LECTURE 01
Introduction to OOP

ITCS123 Object Oriented Programming

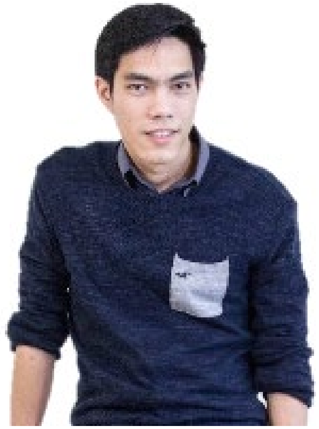
Dr. Siripen Pongpaichet
Dr. Petch Sajjacholapunt
Asst. Prof. Dr. Ananta Srisuphab

Semester 2/2023 10 January 2023

About this course

Course title	ITCS123 Object Oriented Programming
Number of Credits	3 (2-2-5) Credits (Lecture – Laboratory – Self-study)
Class Schedule	Thursday 1PM – 5PM#

Siripen Pongpaichet (Gr 1)
siripen.pon@mahidol.edu



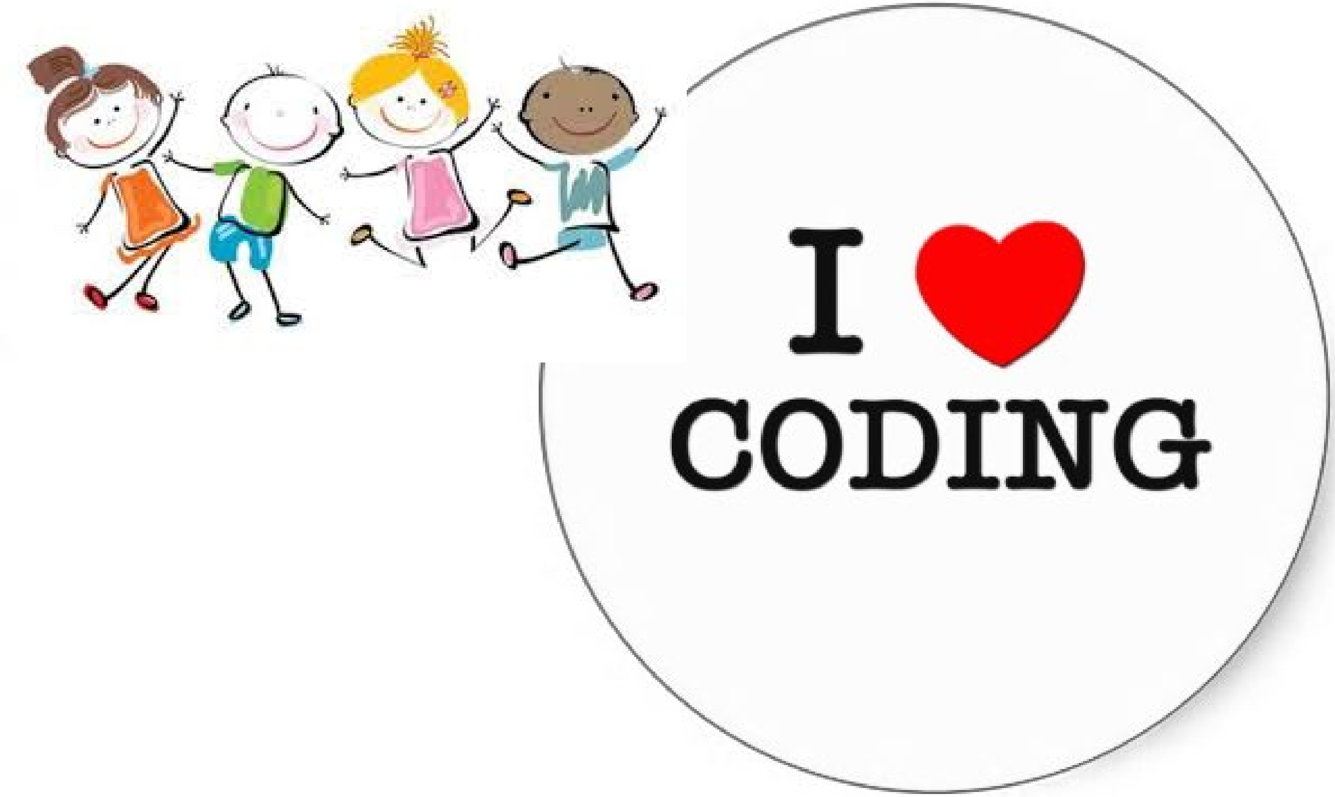
Petch Sajjachonlapunt (Gr 2)
petch.saj@mahidol.edu

Ananta Srisuphab (Gr 3)
ananta.sri@mahidol.edu

Course Learning Outcomes (CLOs)

- CLO1 **EXPLAIN** the concepts of Object-Oriented Programming as well as the purpose and usage principles of encapsulation, polymorphism, and inheritance.
- CLO2 **IDENTIFY** structures and outputs of a given source code written in Object-Oriented Programming paradigm.
- CLO3 **DESIGN** classes, objects, members of a class (e.g., attributes, methods, and data types) and the relationships among them needed for a specific problem.
- CLO4 **DEVELOP** application programs that appropriately use Object-Oriented Programming concepts and practices (e.g., classes, interfaces, access control identifiers, and error exception handling) to solve a given problem.
- CLO5 **IMPLEMENT** Object-Oriented Programs to solve common computer science problems (e.g., recursion, sorting, and searching).
- CLO6 Demonstrate awareness of **ethical responsibilities** in ICT-related disciplines

In fact,… My Course Learning Outcomes



PASSION

LOVE

LIKE

KINDA LIKE

DON'T HATE

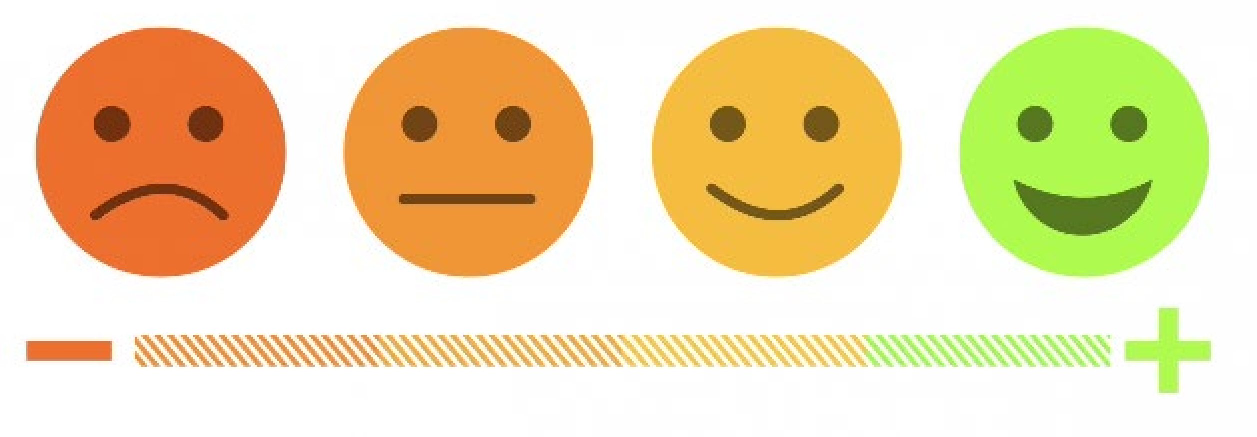
Can we all agree on these topics?

- **Ask Ask Ask away!** There is no such thing as a dumb question.
  - But if you don’t ask, you may remain dumb.
- **Always do your own work.** Using other people's work for your own credit is *stealing*.
- **Protect your own work** by writing your name on each file and putting many comments. DO NOT share your work with others. (^_^)
- **Help you friends** to become as good as or better than you.
  - If you let your friends copy your source code or dictate your friends to write code, they will learn nothing and eventually fail.

Violations of academic integrity will be reported to the Office of Academic Administration for review and often results in severe penalties such as a failing grade (F), permanent notation on transcripts, or worse.

Feedback

- If you think we can make the course better, feel free to give us constructive feedback anytime.



- Note: non-constructive complaints like “your projects are damn hard!” or “this course sucks!” will be ignored.

Assessment – Grading Criteria

Exam (Practical)	**70%**
Midterm	(35%)
Final	(35%)
Activities	**30%**
In-class labs	(10%)
Quizzes	(20%)
Projects	(0%)

5%

Challenge Assignment (3%)

Competition/ online learning (2%)

How to get a GOOD GRADE "A"

- **Exams**
  - Practical Exams. So your need to practice, practice, and practice.
  - You must take both midterm and final exams to pass this course.
- **Labs**
  - Weekly labs, the second half of the lecture.
  - Lab assignments (except Challenge Bonus) must be graded by a TA before 6 PM.
    - ***prepare to answer a few questions about your code :-P*** *Any late submissions will be ignored*
  - (optional) Challenge bonus: Go the extra mile and succeed.
    - *You can either submit within the lab hours of this week or the next lab hours.*
  - Source files (.java) must be uploaded on MyCourses before 6 PM.
    - We will not grade the submitted lab assignments, but they can serve as evidence to prove that you actually have finished the labs on-time.

How to get a GOOD GRADE “A” (Cont.)

- **Quizzes**
  - quizzes in the class.
  - please keep up with the lecture and lab assignments each week.
  - Try to work on the given project even there is no score.

Class Management

Time	Onsite	Online (if any)
1 – 3 PM	Onsite Lecture	Online Lecture
3 – 4 PM	Onsite Lab	Online Lab (Gr X)
4 – 5 PM	• Call TA for grading • Upload code on MyCourses	• Sign-up to submit your code via the provided Excel sheet, and a TA will contact you to grade your work • Upload code on MyCourses
5 – 6 PM	Extra hours for any additional session for students	

Preparation Make your system ready !!

- Compiler and IDE(integrated development environment)

Eclipse IDE

[https://www.eclipse.org/downloads/]

Note that: In this class, I will use Eclipse to demonstrate
and grade the examination.
However, you may use other IDE if you will.

Notepad++
(IDE)

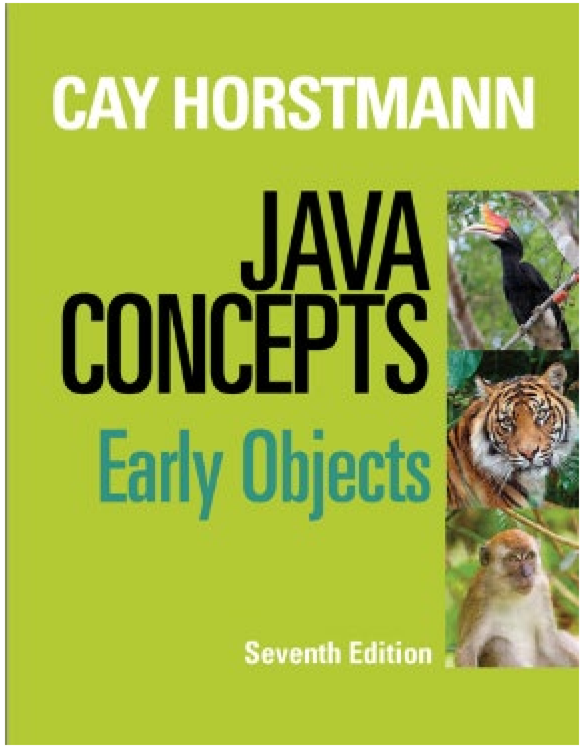
Visual Studio Code
(IDE)

Java JDK v.11

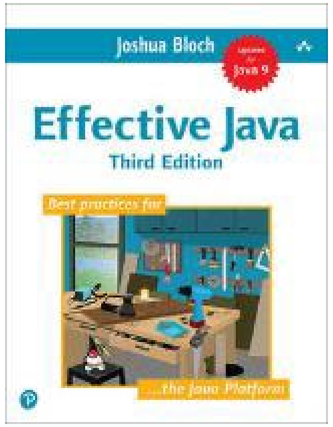
[http://www.oracle.com/technetwork/java/javase/downloads/index.html]

Learning Resources

- Textbook



Reference of this class



Recommended Read

- Recommended Online Resources

sololearn

Programming Methodology (https://see.stanford.edu/Course/CS106A) from Stanford: Overall this is a great class for a beginner programming learner

What will you learn in the following weeks (Tentatively)

Week1	Lecture Slides (Topics)	Note
1	Introduction to Object-Oriented Concepts	
2	Fundamental Data Types & Decisions & Loops	
3	Introduction to Objects and Classes	**Quiz 1**
4	Arrays/Array Lists	
5	Designing Classes	
6	Inheritance	**Quiz 2**
7	Polymorphism & Generic Type	
8	Midterm Examination Review	**Quiz 3**
9	Midterm Exam	-
10	Generic Type	
11	Interface & Java Collection (List, Set, Map)	**Quiz 4**
12	Input/Output and Exception Handling	
13	File Management & Regular Expression	**Quiz 5**
14	Recursion	
15	Object-Oriented Design	**Quiz 6**
16	Final Examination Review + Mock Final Exam	
17	Final Exam	-

Self Check

- When I have any issues, how to contact Ajarn ?
- When do I have to submit the lab assignment for grading?
- Which channels I can use to submit the lab assignment for grading?
- What should I do if I cannot do the lab assignment?
- When does the lab submission on MyCoures close?
- Is it okay to copy other’s code?
- Is it okay to help a friend by giving his/her your own code?
- Can I use other editors besides Eclipse to write my program?

Lecture1: Learning Objectives

After the end of this class, students

- Can **explain** a basic concept of OOP
- Can **setup** and **install** Java & Eclipse IDE
- Can **implement** a simple Java program
- Can **explain** some basic Java's syntax

What is **O**bject **O**riented **P**rogramming?

Object **O**riented **P**rogramming is a kind of

programming paradigm based on the concept of “Object”

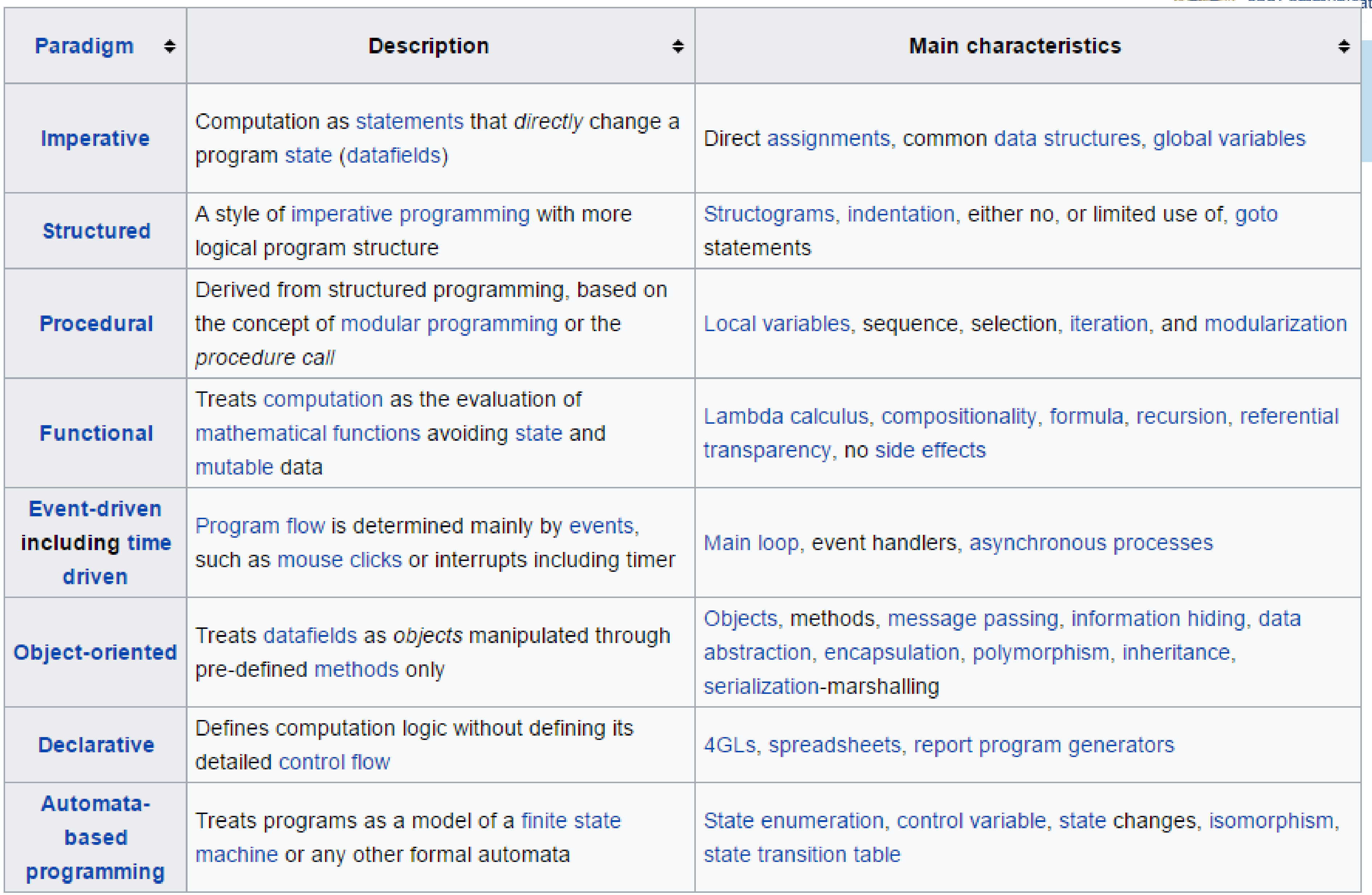
Programing Paradigm

- **Imperative paradigm** : e.g. C, C++, Java , etc...
- **Object-Oriented paradigm** : e.g. Java, C++, etc...
- **Declarative paradigm**: e.g. SQL, CSS, etc..
- **Functional paradigm** : e.g. Clojure, F#, etc…

Notice that one programming language
can be categorized to more than one paradigm.

Definition

Paradigm is a framework that guides the way we do things.

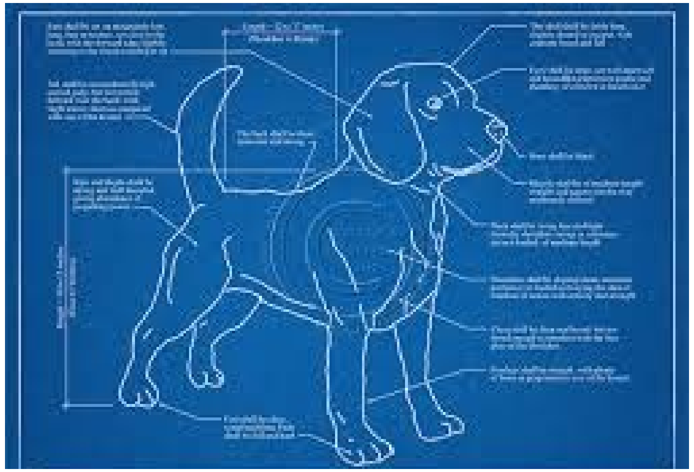


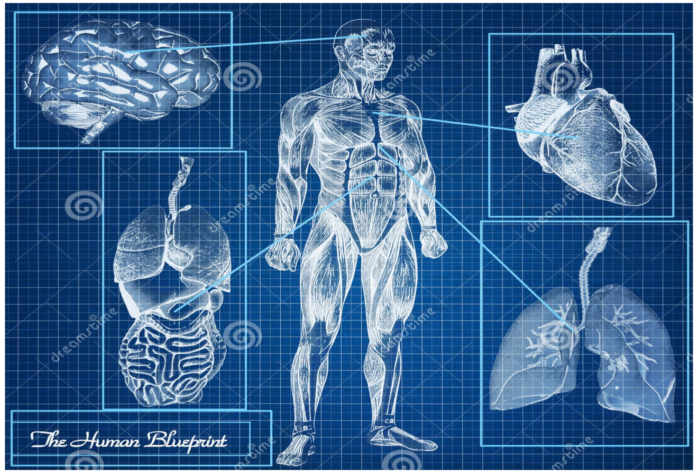
Paradigm	Description	Main characteristics
Imperative	Computation as statements that *directly* change a program state (datafields)	Direct assignments, common data structures, global variables
Structured	A style of imperative programming with more logical program structure	Structograms, indentation, either no, or limited use of, goto statements
Procedural	Derived from structured programming, based on the concept of modular programming or the *procedure call*	Local variables, sequence, selection, iteration, and modularization
Functional	Treats computation as the evaluation of mathematical functions avoiding state and mutable data	Lambda calculus, compositionality, formula, recursion, referential transparency, no side effects
Event-driven including time driven	Program flow is determined mainly by events, such as mouse clicks or interrupts including timer	Main loop, event handlers, asynchronous processes
Object-oriented	Treats datafields as *objects* manipulated through pre-defined methods only	Objects, methods, message passing, information hiding, data abstraction, encapsulation, polymorphism, inheritance, serialization-marshalling
Declarative	Defines computation logic without defining its detailed control flow	4GLs, spreadsheets, report program generators
Automata-based programming	Treats programs as a model of a finite state machine or any other formal automata	State enumeration, control variable, state changes, isomorphism, state transition table

https://en.wikipedia.org/wiki/Comparison_of_programming_paradigms

What is an OBJECT?

- **OOP** is a concept where things are considered as Objects..
- Each Object has it own Attributes and Behavior





Any thing can be an object depends on what level would you consider…

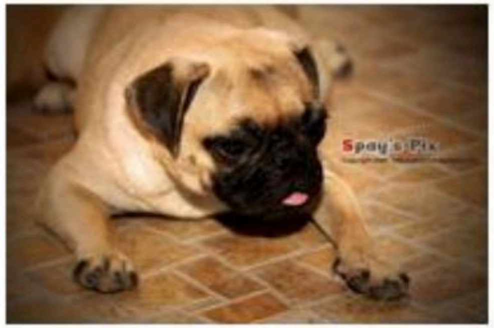
If we consider each DOG as an object

Attribute (same)

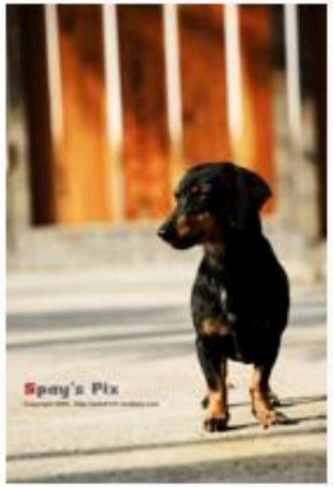
- 2 eyes
- 4 legs
- 1 mouth

Behavior (same)

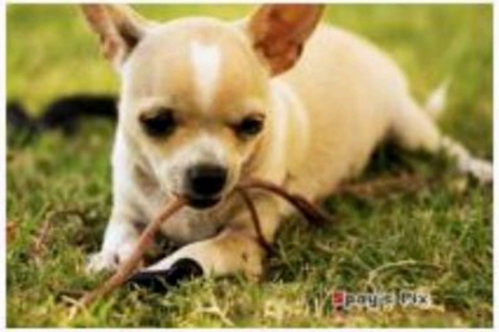
- eat
- run
- bark



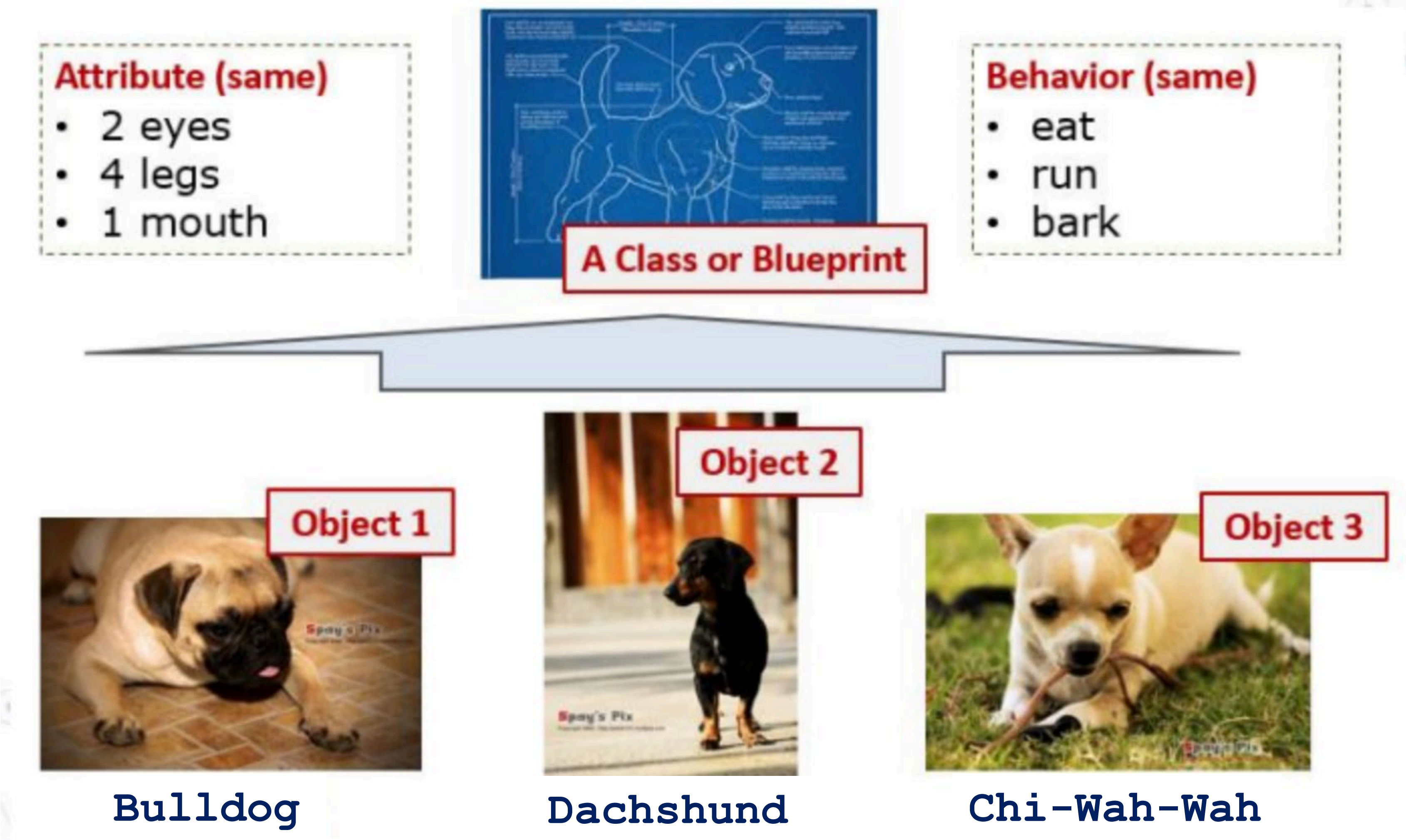
Bulldog



Dachshund



Chi-Wah-Wah



Attribute (same)

- 2 eyes
- 4 legs
- 1 mouth

A Class or Blueprint

Behavior (same)

- eat
- run
- bark

Object 1

Bulldog

Object 2

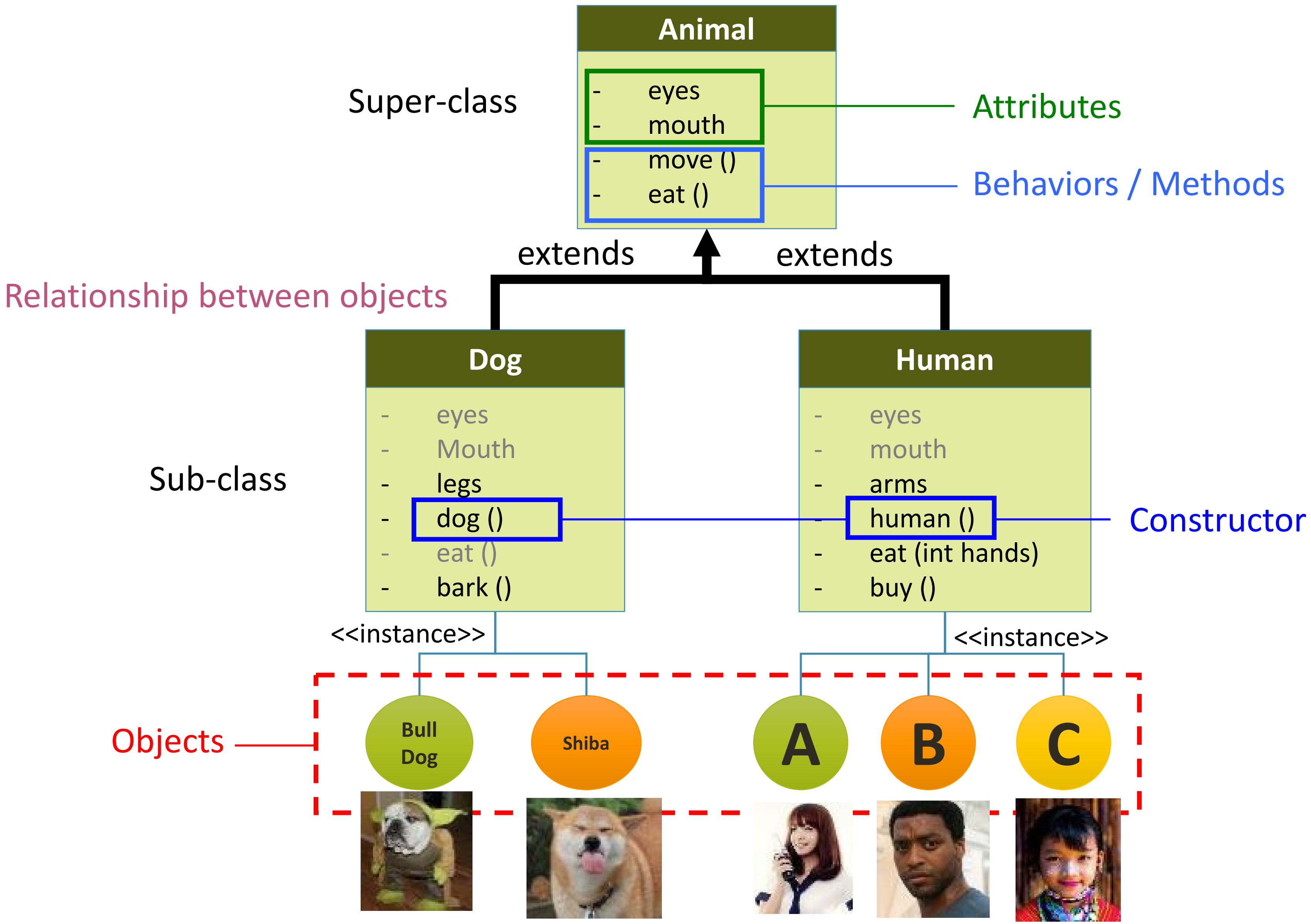
Dachshund

Object 3

Chi-Wah-Wah

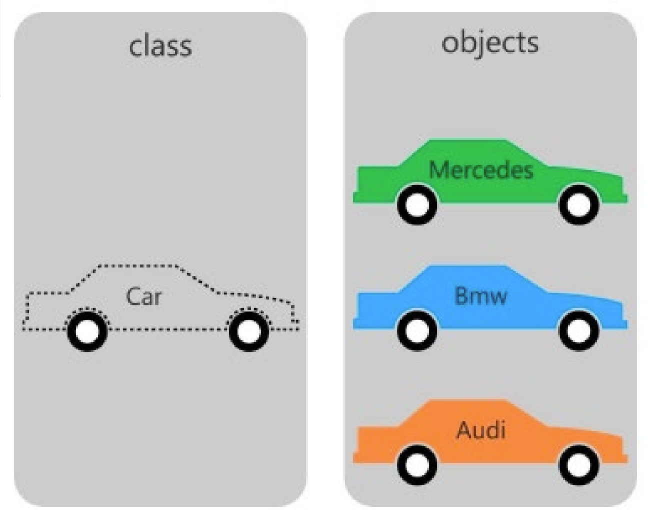
In real-life you can easily find that many individual objects of the same kind such as most of dogs have same physical attributes (4 legs). So the concept of a blueprint (class) is used in world of objects.

Any thing can be an object depends on what level would you consider…



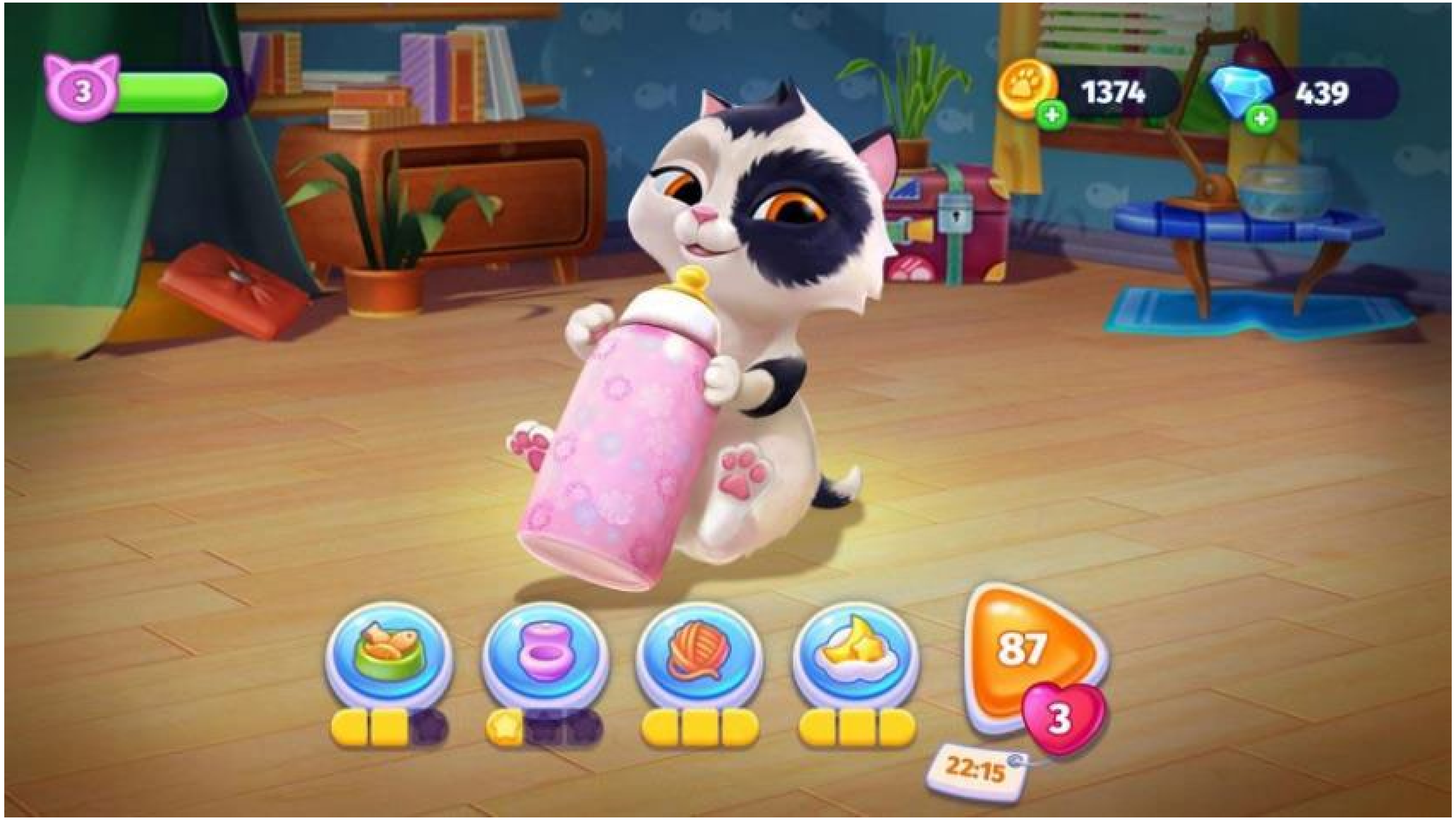
Another Example

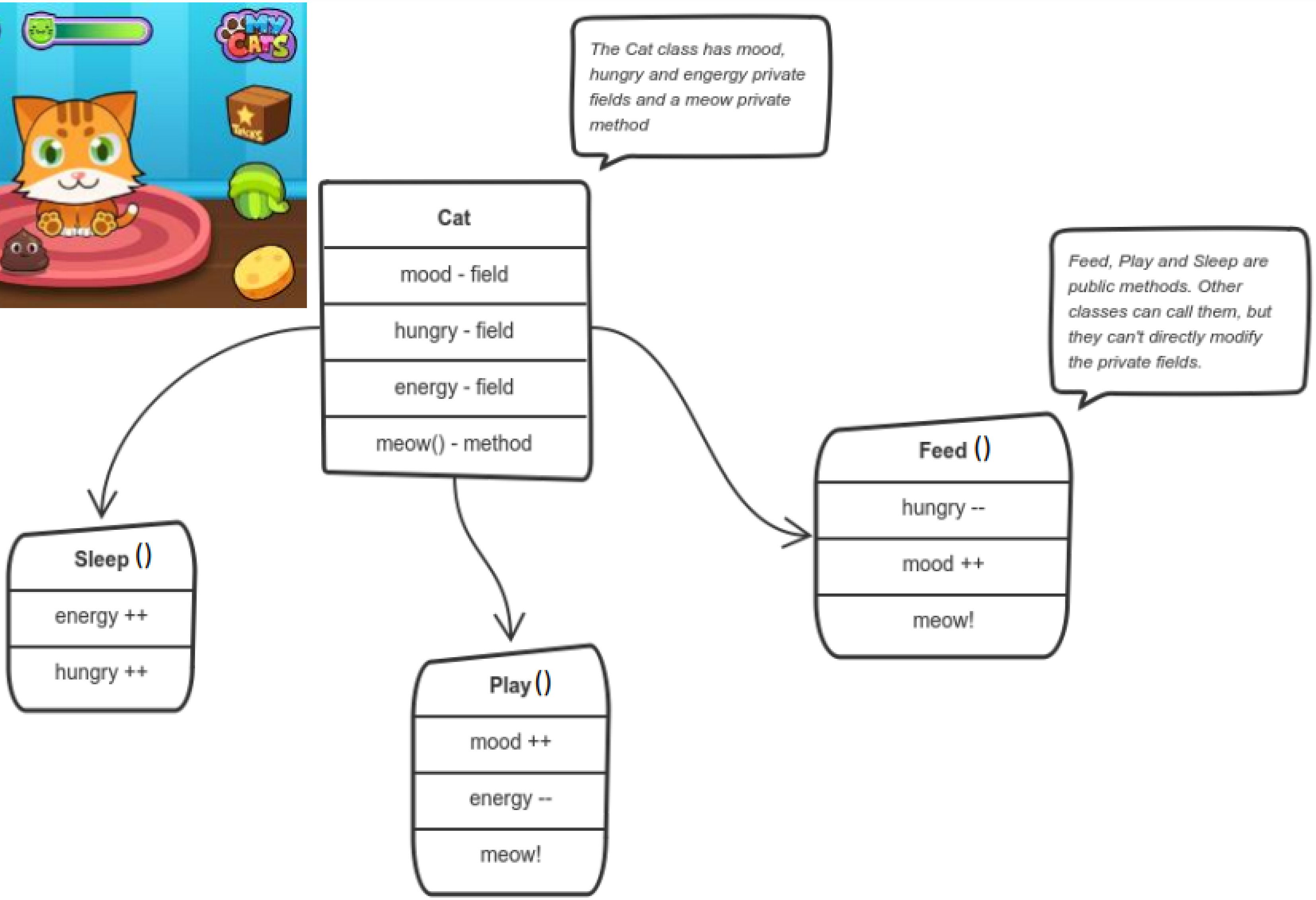
Object	Attributes/ Properties	Behavior/ Methods
	car.name = Fiat	car.start()
	car.model = 500	car.drive()
	car.weight = 850kg	car.brake()
	car.color = white	car.stop()



How about these objects?

How about in the Game?





The Cat class has mood, hungry and engergy private fields and a meow private method

Cat

mood - field

hungry - field

energy - field

meow() - method

Feed, Play and Sleep are public methods. Other classes can call them, but they can't directly modify the private fields.

Sleep ()

energy ++

hungry ++

Play ()

mood ++

energy --

meow!

Feed ()

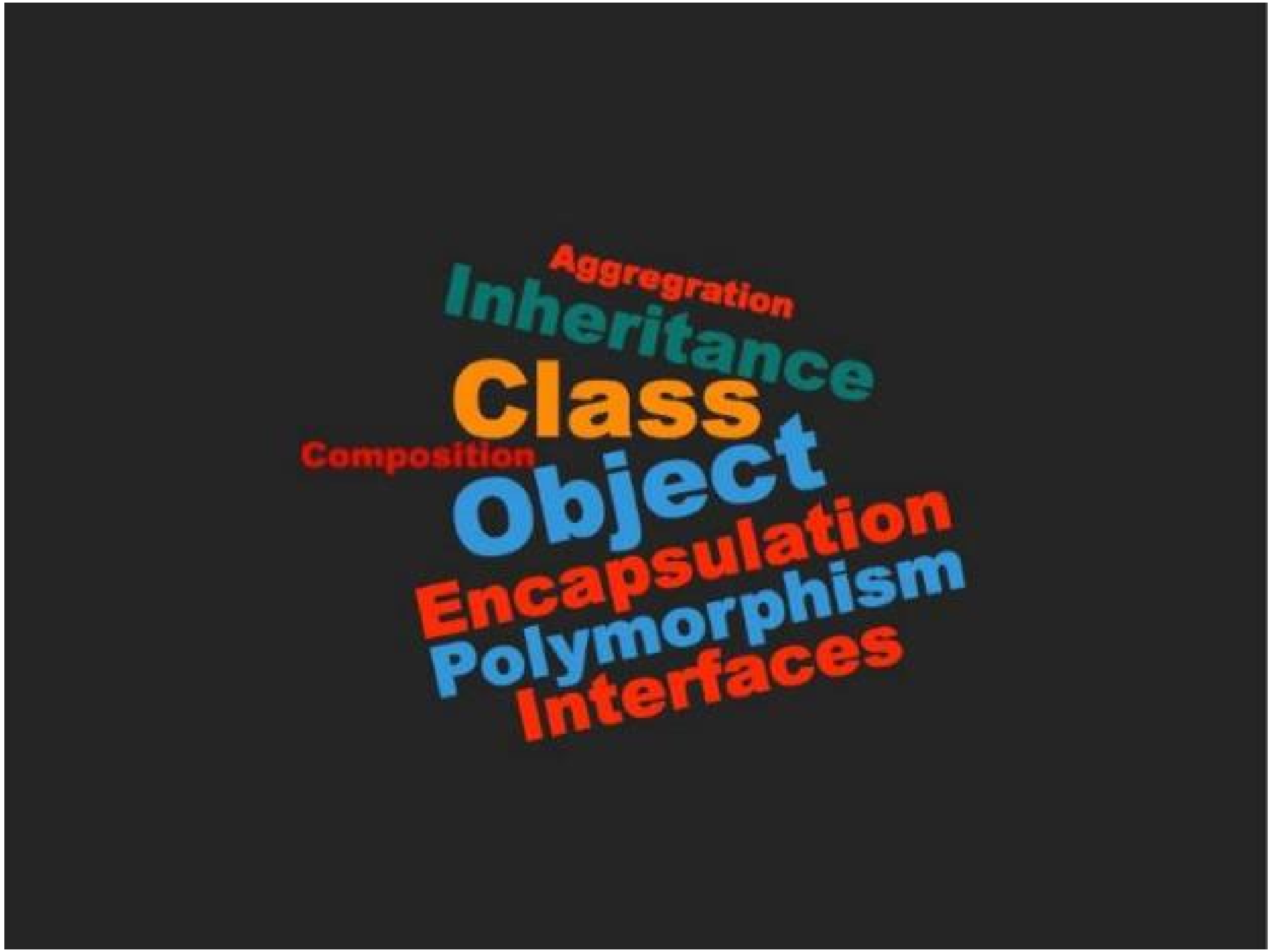
hungry --

mood ++

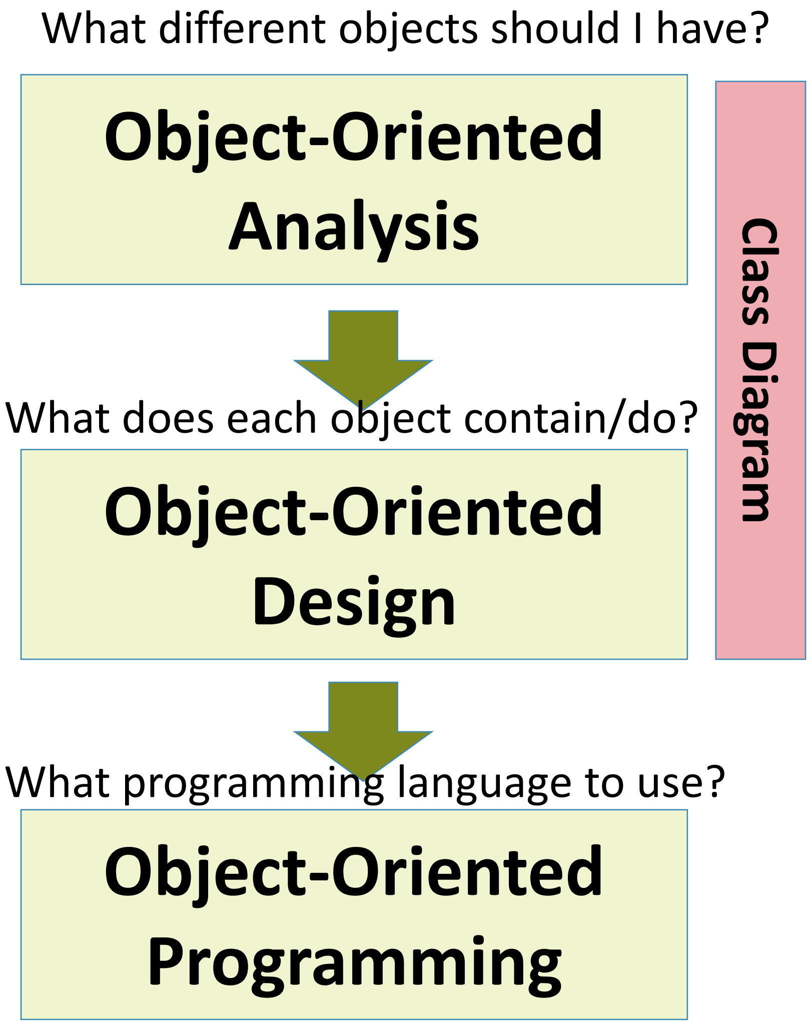
meow!

Ref: https://medium.freecodecamp.org/object-oriented-programming-concepts-21bb035f7260

So…
what are so special about the OOP paradigm



How to start working with OOP Concepts



What different objects should I have?

Object-Oriented Analysis

What does each object contain/do?

Object-Oriented Design

What programming language to use?

Object-Oriented Programming

Class Diagram

Identifying requirements and developing software specifications.

Generating conceptual model

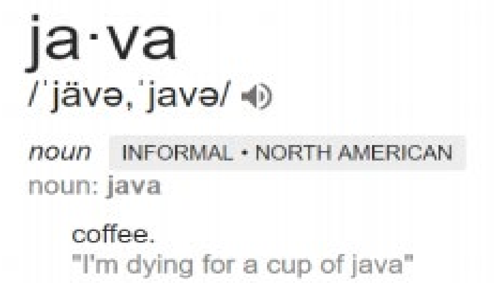
Implementing software based on the OOP design.

Programing Language for OOP

- There are many languages that used the Object Oriented Programming paradigm such as C++, C#, Python, Ruby, and JAVA.

- In this course, the **JAVA** programming language is used

Why JAVA?



ja·va

/ˈjävə,ˈjavə/

noun INFORMAL • NORTH AMERICAN

noun: **java**

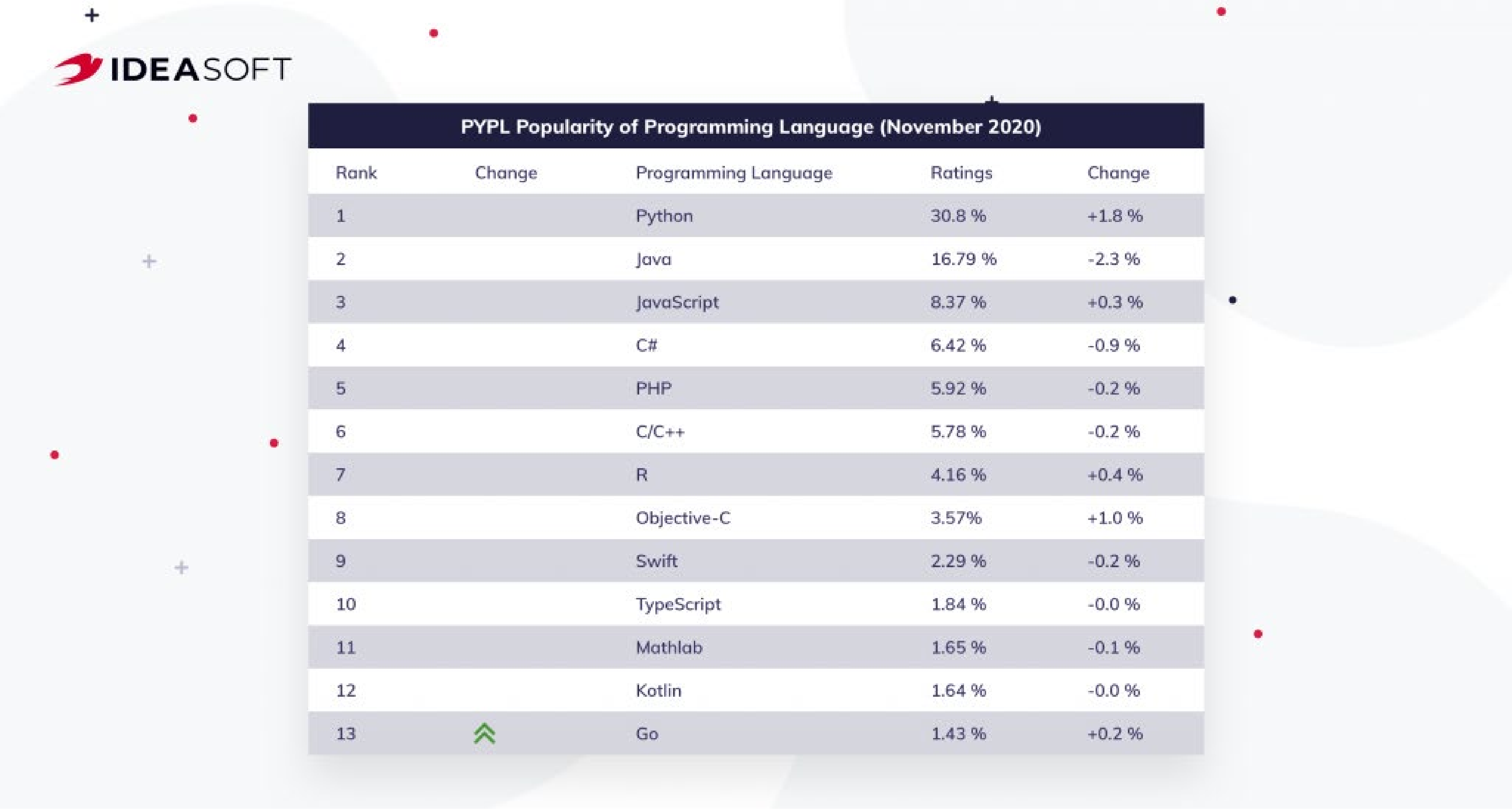
coffee.

"I'm dying for a cup of java"

Most Popular Programming Languages 1965 - 2019

https://www.youtube.com/watch?v=Og847HVwRSI

Programming Community Index

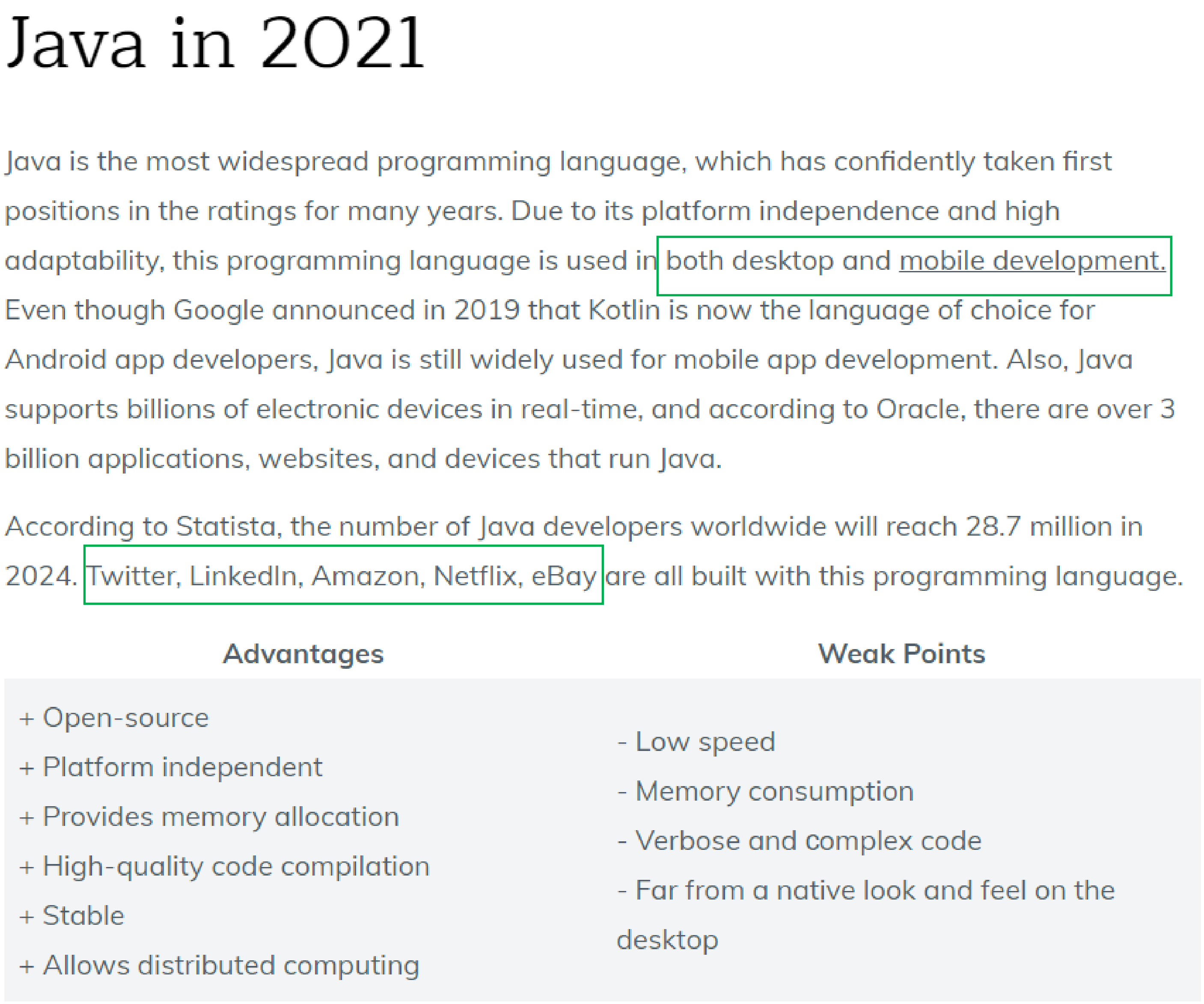


PYPL Popularity of Programming Language (November 2020)

Rank	Change	Programming Language	Ratings	Change
1		Python	30.8 %	+1.8 %
2		Java	16.79 %	-2.3 %
3		JavaScript	8.37 %	+0.3 %
4		C#	6.42 %	-0.9 %
5		PHP	5.92 %	-0.2 %
6		C/C++	5.78 %	-0.2 %
7		R	4.16 %	+0.4 %
8		Objective-C	3.57%	+1.0 %
9		Swift	2.29 %	-0.2 %
10		TypeScript	1.84 %	-0.0 %
11		Mathlab	1.65 %	-0.1 %
12		Kotlin	1.64 %	-0.0 %
13	⇑	Go	1.43 %	+0.2 %

https://ideasoft.io/blog/the-10-most-in-demand-programming-languages-in-2021/

Pros & Cons



Java in 2021

Java is the most widespread programming language, which has confidently taken first positions in the ratings for many years. Due to its platform independence and high adaptability, this programming language is used in both desktop and mobile development. Even though Google announced in 2019 that Kotlin is now the language of choice for Android app developers, Java is still widely used for mobile app development. Also, Java supports billions of electronic devices in real-time, and according to Oracle, there are over 3 billion applications, websites, and devices that run Java.

According to Statista, the number of Java developers worldwide will reach 28.7 million in 2024. Twitter, LinkedIn, Amazon, Netflix, eBay are all built with this programming language.

Advantages	Weak Points
+ Open-source	- Low speed
+ Platform independent	- Memory consumption
+ Provides memory allocation	- Verbose and complex code
+ High-quality code compilation	- Far from a native look and feel on the desktop
+ Stable	
+ Allows distributed computing	

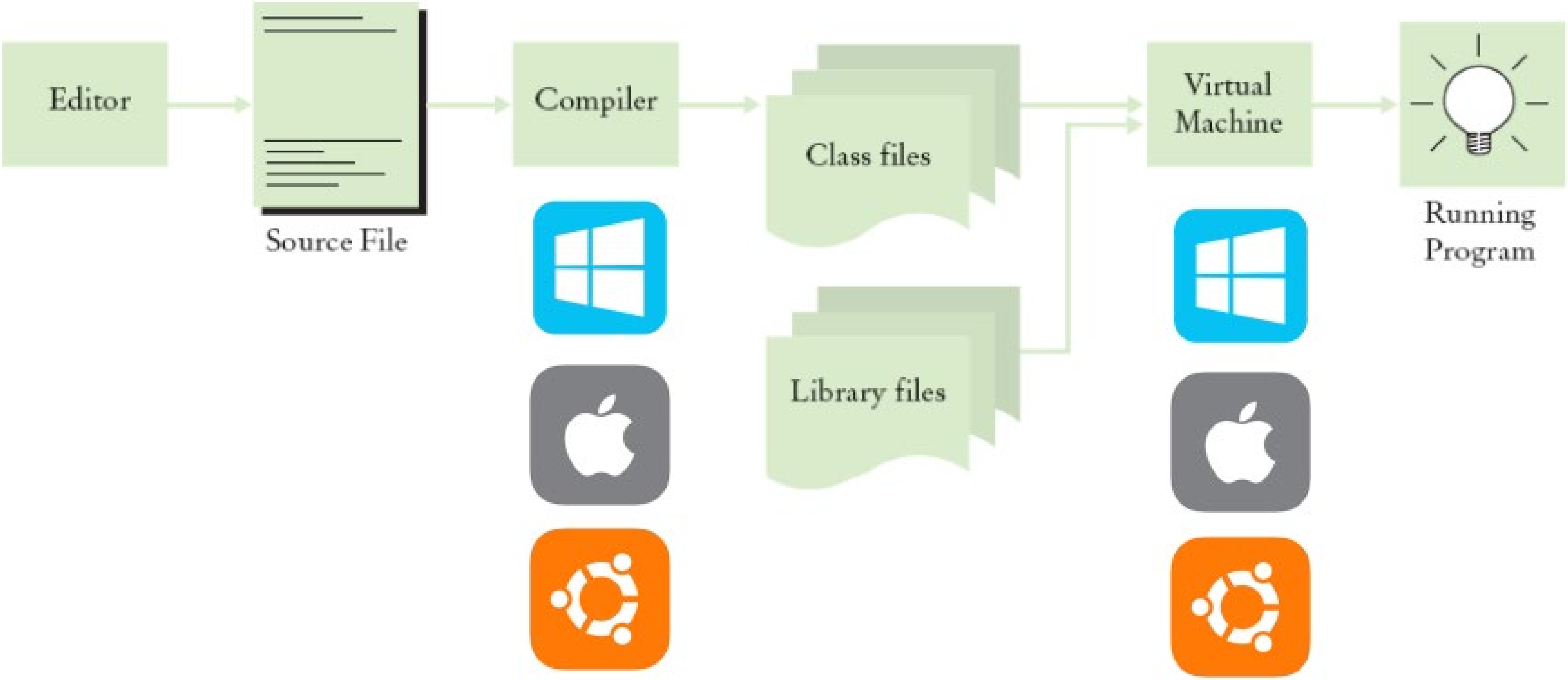
https://ideasoft.io/blog/the-10-most-in-demand-programming-languages-in-2021/

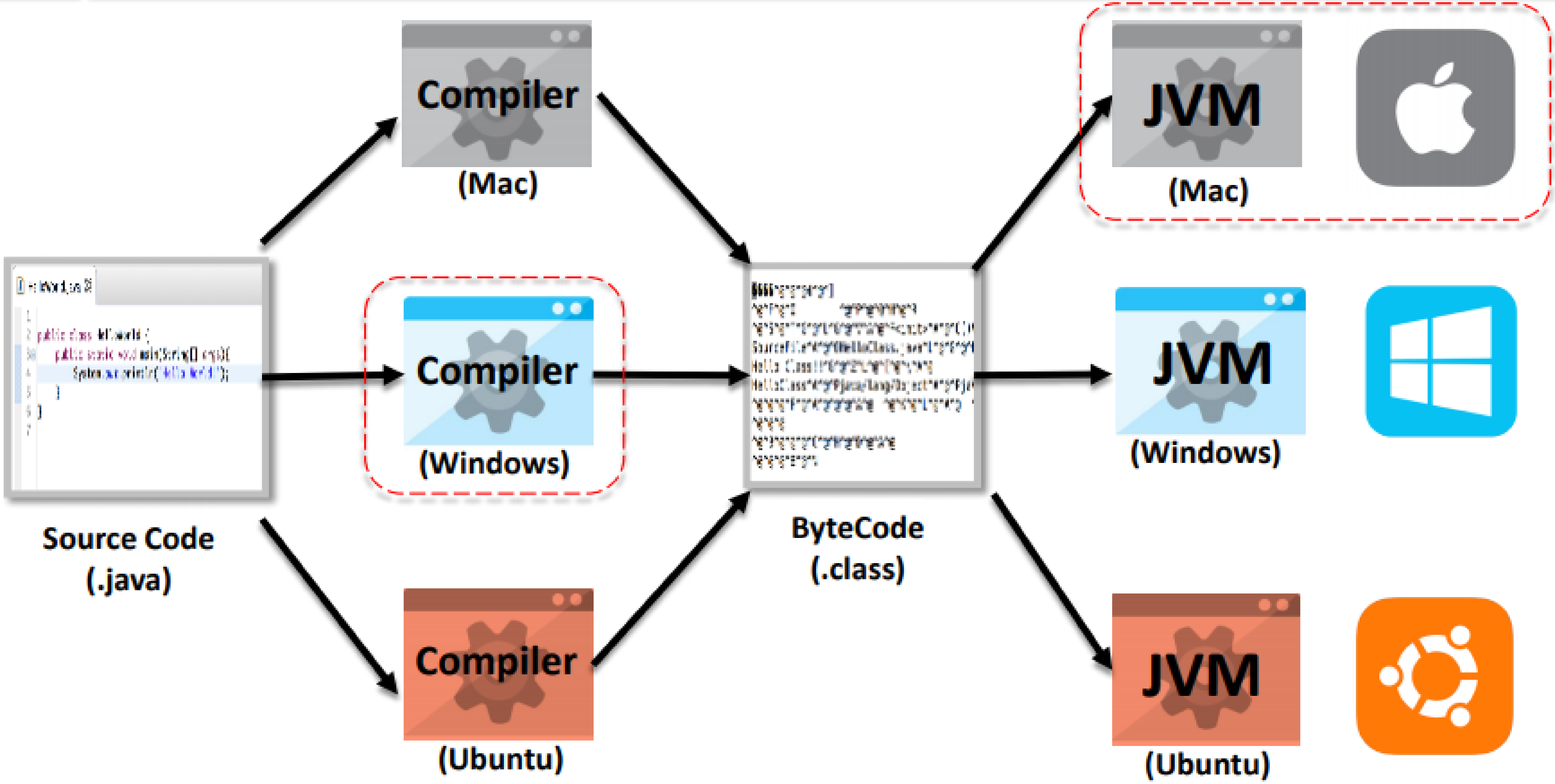
JAVA Programing Language

- **Portability**: the same java program can run, without change, on many different operating systems that have java virtual machine (JVM).

- **Robustness** (reliability): JVM can catch many kinds of beginners’ mistakes and report them accurately compared to other languages.

How JAVA work?



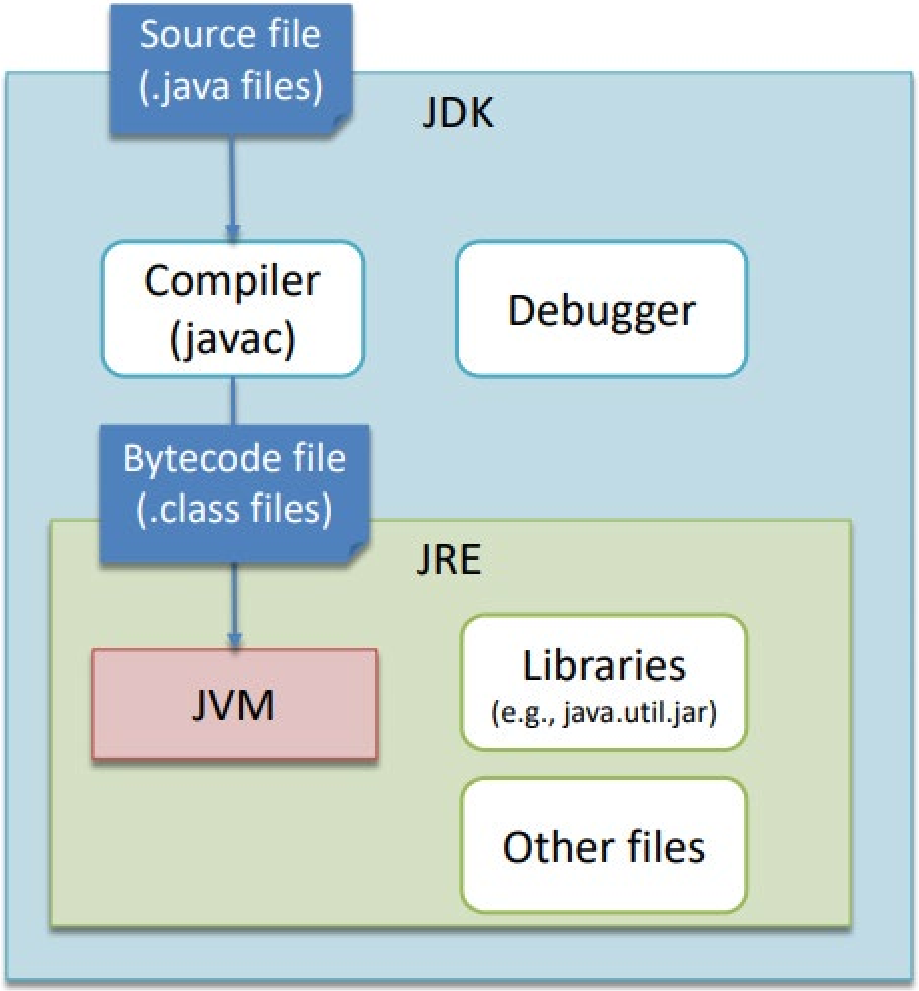


HelloWorld.java

```
public class HelloWorld {
    public static void main(String[] args){
        System.out.println("Hello World");
    }
}
```

Question: Can you COMPILE your source code in Windows and RUN on Mac OS?

JVM vs JRE vs JDK

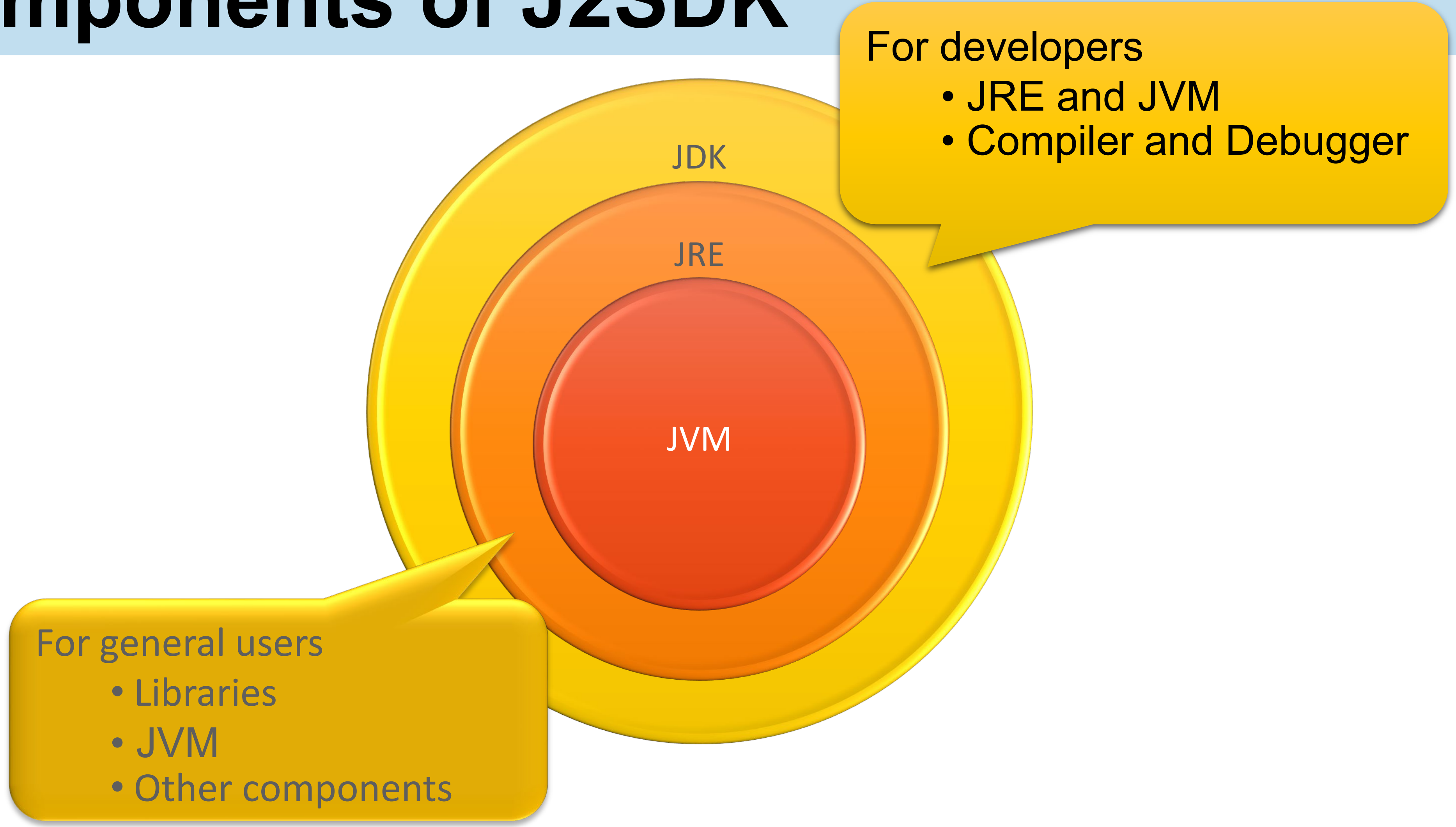


1. JVM: Java Virtual Machine
JVM interprets Java bytecode into native instructions for the platform it runs on.

2. JRE: Java Runtime Environment It is used to provide runtime environment. It contains set of libraries + other files that JVM uses at runtime

3. JDK: Java Development Kit
It contains JRE and development tools, typically including a Java compiler and debugger

Components of J2SDK



For developers

- JRE and JVM
- Compiler and Debugger

For general users

- Libraries
- JVM
- Other components

C vs. JAVA programming

- **C Programming**
  - Imperative
  - Manual Memory Management (bugs prone)
  - Good for low-level programming
    (dealing more with hardware)
  - Can be cross-platform but it’s not easy
- **JAVA Programming**
  - Object-oriented
  - Memory-Management (auto garbage collection)
  - Good for high-level or quick program
  - Cross-platform by design

Credit: Slide form C Minicourse Introduction to C

Download and Install Java & Eclipse

- Read and follow the additional document in MyCourses:
- **Exercise01 Setup & Installation.pdf**

Using Command Line

- **Listing the content of directories**
  - dir (DOS)
  - ls (Unix)
- **Changing Directory**
  - cd [path] (DOS/UNIX)
  - cd [..] (DOS/UNIX) **[..] change to parent directory

Testing JAVA on command line

- *Create* the program by typing the following code into a text editor.

- Save it to a file named **FirstJava.java**

```
public class FirstJava
{
        public static void main(String[] args)
        {
                System.out.println("Hello World! I love coding");
        }
}
```

C Program

```
#include <stdio.h>

int main(void) {
    printf("Hello World! I love coding\n");
    return 0;
}
```

Using Command Line

[JAVA command line]

- Checking java version
  - `java -version`
- Compilation
  - `javac [filename.java]`
- Running
  - `java [classname]`
  - `java -jar [jarfilename.jar]`

Compile and Run Java Program

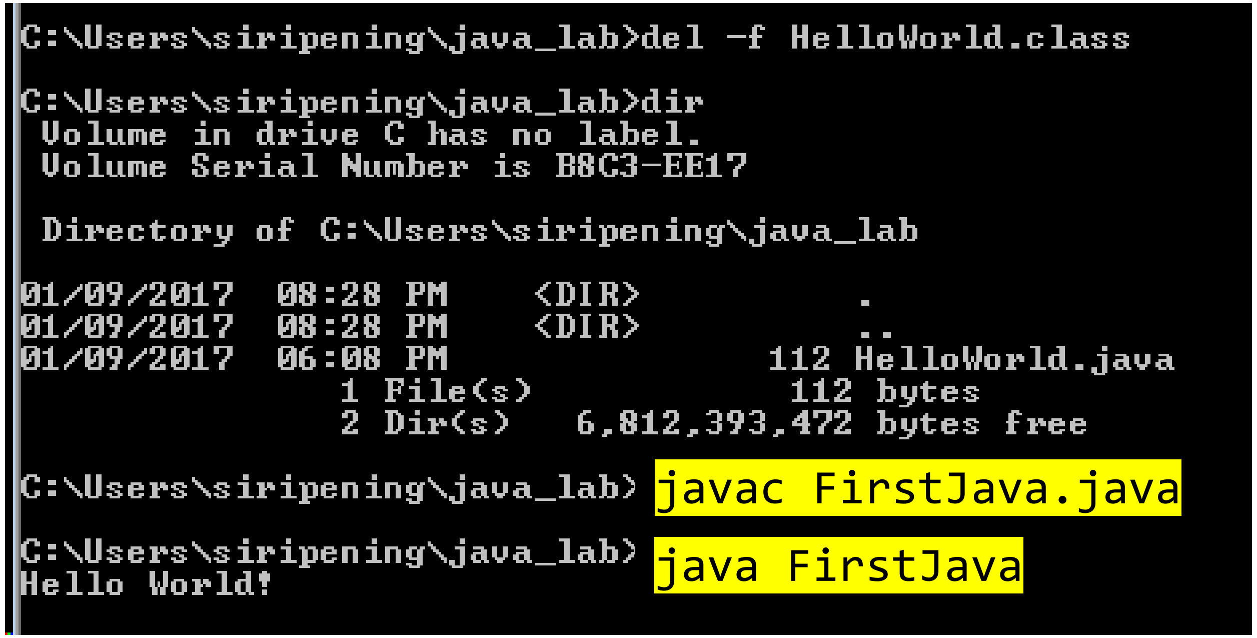
- □ Compile Java file to crate *.class

```
>> javac FirstJava.java
```

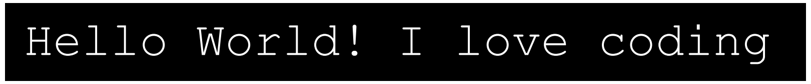
- □ Run Java Program

```
>> java FirstJava
```

CMD on Windows

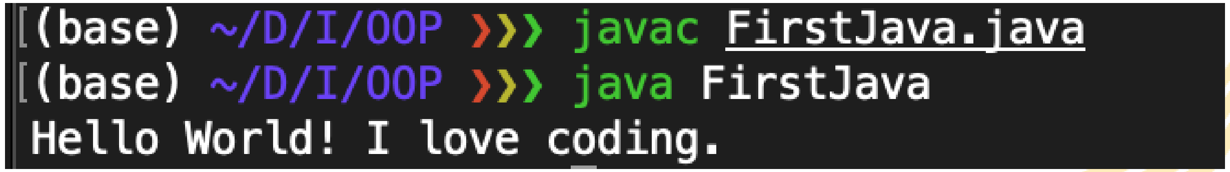


```
C:\Users\siripening\java_lab>del -f HelloWorld.class

C:\Users\siripening\java_lab>dir
 Volume in drive C has no label.
 Volume Serial Number is B8C3-EE17

 Directory of C:\Users\siripening\java_lab

01/09/2017  08:28 PM    <DIR>          .
01/09/2017  08:28 PM    <DIR>          ..
01/09/2017  06:08 PM               112 HelloWorld.java
               1 File(s)            112 bytes
               2 Dir(s)   6,812,393,472 bytes free

C:\Users\siripening\java_lab> javac FirstJava.java

C:\Users\siripening\java_lab> java FirstJava
Hello World!
```



```
Hello World! I love coding
```

Terminal on Mac



```
(base) ~/D/I/OOP ❯❯❯ javac FirstJava.java
(base) ~/D/I/OOP ❯❯❯ java FirstJava
Hello World! I love coding.
```

Creating .JAR File

- JAR stands for Java ARchive. It’s a file format based on the popular ZIP file format and is used for aggregating many files into one.
- Purposes
  - Portable Java Library
  - Portable Executable Program

Creating .JAR File on CMD

- After compile the java class(es) `[>> javac FirstJava.java]`
- Create a manifest file

```
>> echo Main-Class: FirstJava > manifest.txt
```

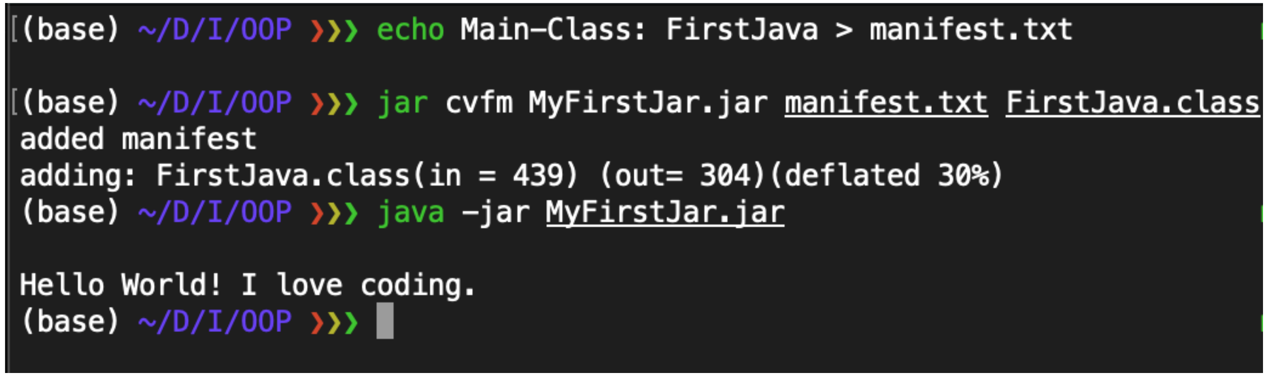
- Crate a jar file

```
>> jar cvfm MyFirstJar.jar manifest.txt FirstJava.class
```

- Test the jar file

```
>> java -jar MyFirstJar.jar
```

cvfm means "create a jar; show verbose output; specify the output jar file name; specify the manifest file name."

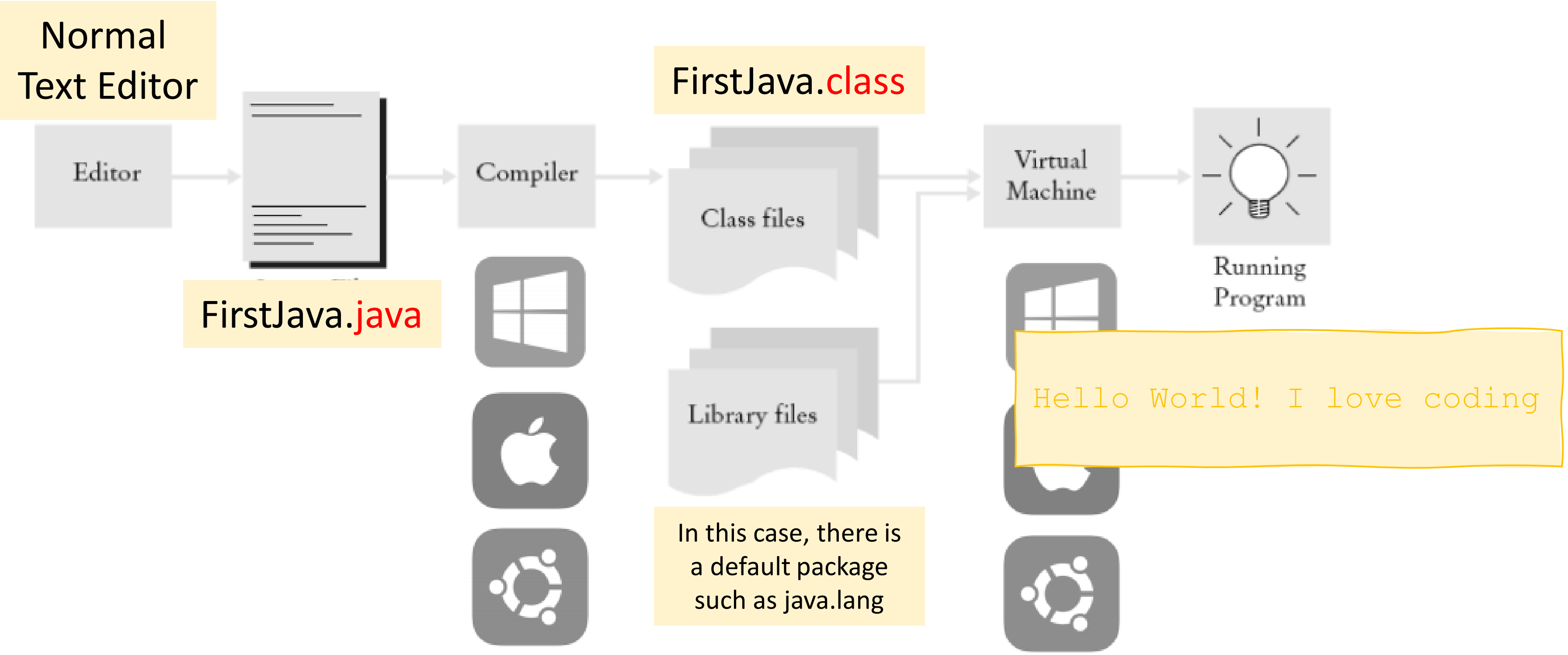


```
[(base) ~/D/I/OOP ❯❯❯ echo Main-Class: FirstJava > manifest.txt

[(base) ~/D/I/OOP ❯❯❯ jar cvfm MyFirstJar.jar manifest.txt FirstJava.class
added manifest
adding: FirstJava.class(in = 439) (out= 304)(deflated 30%)
(base) ~/D/I/OOP ❯❯❯ java -jar MyFirstJar.jar

Hello World! I love coding.
(base) ~/D/I/OOP ❯❯❯
```

Ref: http://www.skylit.com/javamethods/faqs/createjar.html

Recall: How Java Work?

- Scenario 1: Run java on Command Line (without IDE)



Normal
Text Editor

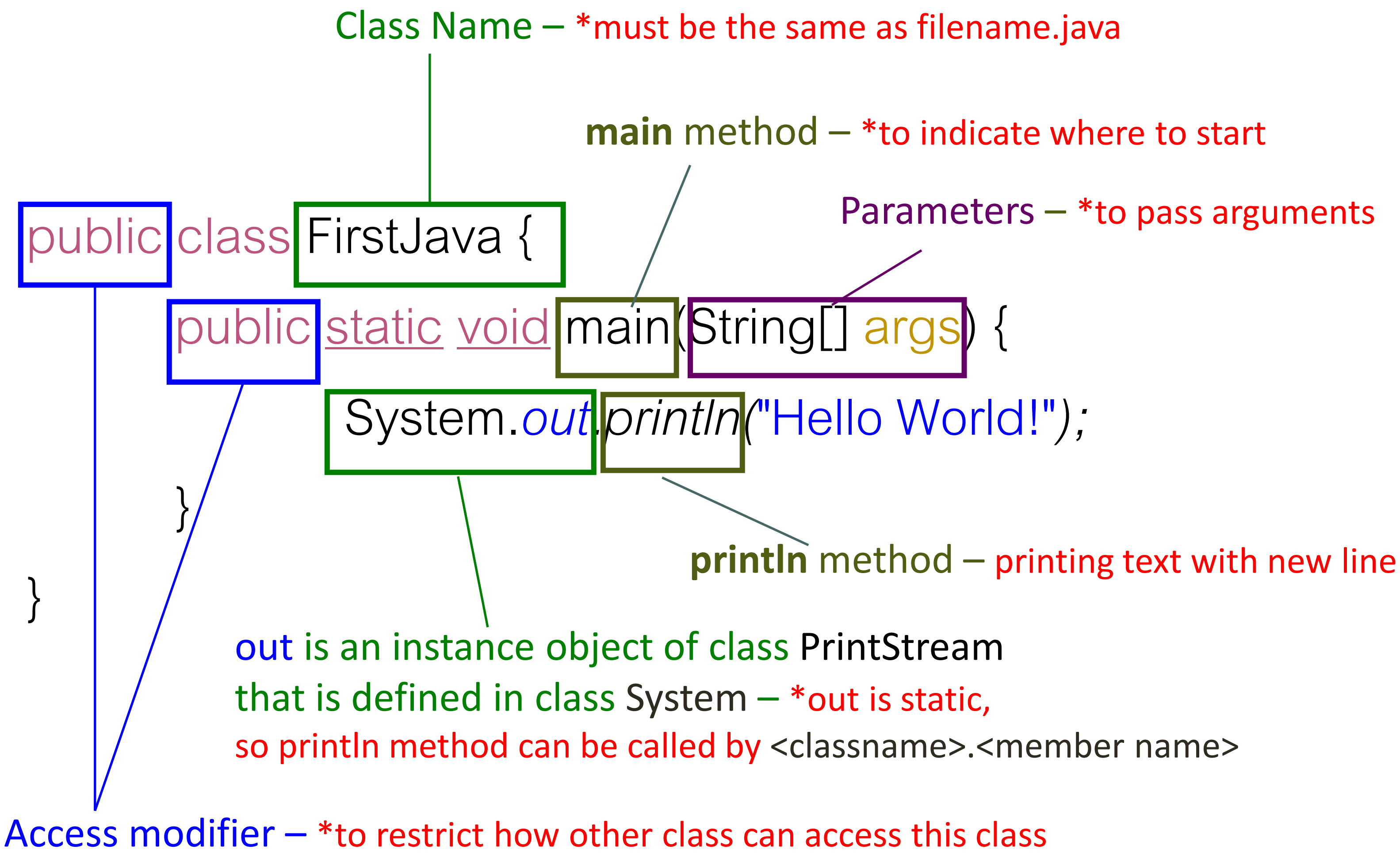
FirstJava.java

FirstJava.class

In this case, there is
a default package
such as java.lang

```
Hello World! I love coding
```

Understand a program (Cont.)



```
public class FirstJava {
        public static void main(String[] args) {
            System.out.println("Hello World!");
        }
}
```

Class Name – *must be the same as filename.java

main method – *to indicate where to start

Parameters – *to pass arguments

println method – printing text with new line

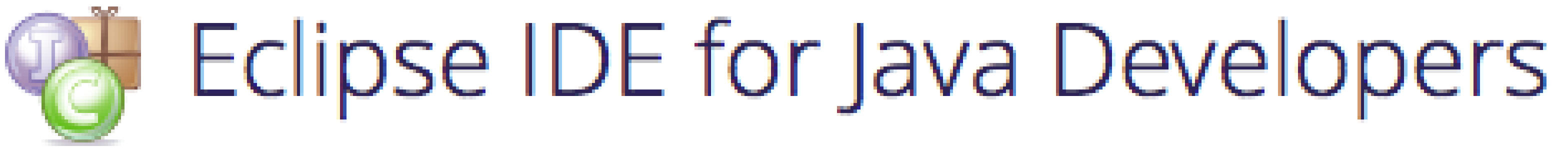
out is an instance object of class PrintStream
that is defined in class System – *out is static,
so println method can be called by <classname>.<member name>

Access modifier – *to restrict how other class can access this class

Eclipse Download and Installation

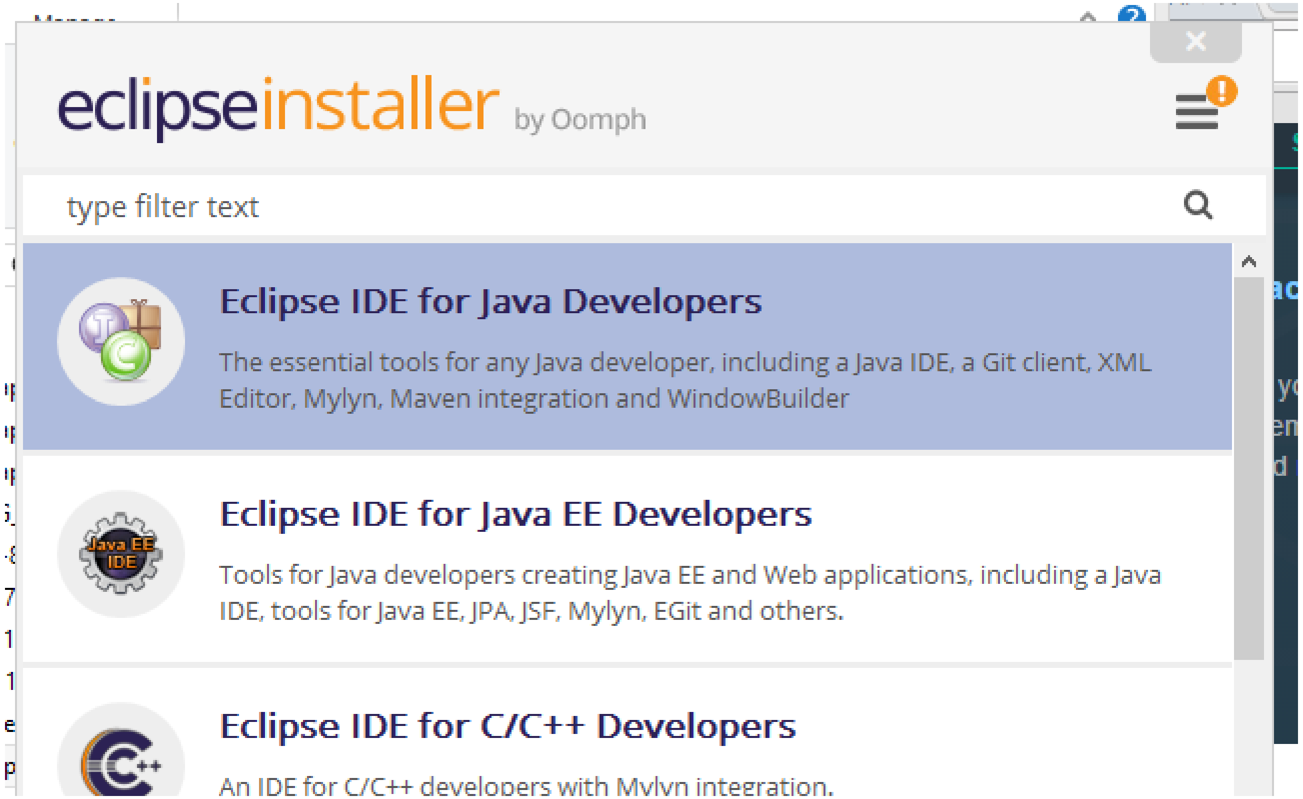
Downloading Eclipse

1. Go to website https://eclipse.org/downloads/
2. Select the **Download Packages > Eclipse IDE for Java Developers**
3. Download the appropriate version for your Operating system and machine



Eclipse IDE for Java Developers

Eclipse Installation



Eclipse IDE for Java Developers

The essential tools for any Java developer, including a Java IDE, a Git client, XML Editor, Mylyn, Maven integration and WindowBuilder

Eclipse IDE for Java EE Developers

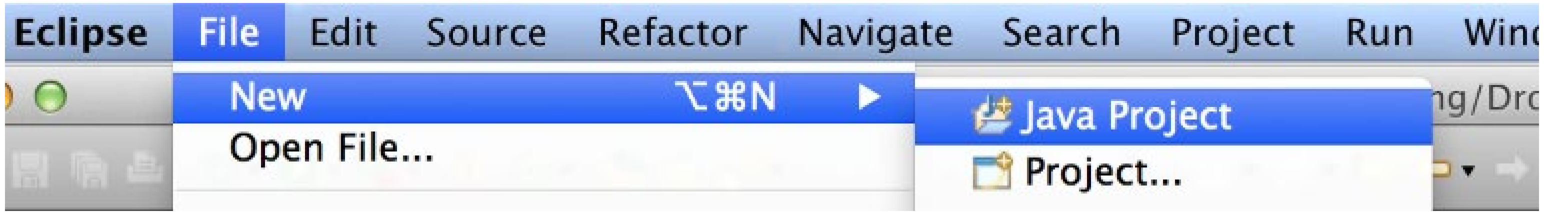
Tools for Java developers creating Java EE and Web applications, including a Java IDE, tools for Java EE, JPA, JSF, Mylyn, EGit and others.

Eclipse IDE for C/C++ Developers

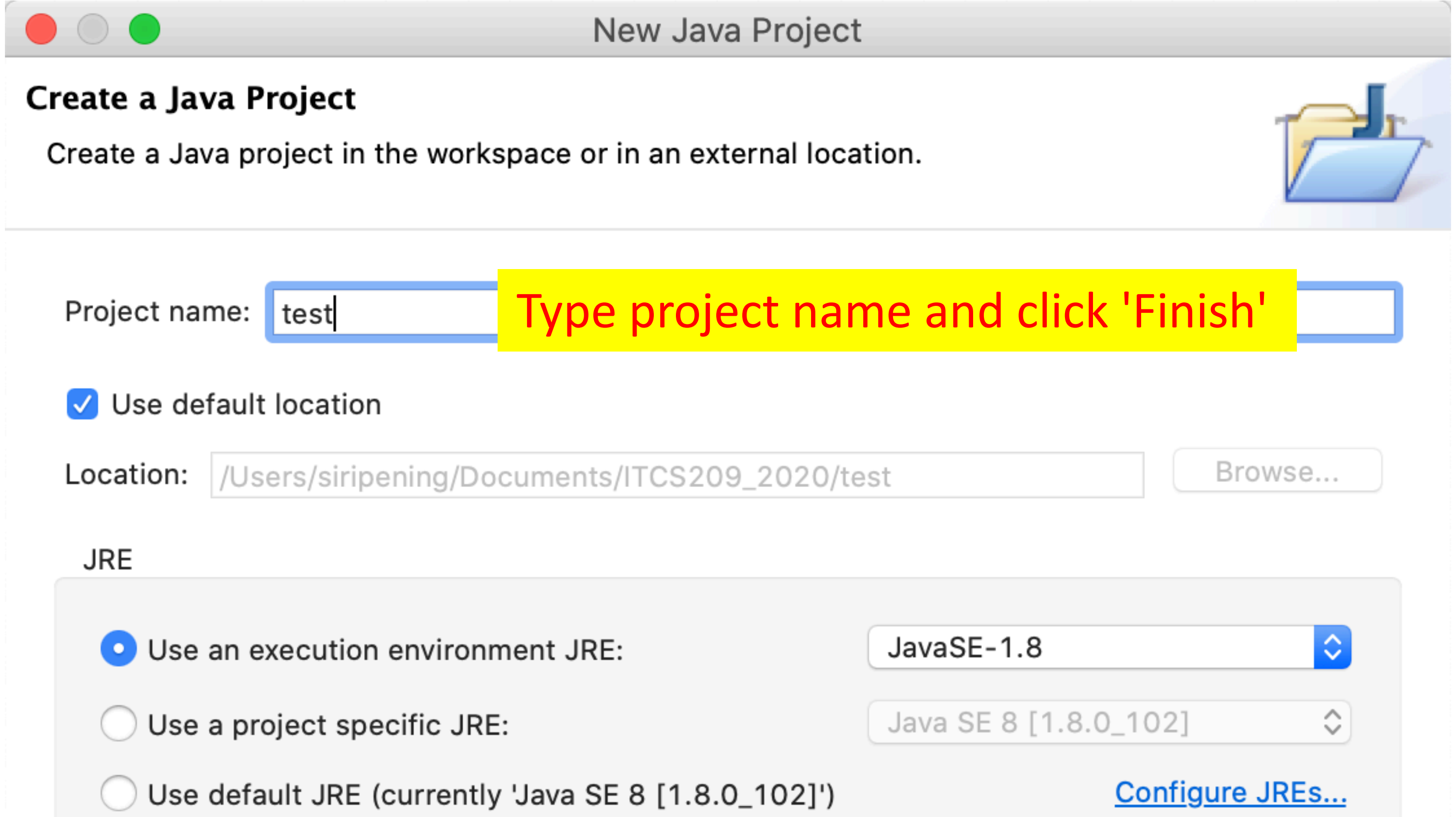
An IDE for C/C++ developers with Mylyn integration.

Create Project - Using Eclipse

1



2



Create a Java Project

Create a Java project in the workspace or in an external location.

Project name: test

Type project name and click 'Finish'

☑ Use default location

Location: /Users/siripening/Documents/ITCS209_2020/test

Browse...

JRE

● Use an execution environment JRE: JavaSE-1.8

○ Use a project specific JRE: Java SE 8 [1.8.0_102]

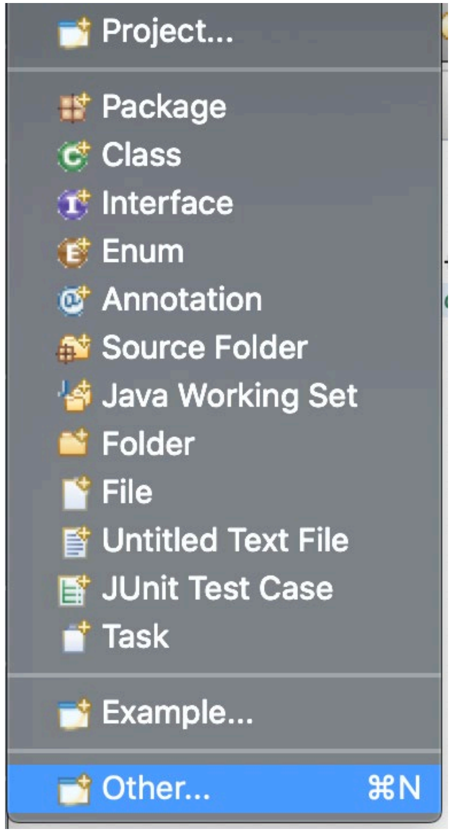
○ Use default JRE (currently 'Java SE 8 [1.8.0_102]')

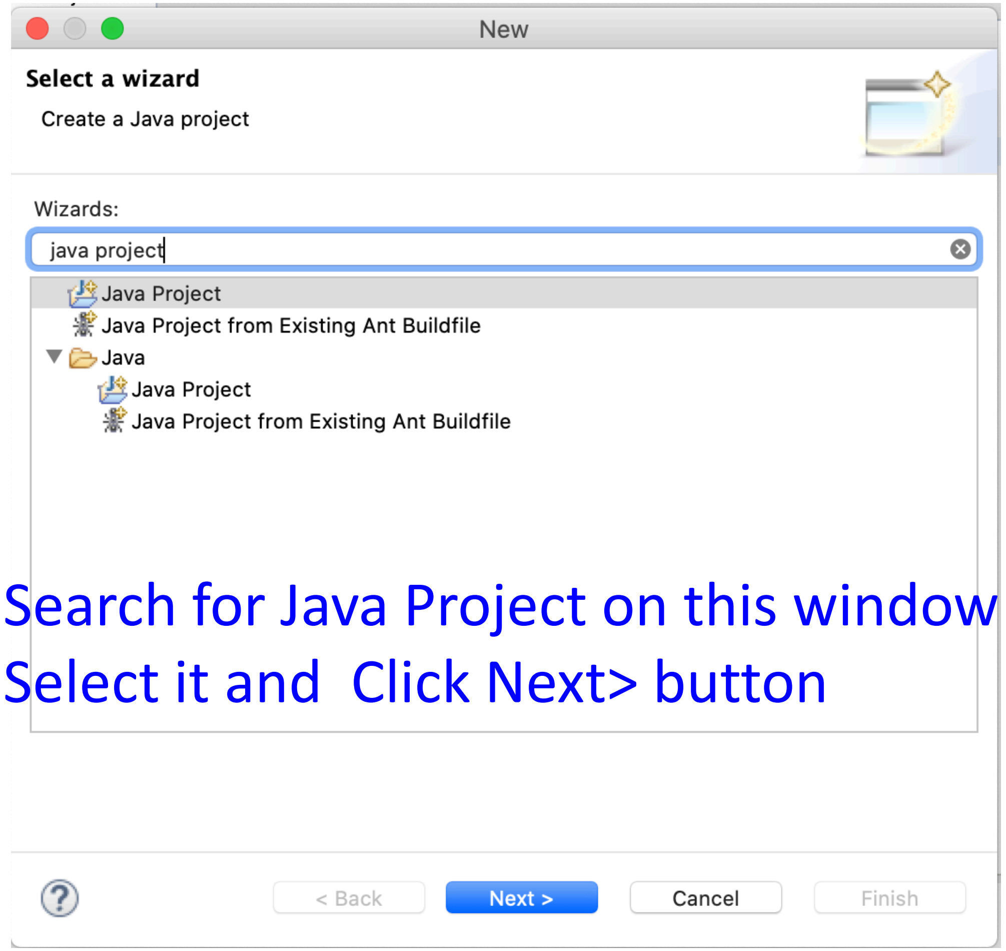
Configure JREs...

Don’t see Java Project? Do this!

Go to menu

File > New > Other



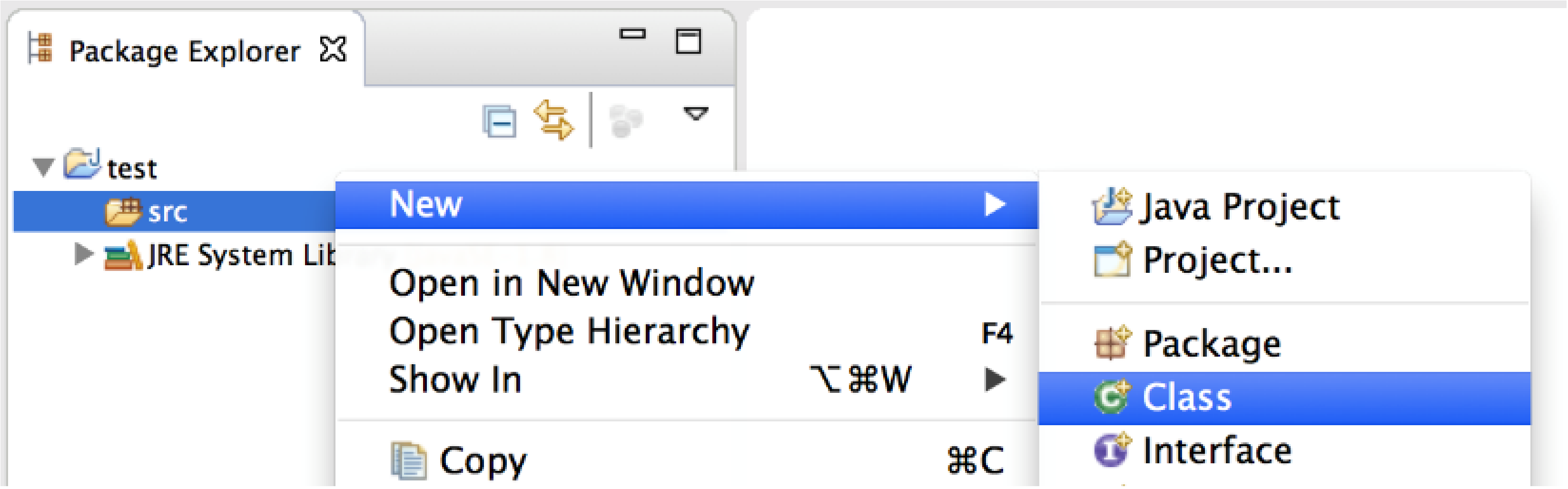


Search for Java Project on this window

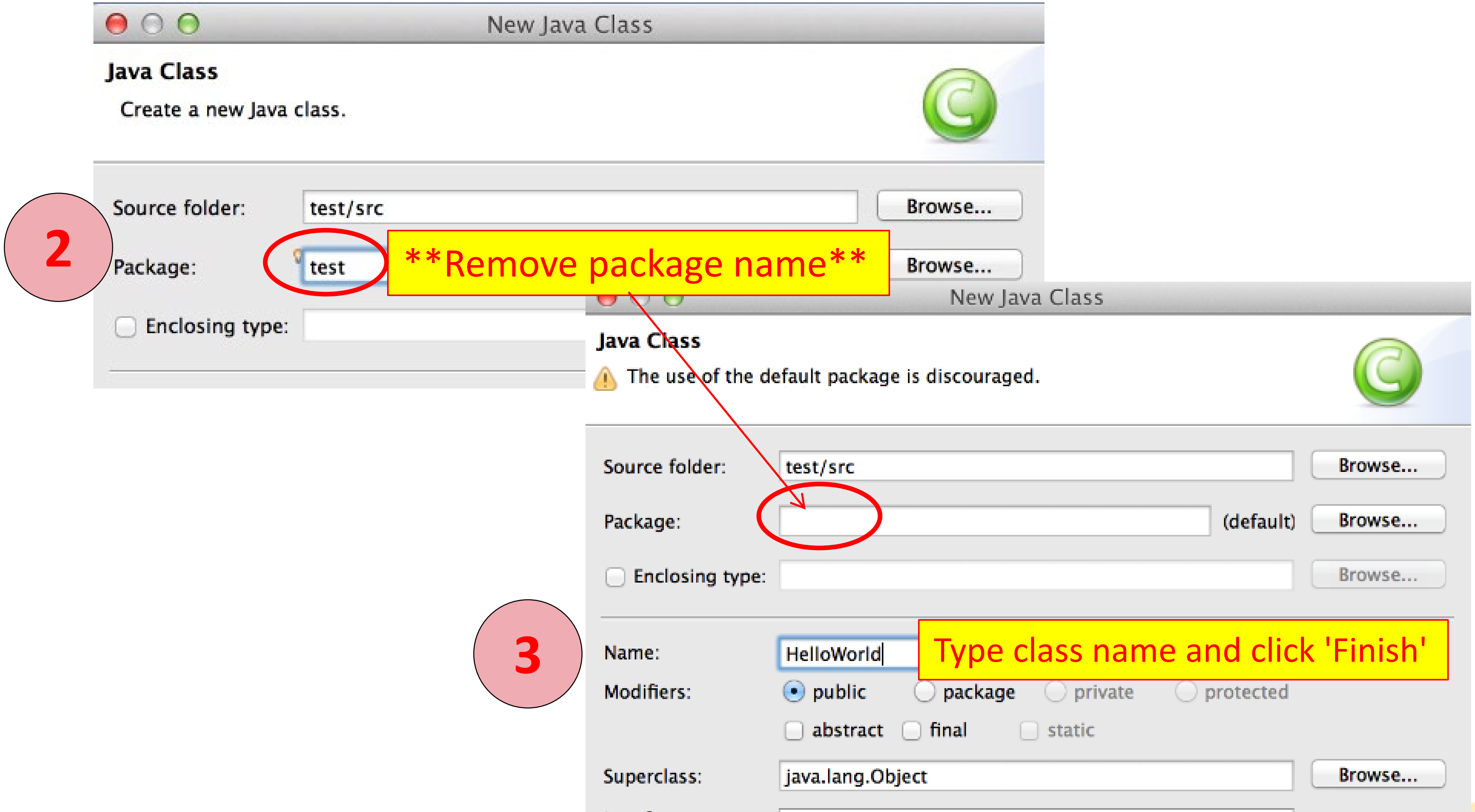
Select it and Click Next> button

Create Class - Using Eclipse

1



Create Class - Using Eclipse (Cont.)



2

New Java Class

Java Class

Create a new Java class.

Source folder: test/src [Browse...]

Package: test [Browse...]

□ Enclosing type:

Remove package name

3

New Java Class

Java Class

⚠ The use of the default package is discouraged.

Source folder: test/src [Browse...]

Package: (default) [Browse...]

□ Enclosing type: [Browse...]

Name: HelloWorld

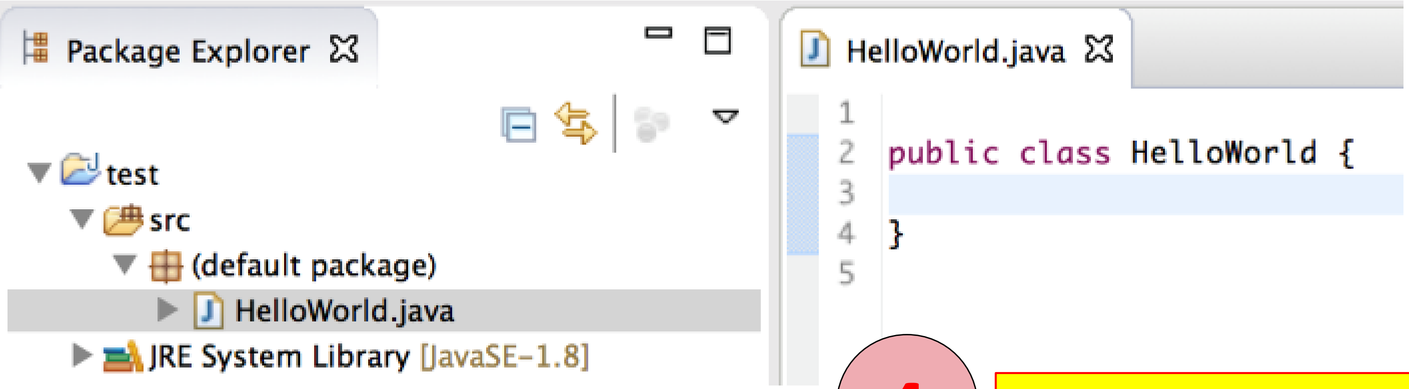
Type class name and click 'Finish'

Modifiers: ● public ○ package ○ private ○ protected

□ abstract □ final □ static

Superclass: java.lang.Object [Browse...]

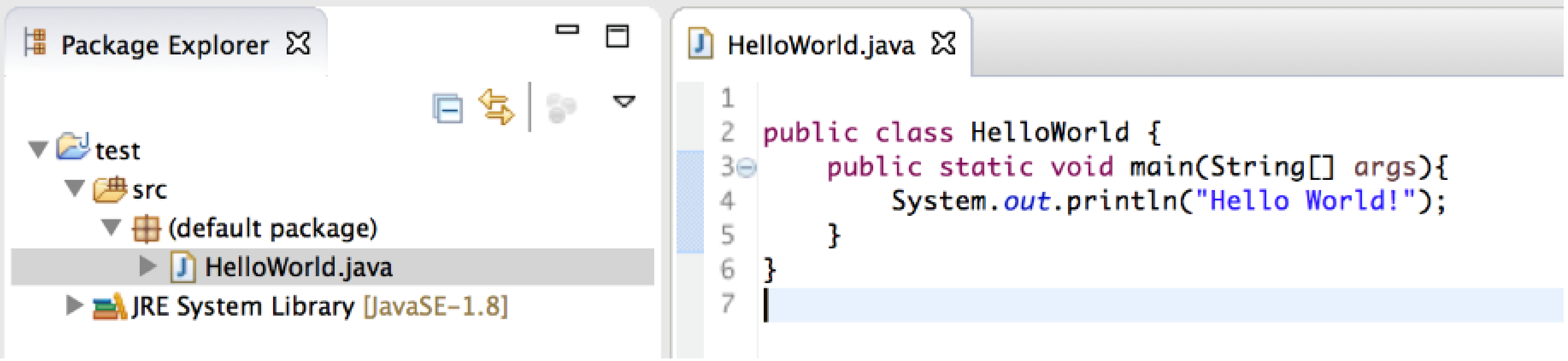
Create Class - Using Eclipse (Cont.)



```

public class HelloWorld {

}

```

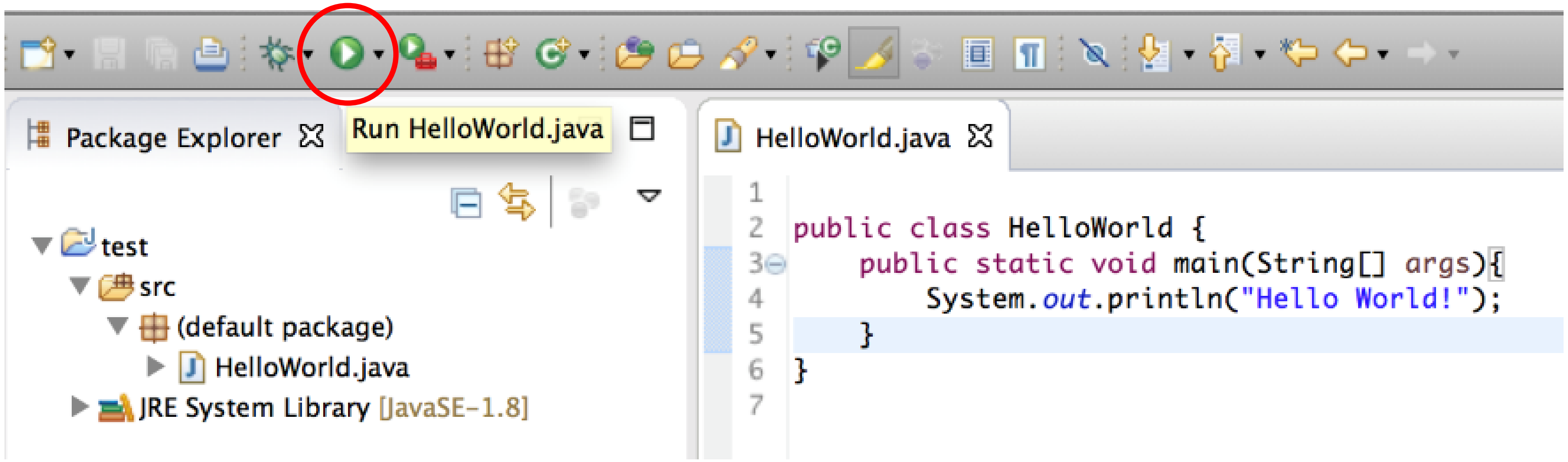
4

Write a simple program as shown below



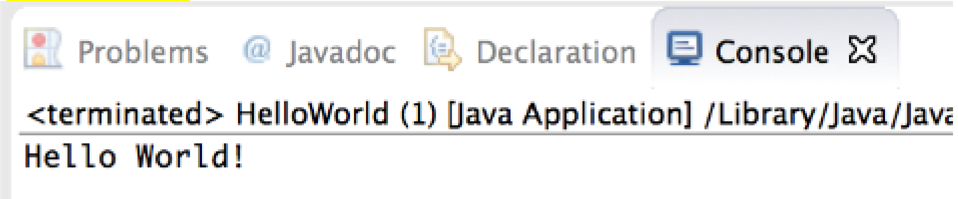
```

public class HelloWorld {
    public static void main(String[] args){
        System.out.println("Hello World!");
    }
}

```

Execute/Run - Using Eclipse



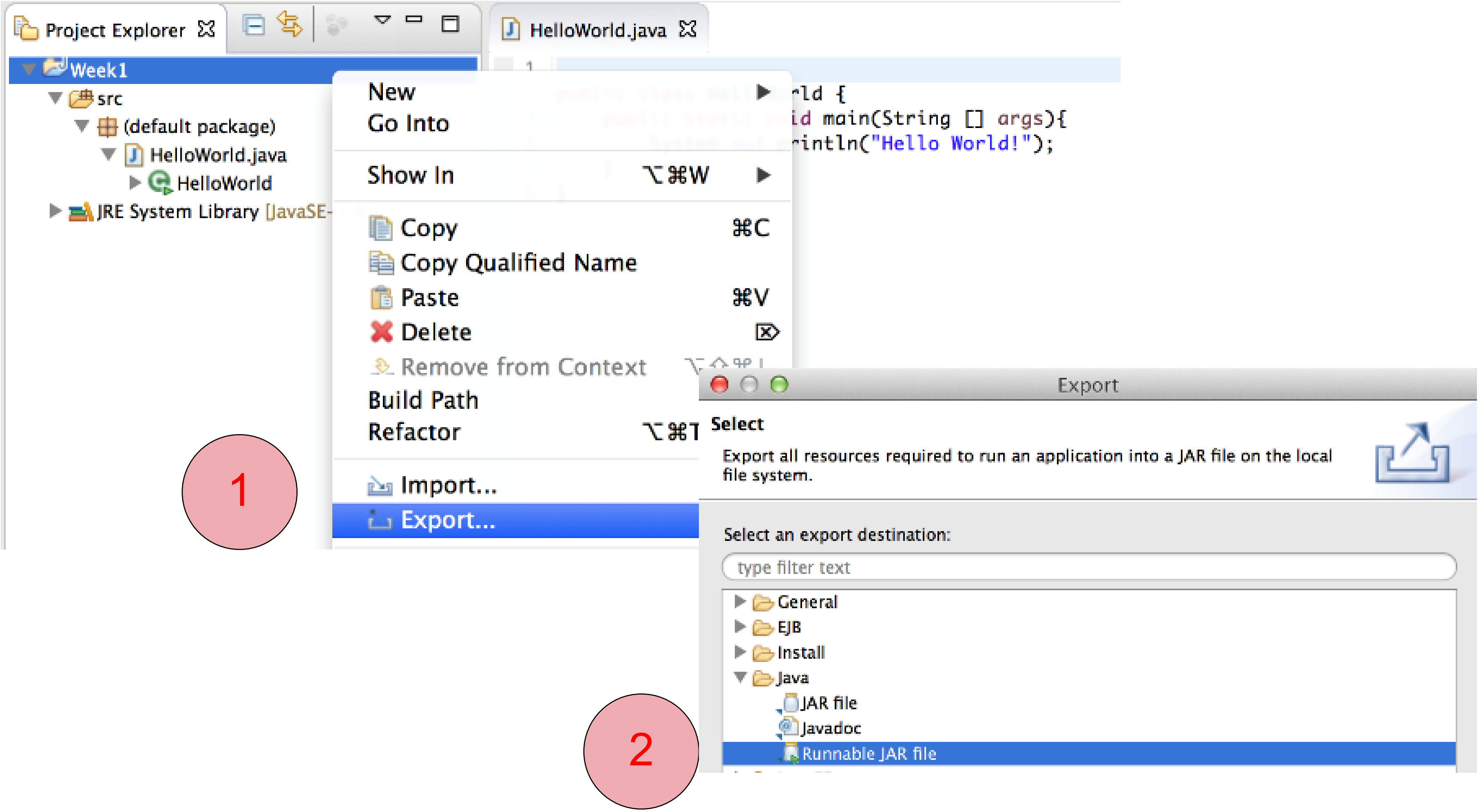
```

public class HelloWorld {
    public static void main(String[] args){
        System.out.println("Hello World!");
    }
}

```

Output



```
<terminated> HelloWorld (1) [Java Application] /Library/Java/Java
Hello World!
```

Creating .jar file



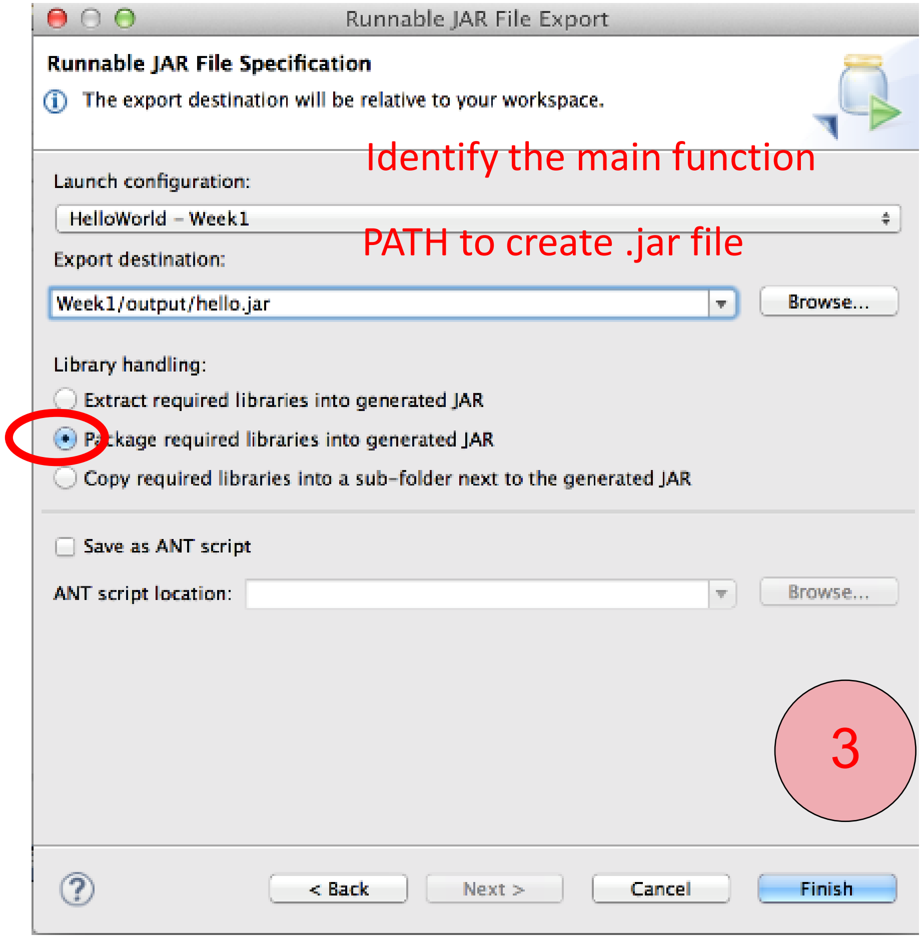
```
rld {
id main(String [] args){
rintln("Hello World!");
```

Select

Export all resources required to run an application into a JAR file on the local file system.

Select an export destination:

Creating .jar file (Cont.)



Runnable JAR File Export

Runnable JAR File Specification

The export destination will be relative to your workspace.

Launch configuration: HelloWorld – Week1

Export destination: Week1/output/hello.jar

Library handling:

○ Extract required libraries into generated JAR

● Package required libraries into generated JAR

○ Copy required libraries into a sub-folder next to the generated JAR

□ Save as ANT script

ANT script location:

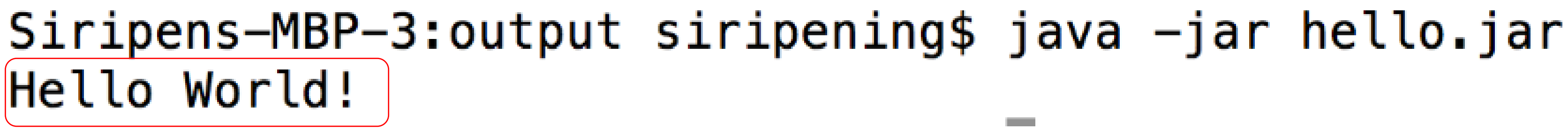
Identify the main function

PATH to create .jar file

3

Executing .jar file

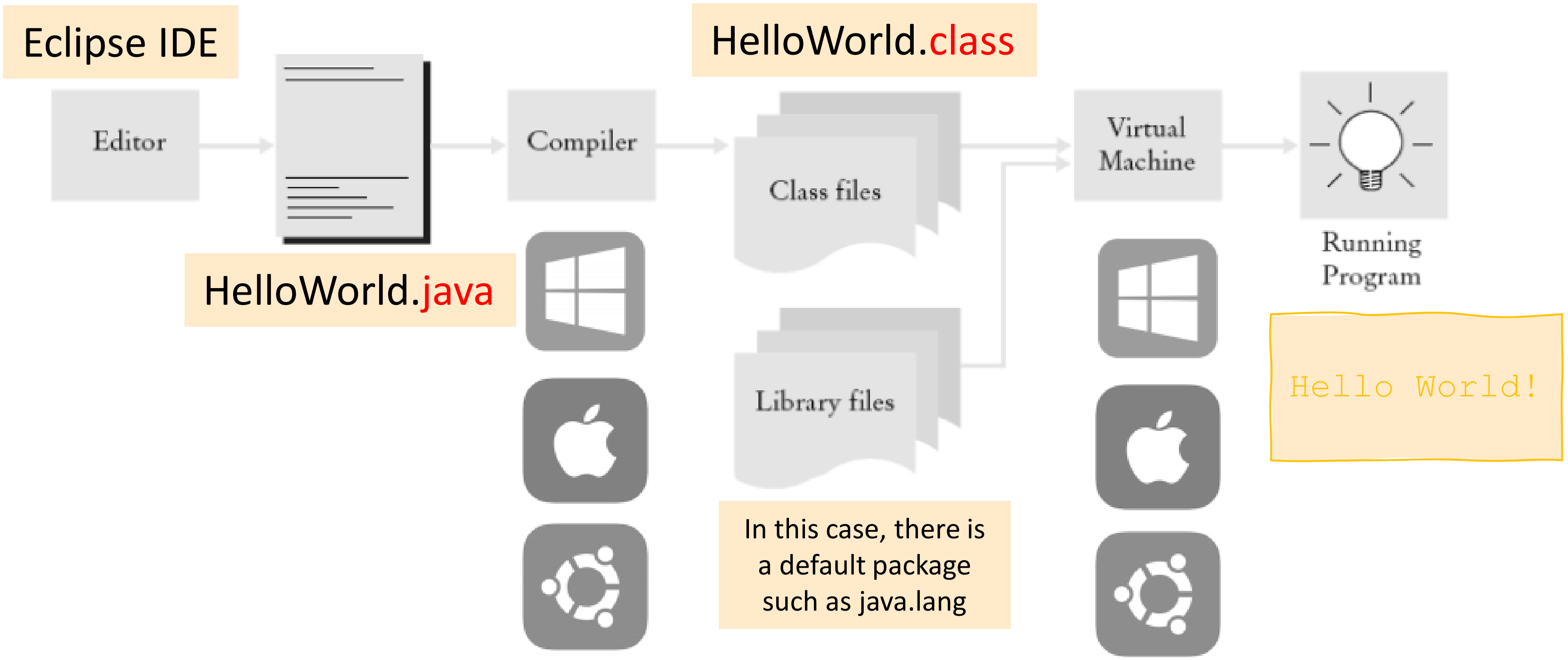
- `>> java -jar hello.jar`



```
Siripens-MBP-3:output siripening$ java -jar hello.jar
Hello World!
```

Recall: How Java Work?

- Scenario 2: Run java using Eclipse IDE



Eclipse IDE

HelloWorld.java

HelloWorld.class

In this case, there is a default package such as java.lang

Learning a new programming language

1. **Understand a concept**
   - Read textbook or online resources

2. **Understand a syntax**
   - Examine the syntax => Find a Cheat Sheet e.g. http://mindprod.com/jgloss/jcheat.html)

3. **Practice a programming logic**
   - Look at the example code, then run it!!!
   - Write you own code as soon as you understand the basic concept and syntax.

Self Check

How would you modify the `HelloWorld` program to print the words `"Hello,"` and `"World!"` on two lines?

Answer:

Self Check

What does the following set of statements print?

```
System.out.print("My lucky number is ");
System.out.println(3 + 4 + 5);
```

Answer:

Self Check

What does the following set of statements print?

```
System.out.println("My lucky number is " + 3 + 4 + 5);
```

Answer:

Errors

- Syntax errors

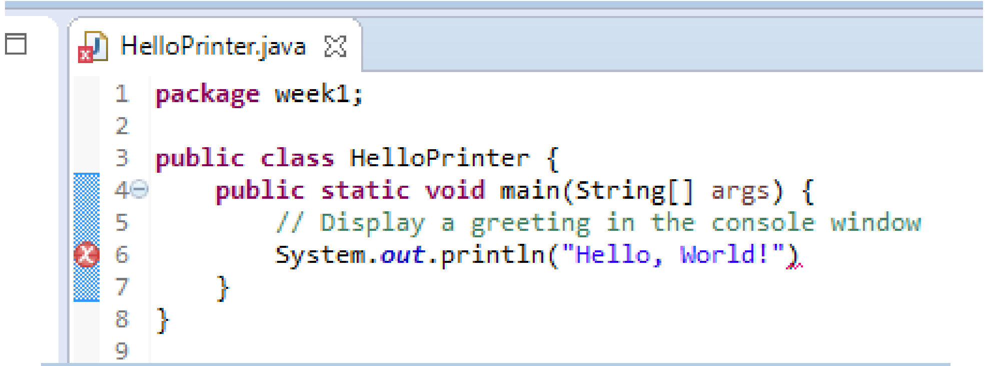
```
System.ouch.print(".  .  .");
System.out.print("Hello);
```

  - Detected by the compiler

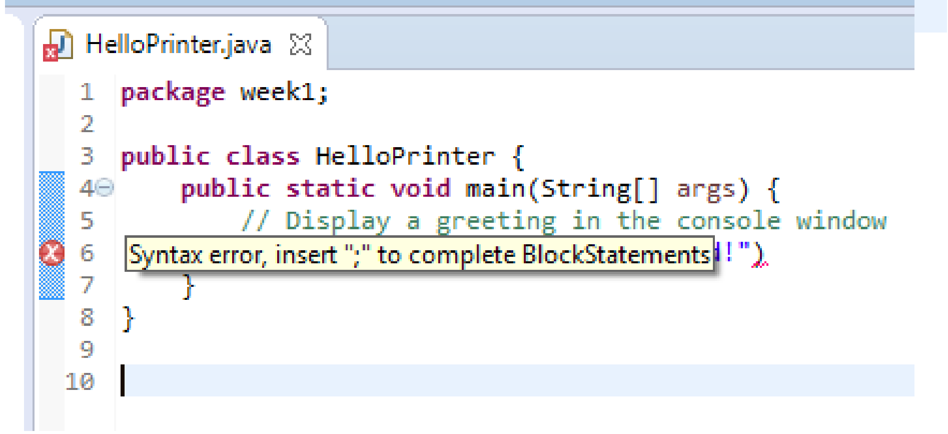
- Logic errors

```
System.out.print("Hell");
```

  - Detected (hopefully) through testing



```
package week1;

public class HelloPrinter {
    public static void main(String[] args) {
        // Display a greeting in the console window
        System.out.println("Hello, World!")
    }
}
```



```
package week1;

public class HelloPrinter {
    public static void main(String[] args) {
        // Display a greeting in the console window
        System.out.println("Hello, World!")
    }
}
```

Syntax error, insert ";" to complete BlockStatements

The Edit-Compile-Test Loop

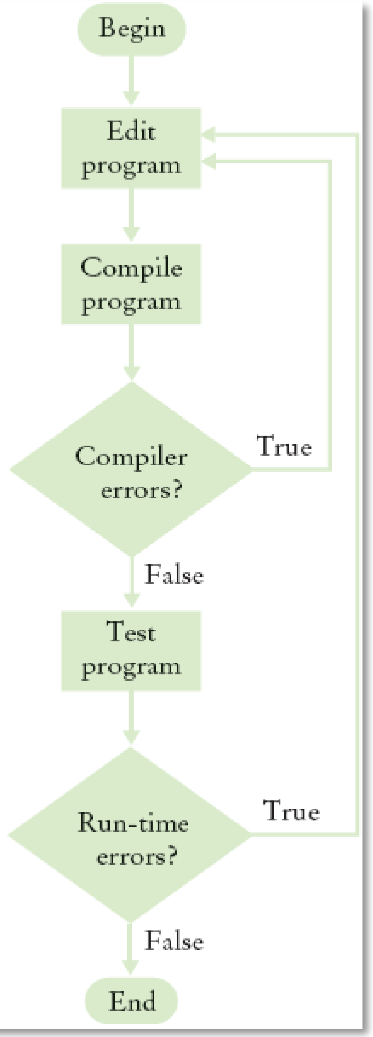


Figure 14
The Edit–Compile–Test Loop

Eclipse Java Code Templates

```
System.out.println("");
```

Let's use code template!

Just type **sysout**

Then press **ctrl + space**

You can add templates with

Preferences->Java->Editor ->Templates.

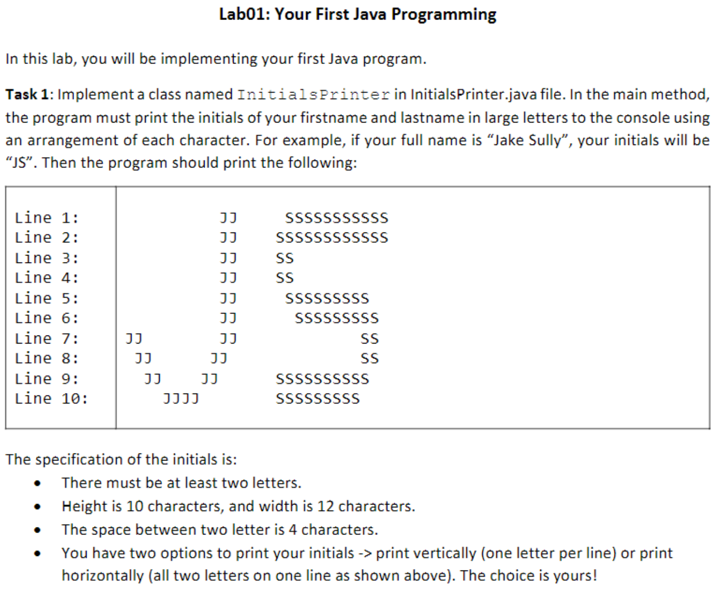
Lab Assignment

Main assignment (mandatory)

Due in class

Max 3 points

Count towards 12% of course scores (Lab)



Lab01: Your First Java Programming

In this lab, you will be implementing your first Java program.

Task 1: Implement a class named `InitialsPrinter` in InitialsPrinter.java file. In the main method, the program must print the initials of your firstname and lastname in large letters to the console using an arrangement of each character. For example, if your full name is "Jake Sully", your initials will be "JS". Then the program should print the following:

```
Line 1:             JJ     SSSSSSSSSSS
Line 2:             JJ    SSSSSSSSSSSS
Line 3:             JJ    SS
Line 4:             JJ    SS
Line 5:             JJ     SSSSSSSSS
Line 6:             JJ      SSSSSSSSS
Line 7:   JJ        JJ             SS
Line 8:    JJ      JJ              SS
Line 9:     JJ    JJ      SSSSSSSSSS
Line 10:      JJJJ        SSSSSSSSS
```

The specification of the initials is:

- There must be at least two letters.
- Height is 10 characters, and width is 12 characters.
- The space between two letter is 4 characters.
- You have two options to print your initials -> print vertically (one letter per line) or print horizontally (all two letters on one line as shown above). The choice is yours!

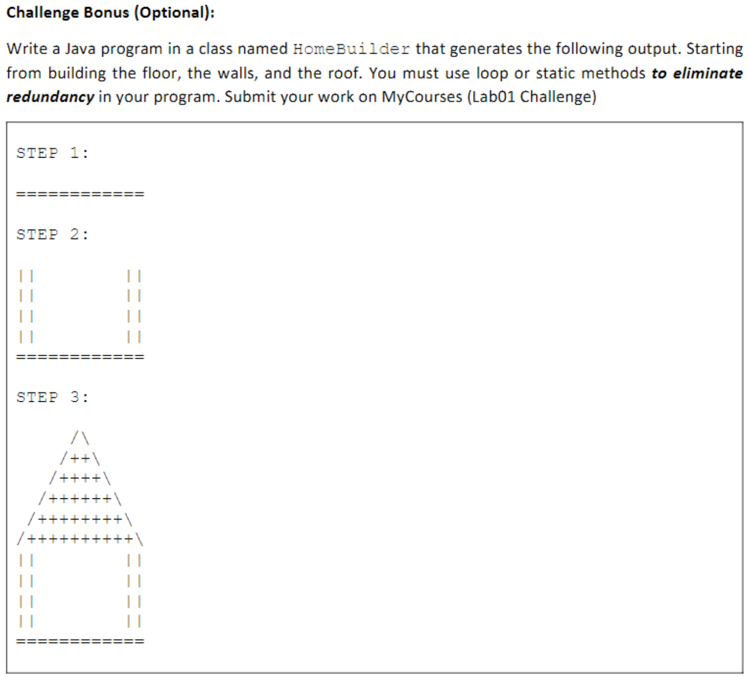
Deliverables

- Show the solution to TA by 6PM for grades
- Submit .java files on MyCourses for evidence

Challenge Assignment Bonus (Optional)

Max 3 Points

Count towards 5% bonus



Challenge Bonus (Optional):

Write a Java program in a class named `HomeBuilder` that generates the following output. Starting from building the floor, the walls, and the roof. You must use loop or static methods ***to eliminate redundancy*** in your program. Submit your work on MyCourses (Lab01 Challenge)

```
STEP 1:

============

STEP 2:

||        ||
||        ||
||        ||
||        ||
============

STEP 3:

     /\
    /++\
   /++++\
  /++++++\
 /++++++++\
/++++++++++\
||        ||
||        ||
||        ||
||        ||
============
```

Deliverables

- Show the solution to TA by the next lab meeting for grades
- Submit .java files on MyCourses Challenge Assignment