OPEN A CLASS BY NAME)?

Let's assume we want to find the class GameState. An efficient way is to choose In IntelliJ use {Navigate > Class...} the class GameState that is listed. This class will then appear in the editor window.

HOW DO YOU GET AN OVERVIEW OF METHODS AND FIELDS OF A CLASS?

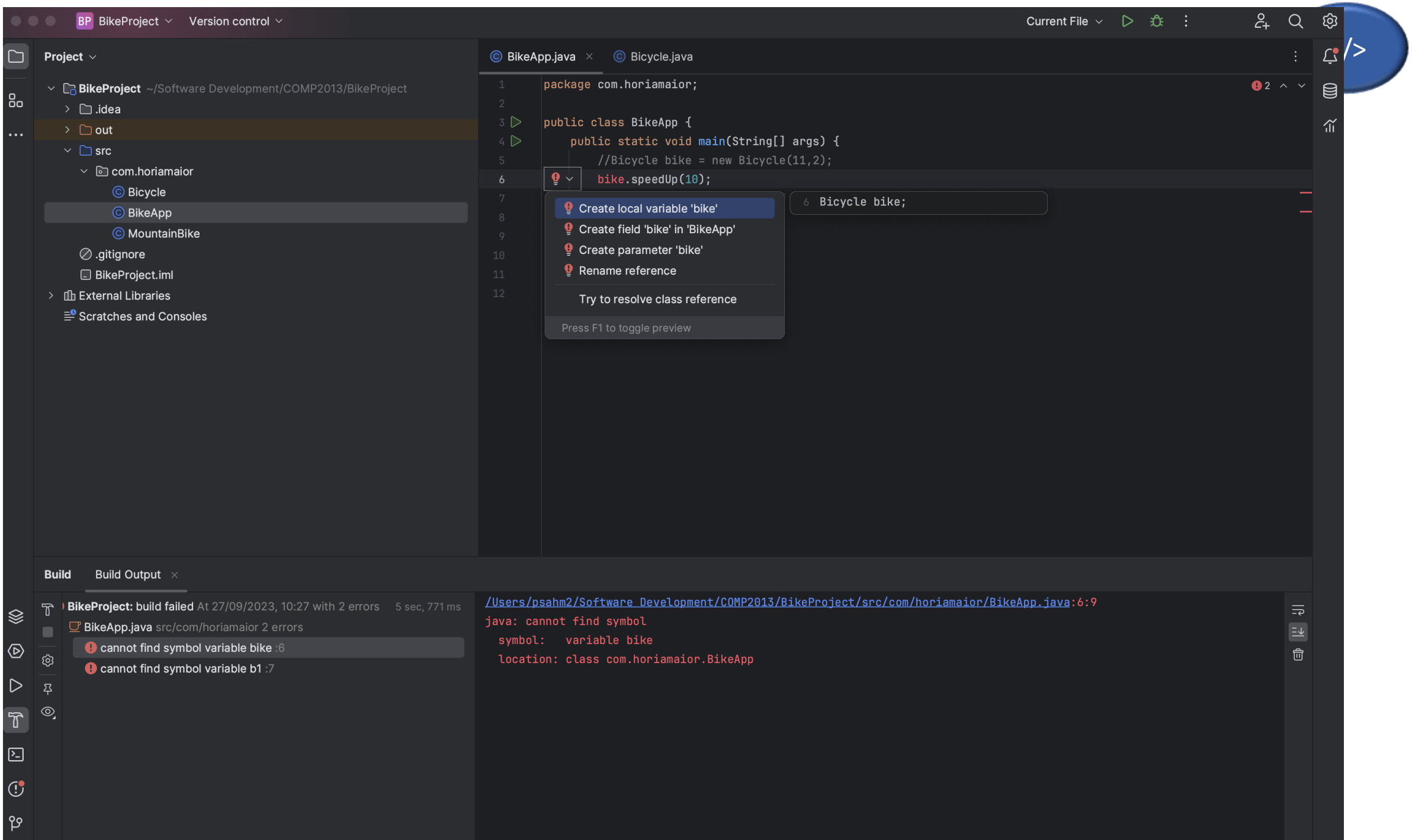
Use {View > Tool Window > Structure}

HOW CAN YOU SEE THE INHERITANCE TREE OF A SPECIFIC CLASS?

Lab – Looking ahead

- Error Handling and Debugging
  - Hoover over error
  - Click on the line and the appearing red light bulb
  - Compile the code, ignoring the warning, and you get a list of errors
  - Following up errors > always start reading them from the bottom to the top

  - You need to try to solve fix the code yourselves!!
  - Talk to a rubber duck!
  - Search the web
    - You cannot just ask us without you spending enough time on it. This is how you learn!


BikeProject
Version control
Current File
Project
BikeProject ~/Software Development/COMP2013/BikeProject
.idea
out
src
com.horiamaior
Bicycle
BikeApp
MountainBike
.gitignore
BikeProject.iml
External Libraries
Scratches and Consoles
BikeApp.java
Bicycle.java
package com.horiamaior;
public class BikeApp {
public static void main(String[] args) {
//Bicycle bike = new Bicycle(11,2);
bike.speedUp(10);
Create local variable 'bike'
Create field 'bike' in 'BikeApp'
Create parameter 'bike'
Rename reference
Try to resolve class reference
Press F1 to toggle preview
6 Bicycle bike;
Build
Build Output
BikeProject: build failed At 27/09/2023, 10:27 with 2 errors 5 sec, 771 ms
BikeApp.java src/com/horiamaior 2 errors
cannot find symbol variable bike :6
cannot find symbol variable b1 :7
/Users/psahm2/Software Development/COMP2013/BikeProject/src/com/horiamaior/BikeApp.java:6:9
java: cannot find symbol
symbol: variable bike
location: class com.horiamaior.BikeApp

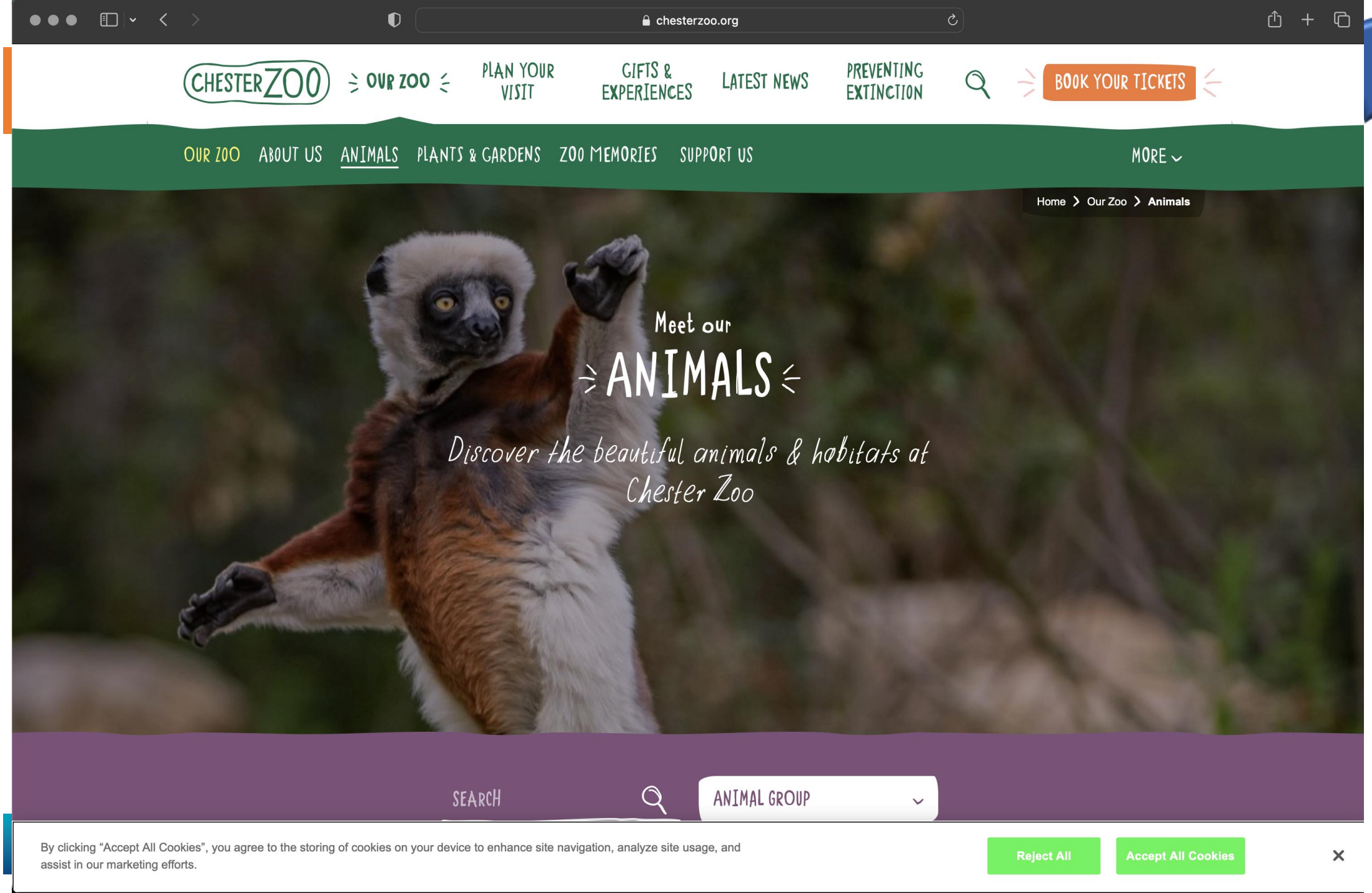
oo concepts & java programming refresher

OO Design – Class Diagram and Initial Planning

- Requires you to sketch out classes and methods to implement technical problems or real life objects.
- Think about Input, Output,
- Key Stakeholders,
- How things interact

OO Design - Problems/steps in this process

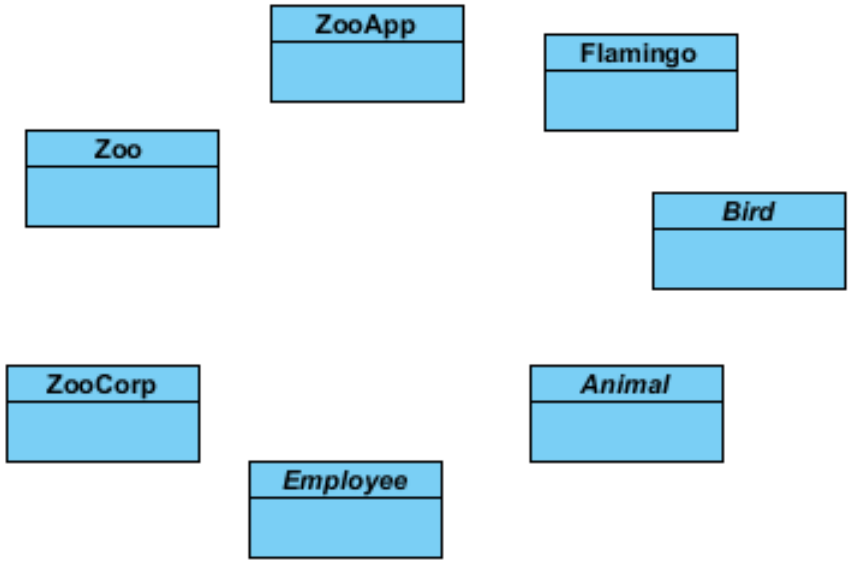
- Handling Ambiguity
- Define Core Objects
- Analise relationships
- Investigate actions


chesterzoo.org
CHESTER ZOO
OUR ZOO
PLAN YOUR VISIT
GIFTS & EXPERIENCES
LATEST NEWS
PREVENTING EXTINCTION
BOOK YOUR TICKETS
OUR ZOO
ABOUT US
ANIMALS
PLANTS & GARDENS
ZOO MEMORIES
SUPPORT US
MORE
Home > Our Zoo > Animals
Meet our
ANIMALS
Discover the beautiful animals & habitats at Chester Zoo
SEARCH
ANIMAL GROUP
By clicking "Accept All Cookies", you agree to the storing of cookies on your device to enhance site navigation, analyze site usage, and assist in our marketing efforts.
Reject All
Accept All Cookies

Step 1: Ambiguity

- 6Ws
  - Who?
  - What?
  - Where?
  - When?
  - How?
  - Why?

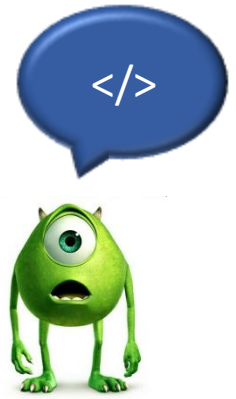
Step 2: Define Core Objects



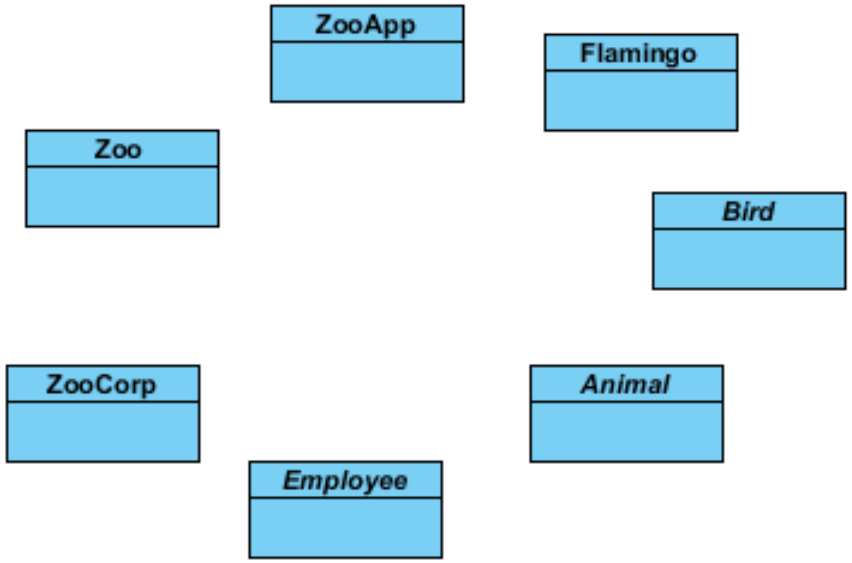
Step 3: Analyse Relationships

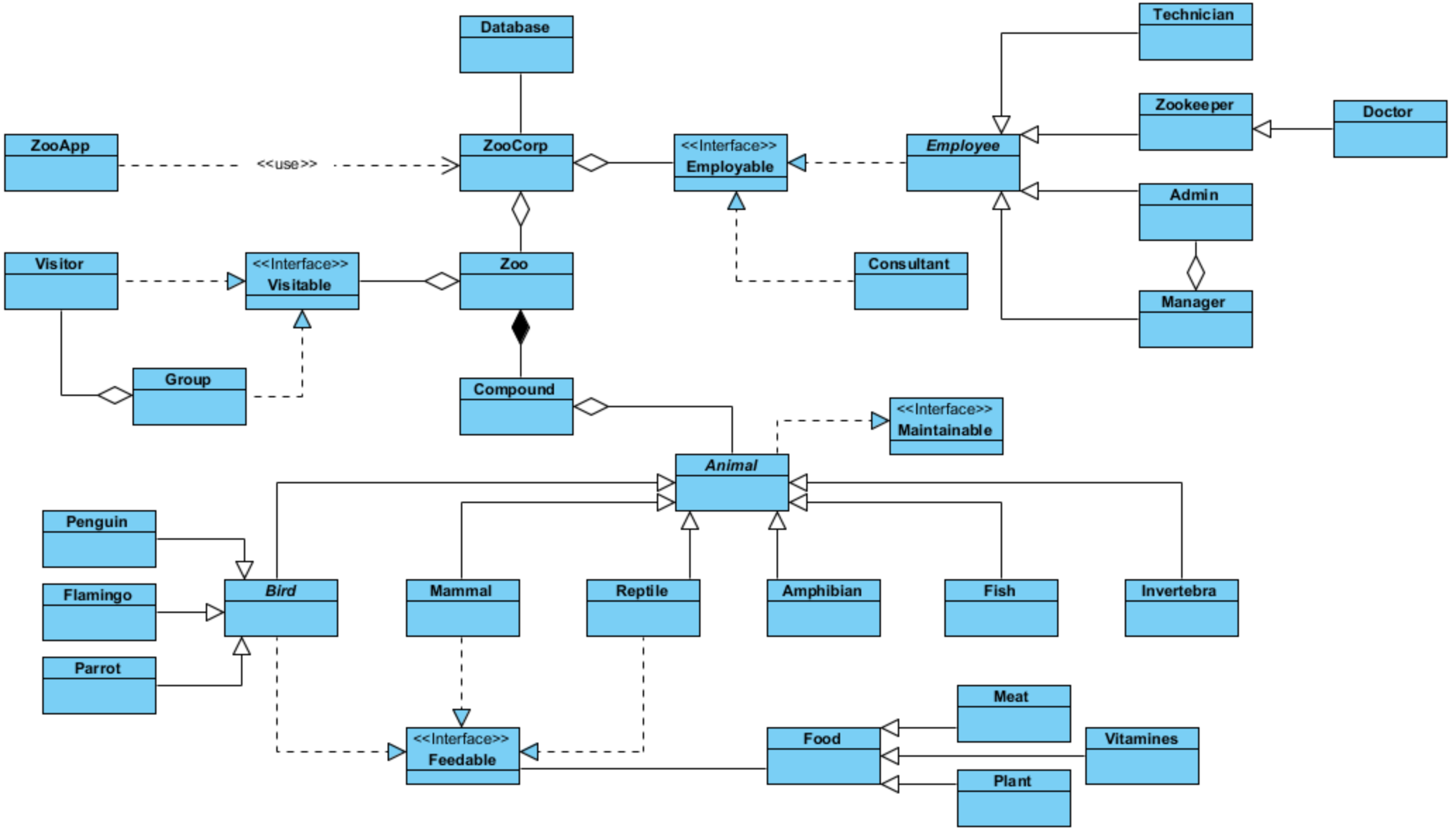
Step 4: Investigate Actions

Case Study: Zoo Management



- Come up with a draft class diagram for the Zoom Management problem
  - Note that this is only a small choice of relevant classes!




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

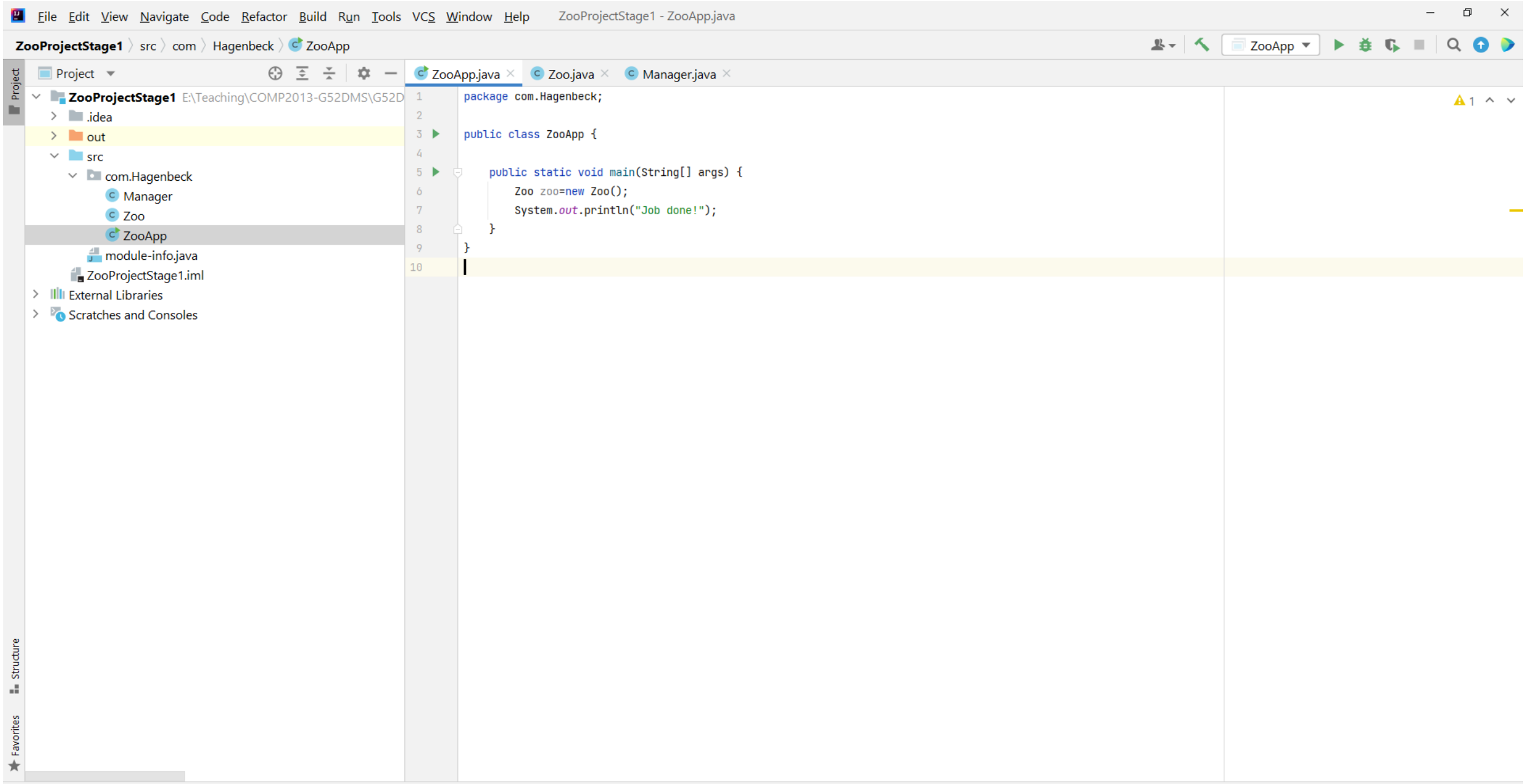
What assumptions have you made?

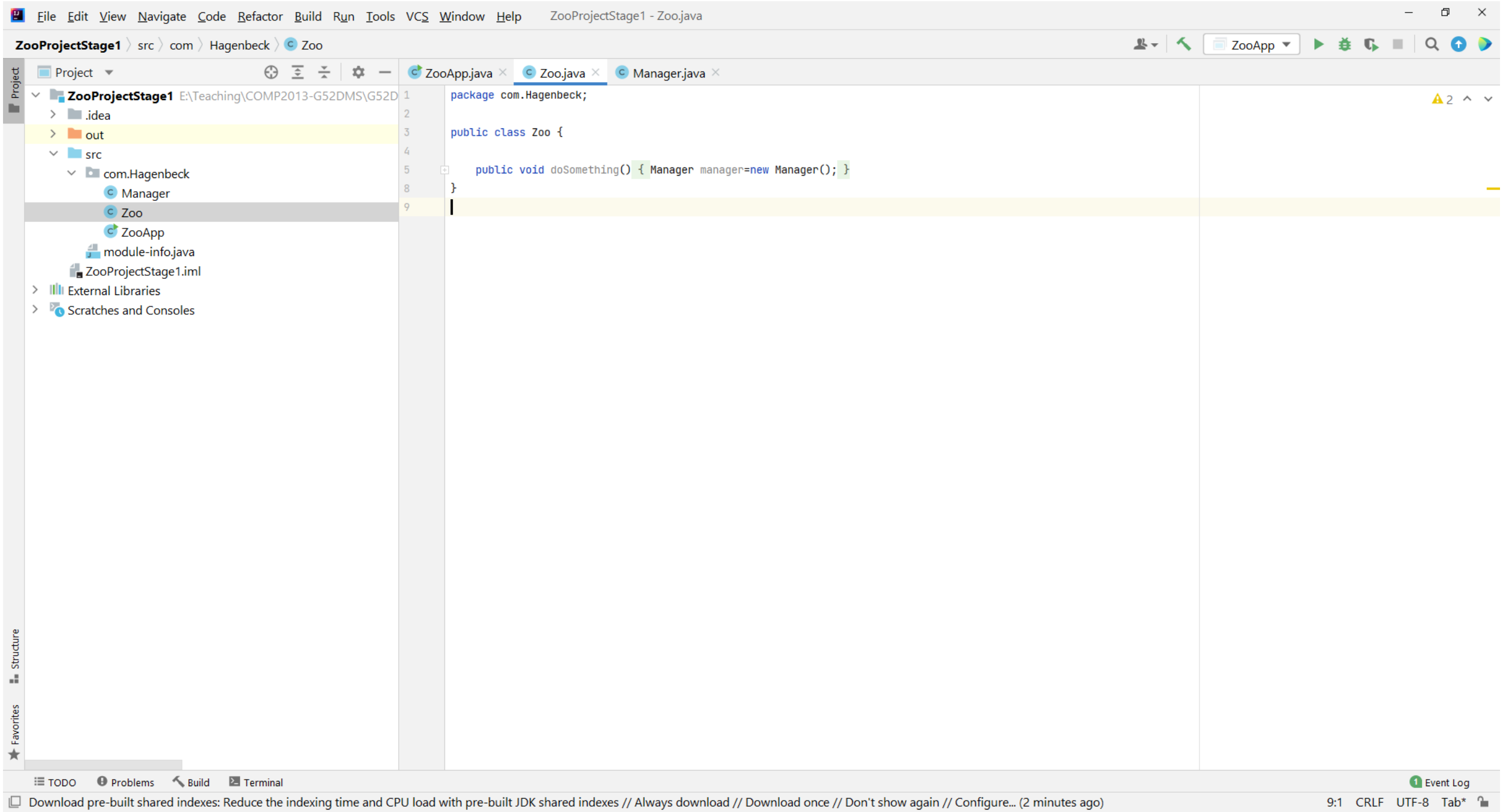
- What were the assumptions that you made when designing this?
- How would it change if you worked with a customer?

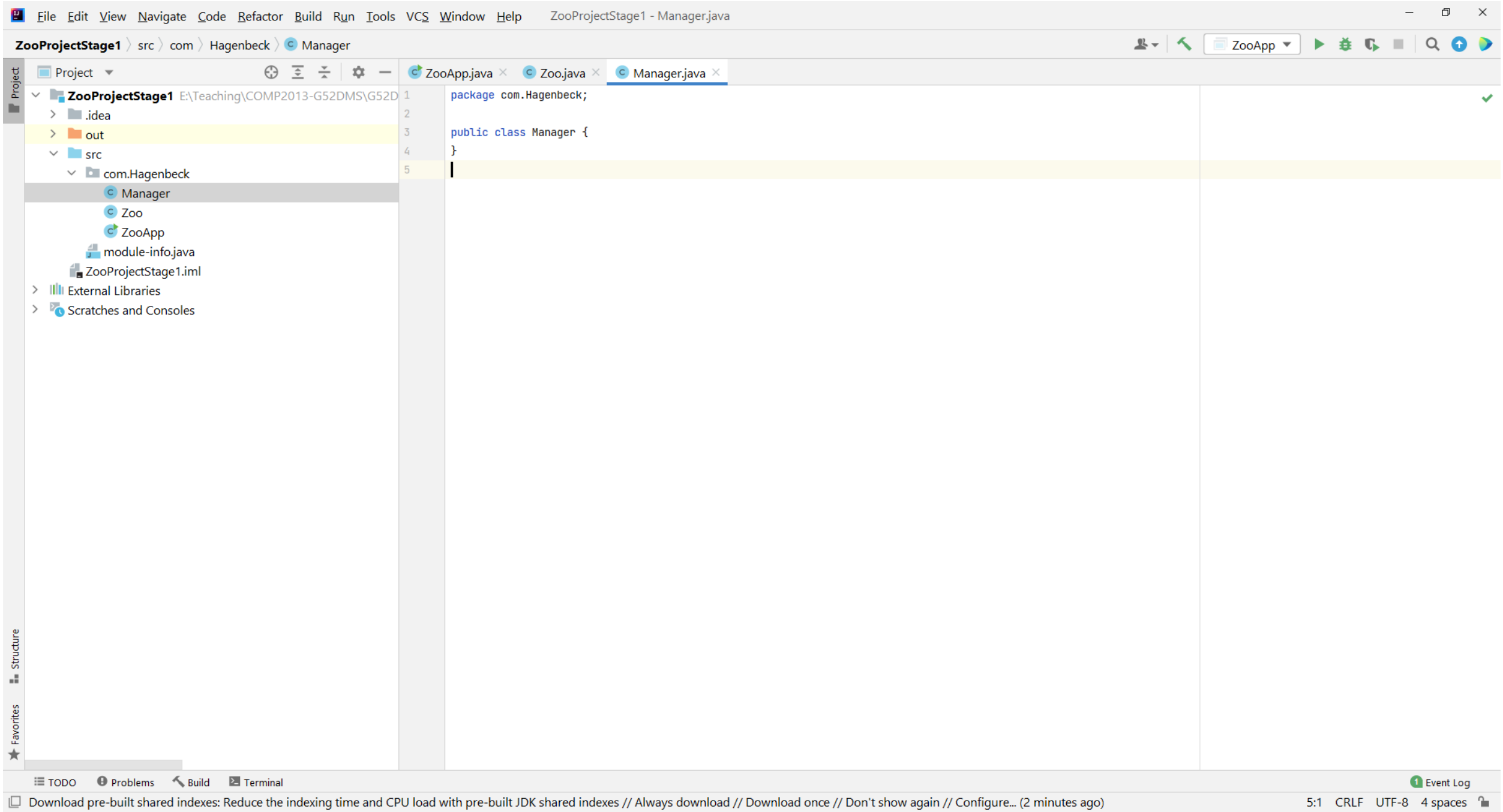
Case Study: Zoo Management

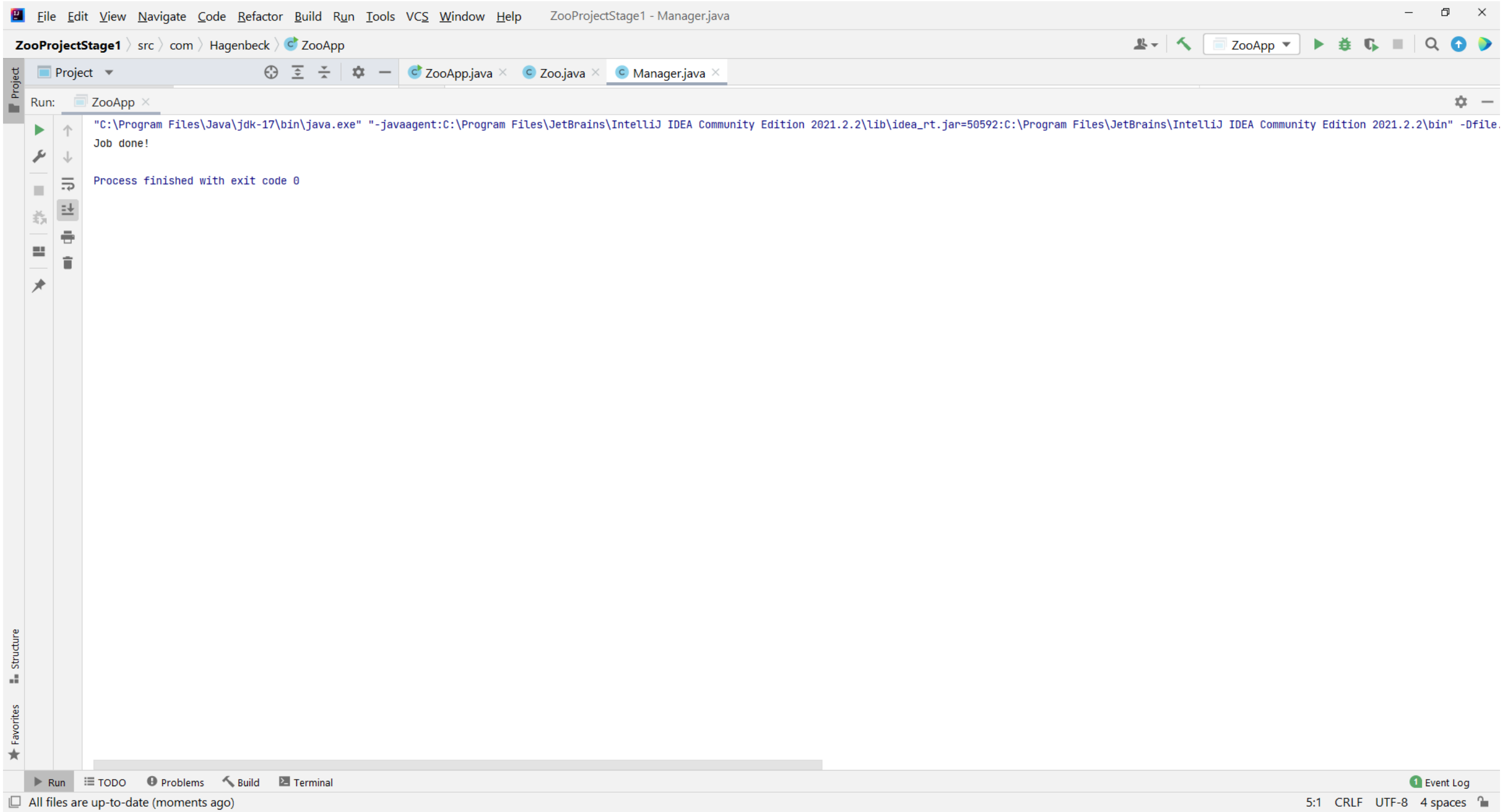
As we focus on Java basics today we want to keep it simple ...

ZooProjectStage1


File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
ZooProjectStage1 - ZooApp.java
ZooProjectStage1 〉 src 〉 com 〉 Hagenbeck 〉 ZooApp
ZooApp
Project
ZooProjectStage1 E:\Teaching\COMP2013-G52DMS\G52D
.idea
out
src
com.Hagenbeck
Manager
Zoo
ZooApp
module-info.java
ZooProjectStage1.iml
External Libraries
Scratches and Consoles
ZooApp.java
Zoo.java
Manager.java
1 package com.Hagenbeck;
2
3 public class ZooApp {
4
5 public static void main(String[] args) {
6 Zoo zoo=new Zoo();
7 System.out.println("Job done!");
8 }
9 }
10
Structure
Favorites
TODO
Problems
Build
Terminal
Event Log
Download pre-built shared indexes: Reduce the indexing time and CPU load with pre-built JDK shared indexes // Always download // Download once // Don't show again // Configure... (a minute ago)
10:1 CRLF UTF-8 Tab*


File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
ZooProjectStage1 - Zoo.java
ZooProjectStage1 〉 src 〉 com 〉 Hagenbeck 〉 Zoo
ZooApp
Project
ZooProjectStage1 E:\Teaching\COMP2013-G52DMS\G52D
.idea
out
src
com.Hagenbeck
Manager
Zoo
ZooApp
module-info.java
ZooProjectStage1.iml
External Libraries
Scratches and Consoles
ZooApp.java
Zoo.java
Manager.java
1 package com.Hagenbeck;
2
3 public class Zoo {
4
5 public void doSomething() { Manager manager=new Manager(); }
8 }
9
2
Structure
Favorites
TODO
Problems
Build
Terminal
Event Log
Download pre-built shared indexes: Reduce the indexing time and CPU load with pre-built JDK shared indexes // Always download // Download once // Don't show again // Configure... (2 minutes ago)
9:1 CRLF UTF-8 Tab*


File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
ZooProjectStage1 - Manager.java
ZooProjectStage1 › src › com › Hagenbeck › Manager
ZooApp
Project
ZooProjectStage1 E:\Teaching\COMP2013-G52DMS\G52D
.idea
out
src
com.Hagenbeck
Manager
Zoo
ZooApp
module-info.java
ZooProjectStage1.iml
External Libraries
Scratches and Consoles
ZooApp.java
Zoo.java
Manager.java
1 package com.Hagenbeck;
2
3 public class Manager {
4 }
5
Structure
Favorites
TODO
Problems
Build
Terminal
Event Log
Download pre-built shared indexes: Reduce the indexing time and CPU load with pre-built JDK shared indexes // Always download // Download once // Don't show again // Configure... (2 minutes ago)
5:1 CRLF UTF-8 4 spaces


File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
ZooProjectStage1 - Manager.java
ZooProjectStage1 src com Hagenbeck ZooApp
ZooApp
Project
ZooApp.java
Zoo.java
Manager.java
Run: ZooApp
"C:\Program Files\Java\jdk-17\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2021.2.2\lib\idea_rt.jar=50592:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2021.2.2\bin" -Dfile.
Job done!
Process finished with exit code 0
Run
TODO
Problems
Build
Terminal
Event Log
All files are up-to-date (moments ago)
5:1 CRLF UTF-8 4 spaces

Basic OO Concepts

- Object-oriented programming is founded on these ideas:
  - **Abstraction**: Simple things like objects represent more complex underlying code and data
  - **Encapsulation**: The ability to protect some components of the object from external access
  - **Inheritance**: The ability for a class ("subclass") to extend or override functionality of another class ("superclass")

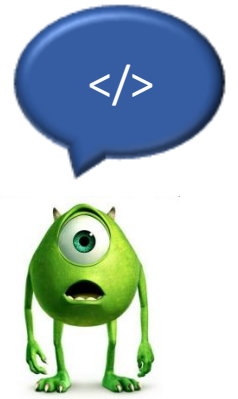
Basic OO Concepts

- Object-oriented programming is founded on these ideas:
  - **Polymorphism**: The provision of a single interface to entities of different types
    - Compile time polymorphism: Method overloading
    - Run time polymorphism: Method overriding

Programming Refresher

- Public vs. Private
- Accessors and Modifiers
- Encapsulation
- The "this" Keyword
- Constructors
- Passing by Value or Reference?
- More Hacks
- Static Fields and Methods

Public vs. Private



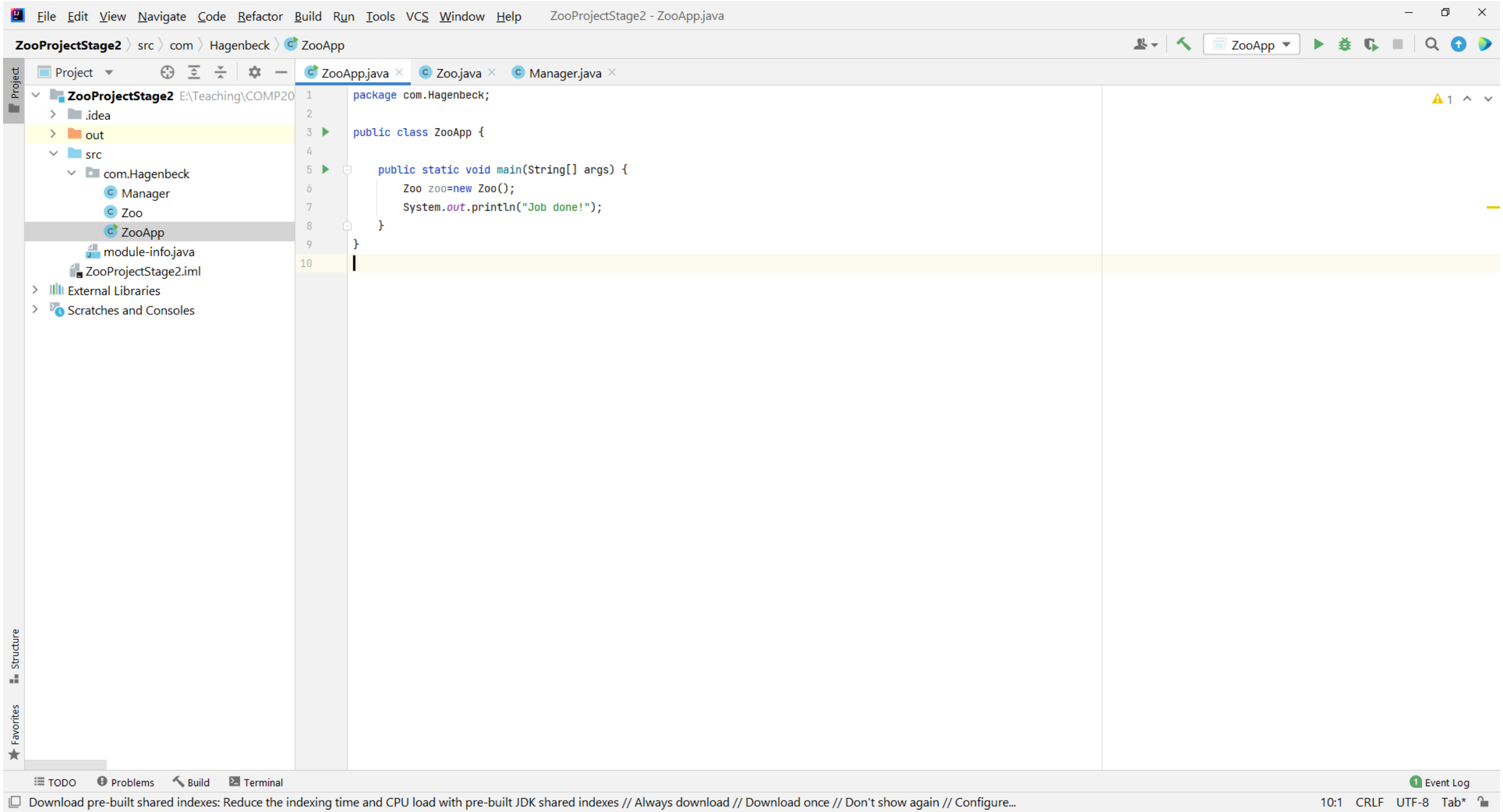
- What are the general access rules for constructors, methods, helper methods, fields, and static constants?
  - Constructors and methods: Usually declared public
  - Helper methods that are needed only inside the class: Usually declared private
  - Fields: Usually declared private
  - Static constants: Usually declared public

Accessors and Modifiers

- Accessors (also called Getters):
  - Methods that return values of private fields

- Modifiers (also called Mutators or Setters):
  - Methods that set values of private fields

ZooProjectStage2


File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
ZooProjectStage2 - ZooApp.java
ZooProjectStage2 〉 src 〉 com 〉 Hagenbeck 〉 ZooApp
ZooApp
Project
ZooProjectStage2 E:\Teaching\COMP20
.idea
out
src
com.Hagenbeck
Manager
Zoo
ZooApp
module-info.java
ZooProjectStage2.iml
External Libraries
Scratches and Consoles
ZooApp.java
Zoo.java
Manager.java
1 package com.Hagenbeck;
2
3 public class ZooApp {
4
5 public static void main(String[] args) {
6 Zoo zoo=new Zoo();
7 System.out.println("Job done!");
8 }
9 }
10
1
Structure
Favorites
TODO
Problems
Build
Terminal
Event Log
Download pre-built shared indexes: Reduce the indexing time and CPU load with pre-built JDK shared indexes // Always download // Download once // Don't show again // Configure...
10:1 CRLF UTF-8 Tab*

File Edit View Navigate Code Refactor Build Run Tools VCS Window Help ZooProjectStage2 - Zoo.java

ZooProjectStage2 〉 src 〉 com 〉 Hagenbeck 〉 Zoo

ZooApp

Project

ZooApp.java | Zoo.java | Manager.java

```
ZooProjectStage2 E:\Teaching\COMP20
  .idea
  out
  src
    com.Hagenbeck
      Manager
      Zoo
      ZooApp
    module-info.java
  ZooProjectStage2.iml
External Libraries
Scratches and Consoles
```

```
1  package com.Hagenbeck;
2
3  public class Zoo {
4
5      private String location;
6
7      public void doSomething() { Manager manager=new Manager(); }
10
11     public String getLocation() { return location; }
14
15     public void setLocation(String location) { this.location = location; }
18 }
19
```

4

Navigate Code Refactor Build Run Tools VCS Window He

com

Override Methods... Ctrl+O
Implement Methods... Ctrl+I
Delegate Methods...
Generate... Alt+Insert

Generate

Constructor
Getter
Setter
Getter and Setter
equals() and hashCode()
toString()
Override Methods... Ctrl+O
Delegate Methods...
Test...
Copyright

Structure Favorites

TODO Problems Build Terminal Event Log

Download pre-built shared indexes: Reduce the indexing time and CPU load with pre-built JDK shared indexes // Always download // Download once // Don't show again // Configure... (a minute ago) 19:1 CRLF UTF-8 Tab*

File Edit View Navigate Code Refactor Build Run Tools VCS Window Help ZooProjectStage2 - Manager.java

ZooProjectStage2 〉 src 〉 com 〉 Hagenbeck 〉 Manager

ZooApp

Project

ZooApp.java Zoo.java Manager.java

```
1 package com.Hagenbeck;
2
3 public class Manager {
4 }
5
```

- ZooProjectStage2 E:\Teaching\COMP20
  - .idea
  - out
  - src
    - com.Hagenbeck
      - Manager
      - Zoo
      - ZooApp
    - module-info.java
  - ZooProjectStage2.iml
- External Libraries
- Scratches and Consoles

Structure Favorites

TODO Problems Build Terminal Event Log

Download pre-built shared indexes: Reduce the indexing time and CPU load with pre-built JDK shared indexes // Always download // Download once // Don't show again // Configure... (a minute ago) 5:1 CRLF UTF-8 4 spaces

File Edit View Navigate Code Refactor Build Run Tools VCS Window Help ZooProjectStage2 - Manager.java

ZooProjectStage2 〉 src 〉 com 〉 Hagenbeck 〉 Manager

ZooApp

Project

ZooApp.java Zoo.java Manager.java

Run: ZooApp

```
"C:\Program Files\Java\jdk-17\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2021.2.2\lib\idea_rt.jar=50606:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2021.2.2\bin" -Dfile.
Job done!

Process finished with exit code 0
```

Run TODO Problems Build Terminal Event Log

All files are up-to-date (moments ago) 5:1 CRLF UTF-8 4 spaces

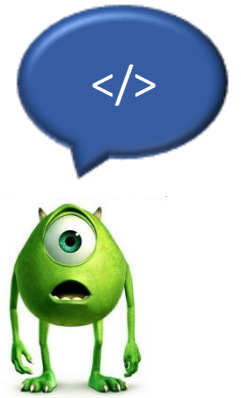
Encapsulation

- Hiding the implementation details of a class (making all fields and helper methods private) is called encapsulation
- Encapsulation helps with program maintenance: a change in one class does not affect other classes
- A client of a class interacts with the class only through well-documented public constructors and methods; this facilitates team development

The Keyword "this"

- "this" refers to the implicit parameter inside your class
  - A variable that stores the object on which a method is called
  - Refer to a field
    - this.field
  - Call a method
    - this.method(parameters);
  - One constructor can call another
    - this(parameters);

Constructors



- What is special about constructors (compared to other method types)?

Constructors

- The constructor method is a special method of a class for creating object instances of that class
  - They often initialise an object's fields
  - They do not have a return type
  - All constructors in a class have the same name (the name of the class)
  - They may take parameters
- If a class has more than one constructor, they must have different numbers and/or types of parameters (constructor overloading)
- Important!
  - Java provides a default constructor for a specific class
  - If you define a constructor for a class, Java does not provide the default constructor any more

Constructors

- Constructors of a class can call each other using the keyword "this" (referred to as constructor chaining) - a good way to avoid duplicating code

- Constructors are invoked using the operator new.
  - Declare a reference variable of the required type and then invoke the constructor method after the "new" keyword

- Parameters passed to "new" must match the number, types, and order of parameters expected by one of the constructors.

ZooProjectStage3

File Edit View Navigate Code Refactor Build Run Tools VCS Window Help ZooProjectStage3 - ZooApp.java

ZooProjectStage3 〉 src 〉 com 〉 Hagenbeck 〉 ZooApp

ZooApp

Project

ZooProjectStage3 E:\Teaching\COMP20
- .idea
- out
- src
  - com.Hagenbeck
    - Manager
    - Zoo
    - ZooApp
  - module-info.java
- ZooProjectStage3.iml
- External Libraries
- Scratches and Consoles

ZooApp.java | Zoo.java | Manager.java

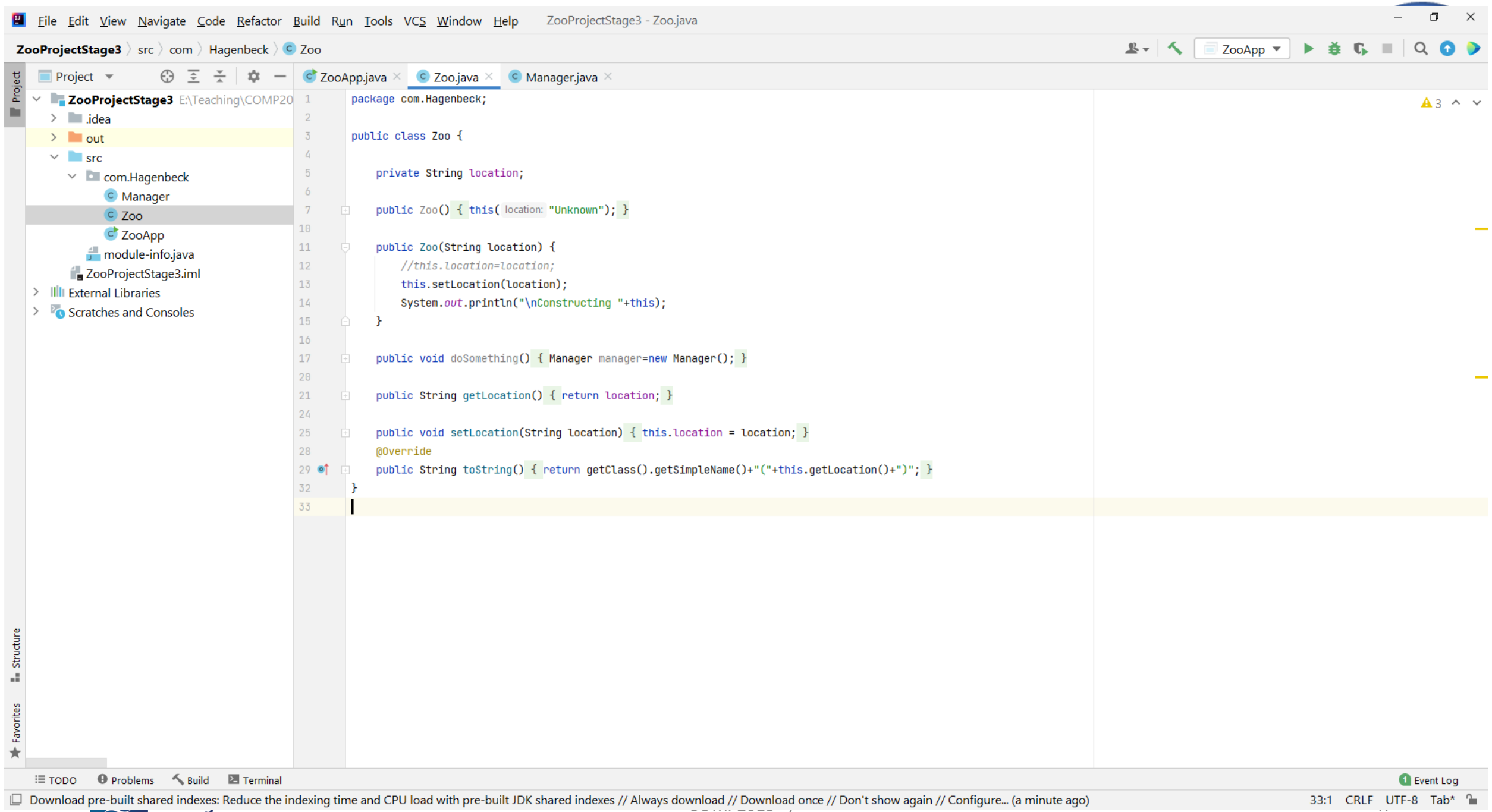
```
1  package com.Hagenbeck;
2
3  public class ZooApp {
4
5      public static void main(String[] args) {
6          //Zoo zoo=new Zoo();
7          Zoo zoo=new Zoo( location: "Hamburg");
8          System.out.println("\nZoo:"+zoo);
9      }
10 }
11
```

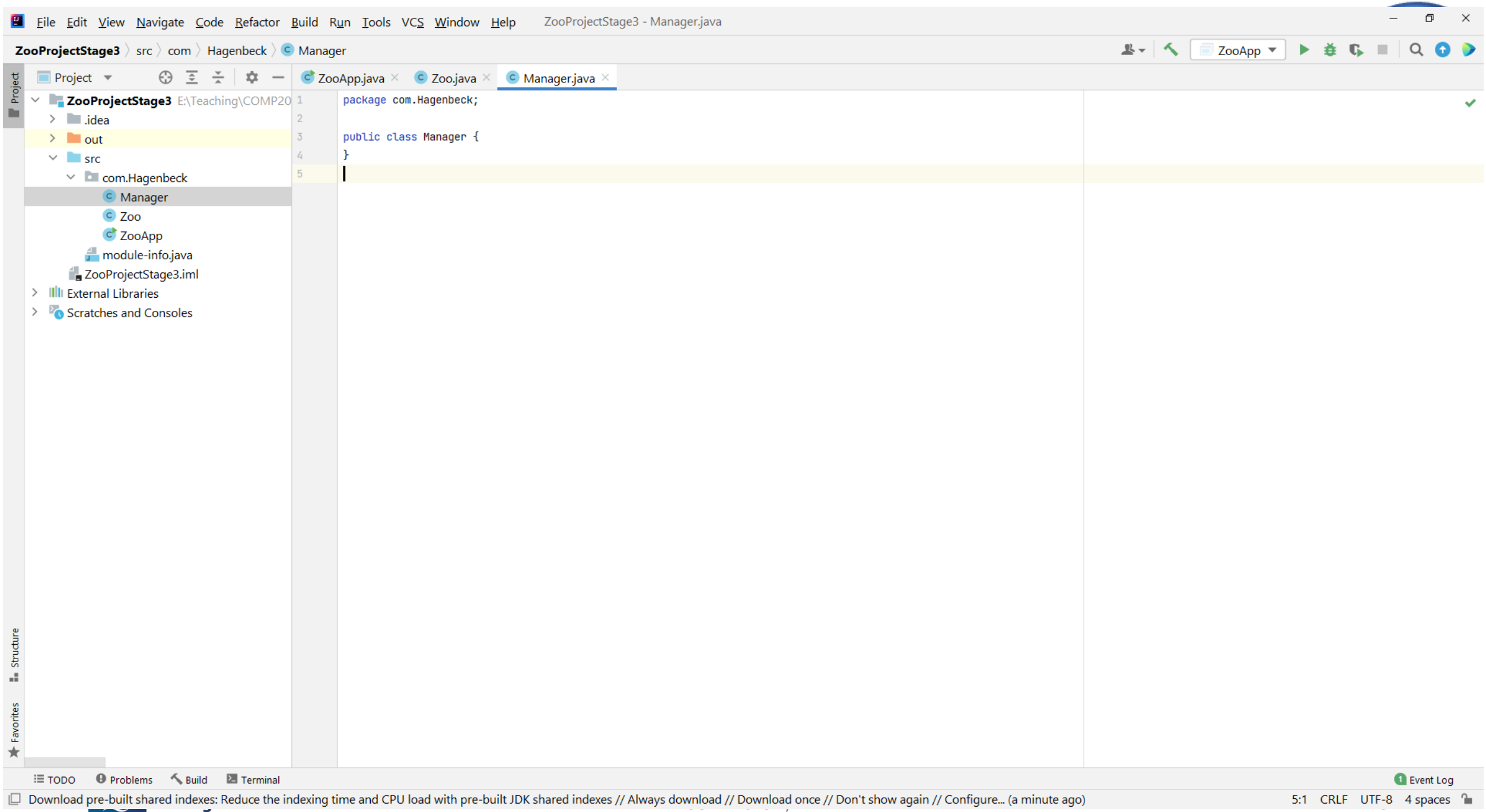
Structure Favorites

TODO Problems Build Terminal Event Log

Download pre-built shared indexes: Reduce the indexing time and CPU load with pre-built JDK shared indexes // Always download // Download once // Don't show again // Configure...

11:1 CRLF UTF-8 Tab*


File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
ZooProjectStage3 - Zoo.java
ZooProjectStage3 〉 src 〉 com 〉 Hagenbeck 〉 Zoo
ZooApp
Project
ZooProjectStage3 E:\Teaching\COMP20
.idea
out
src
com.Hagenbeck
Manager
Zoo
ZooApp
module-info.java
ZooProjectStage3.iml
External Libraries
Scratches and Consoles
ZooApp.java
Zoo.java
Manager.java
1 package com.Hagenbeck;
2
3 public class Zoo {
4
5 private String location;
6
7 public Zoo() { this(location: "Unknown"); }
10
11 public Zoo(String location) {
12 //this.location=location;
13 this.setLocation(location);
14 System.out.println("\nConstructing "+this);
15 }
16
17 public void doSomething() { Manager manager=new Manager(); }
20
21 public String getLocation() { return location; }
24
25 public void setLocation(String location) { this.location = location; }
28 @Override
29 public String toString() { return getClass().getSimpleName()+"("+this.getLocation()+")"; }
32 }
33
3
Structure
Favorites
TODO
Problems
Build
Terminal
Event Log
Download pre-built shared indexes: Reduce the indexing time and CPU load with pre-built JDK shared indexes // Always download // Download once // Don't show again // Configure... (a minute ago)
33:1 CRLF UTF-8 Tab*
UK | CHINA | MALAYSIA


File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
ZooProjectStage3 - Manager.java
ZooProjectStage3 〉 src 〉 com 〉 Hagenbeck 〉 Manager
ZooApp
Project
ZooProjectStage3 E:\Teaching\COMP20
.idea
out
src
com.Hagenbeck
Manager
Zoo
ZooApp
module-info.java
ZooProjectStage3.iml
External Libraries
Scratches and Consoles
ZooApp.java
Zoo.java
Manager.java
1 package com.Hagenbeck;
2
3 public class Manager {
4 }
5
Structure
Favorites
TODO
Problems
Build
Terminal
Event Log
Download pre-built shared indexes: Reduce the indexing time and CPU load with pre-built JDK shared indexes // Always download // Download once // Don't show again // Configure... (a minute ago)
5:1 CRLF UTF-8 4 spaces

File Edit View Navigate Code Refactor Build Run Tools VCS Window Help ZooProjectStage3 - Manager.java

ZooProjectStage3 〉 src 〉 com 〉 Hagenbeck 〉 ZooApp

ZooApp

Project

ZooApp.java | Zoo.java | Manager.java

Run: ZooApp

```
"C:\Program Files\Java\jdk-17\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2021.2.2\lib\idea_rt.jar=50628:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2021.2.2\bin" -Dfile.

Constructing Zoo(Hamburg)

Zoo:Zoo(Hamburg)

Process finished with exit code 0
```

Run TODO Problems Build Terminal Event Log

All files are up-to-date (moments ago) 5:1 CRLF UTF-8 4 spaces

Some extra thought …

Extras: Google/Facebook/etc OOP Interview Questions

Exercise 1: Jukebox: Design a musical jukebox using object-oriented principles.

Exercise 2: Parking Lot: Design a parking lot using object-oriented principles.

Exercise 3: Online Book Reader: Design the data structures for an online book reader system.

Source: Cracking the Coding Interview 6th edition, by Gayle Laakmann McDowell

Constructors

- What does the output look like?

```
public static void main(String[] args) {
    Zoo zoo1;
    zoo1=new Zoo( location: "Hamburg");
    zoo1=new Zoo( location: "Munic");
    Zoo zoo2=zoo1;
    Zoo zoo3=new Zoo();
    System.out.println("\nZoo1:"+zoo1);
    System.out.println("Zoo2:"+zoo2);
    System.out.println("Zoo3:"+zoo3);
    zoo3.setLocation("Berlin");
    zoo1.setLocation("Berlin");
    System.out.println("\nZoo1:"+zoo1);
    System.out.println("Zoo2:"+zoo2);
    System.out.println("Zoo3:"+zoo3);
    zoo1=new Zoo( location: "SanDiego");
    System.out.println("\nZoo1:"+zoo1);
    System.out.println("Zoo2:"+zoo2);
    System.out.println("Zoo3:"+zoo3);
}
```

```
Constructing Zoo(Hamburg)
Constructing Zoo(Munic)
Constructing Zoo(Unknown)

Zoo1:Zoo(Munic)
Zoo2:Zoo(Munic)
Zoo3:Zoo(Unknown)

Zoo1:Zoo(Berlin)
Zoo2:Zoo(Berlin)
Zoo3:Zoo(Berlin)

Constructing Zoo(SanDiego)

Zoo1:Zoo(SanDiego)
Zoo2:Zoo(Berlin)
Zoo3:Zoo(Berlin)
```

ZooProjectStage4

Acknowledgement

- Slides based on material from
  - Bill Leahy's lecture slides
    - http://www.cc.gatech.edu/~bleahy/xjava/cs1311xjava05_poly.ppt
  - Maria Litvin's & Gary Litvin's book slides
    - http://skylit.com/javamethods/ppt/Ch10.ppt
  - Marty Stepp's lecture slides
    - http://www.cs.washington.edu/331/
  - Cracking the Coding Interview by Gayle Laakmann McDowell
  - And other lecturers delivering this module in the past ...

But I also contributed some stuff myself :-)