COMP3111: Software Engineering

Java Programming Basics

Learning Outcomes:

- Be able to write a Java program and compile it with Gradle
- Be able to describe the concept interface and inheritance in Java

Environment: Eclipse (Version: 2020-12) with Java Development Kit (JDK 15) installed on Windows 10. The steps involved may be slightly different if you are using other versions of Eclipse, or Eclipse on Mac.

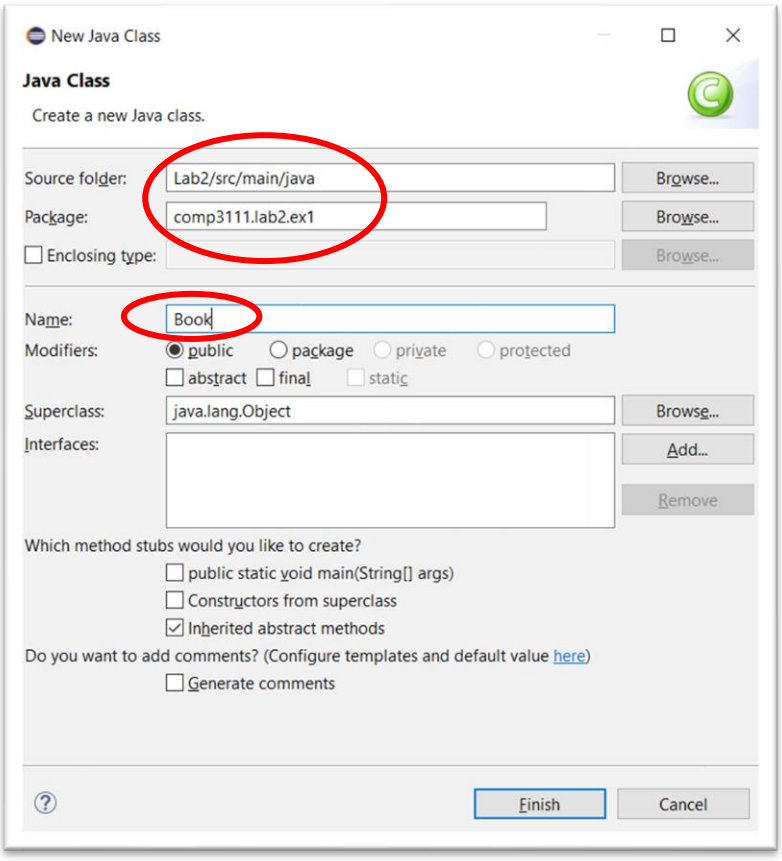
Pre-requisite Reading:
We assume you have C++ experience (from COMP2011/2012 or 2012H) while you might not have any Java experience. Before coming to the lab, please read the supplementary materials (available in Canvas):

- PPT Slides: COMP3021 Topic 1: Introduction to Java Programming

Exercise 1: Programming the basics

Step 1.1: Refer to the worksheet of Lab1 (Step 1.1 to Step 1.4) and create a new Gradle project (name the project as "Lab2" instead of "Lab1").

Step 1.2: Choose "File > New > Class". Create a class **Book** under the package **comp3111.lab2.ex1**:



Step 1.3: Open Library.java and edit it as follows. Please pay attention to the first line, which indicates that we are going to use the class Book from the package comp3111.lab2.ex1.

Library.java

```java
package Lab2;

import comp3111.lab2.ex1.Book;

public class Library {
    /* Add this function */
    public static void main(String arg[]) {
        final String array[] = {"Basic Java", "Advanced Java", "Guru Java"};
        Book b = new Book(array);
        int k = 2;
        System.out.println("The title of Chapter " + k + " is " + b.getChapter(k - 1));
        String anotherArray[] = b.getChapters();

        System.out.println("There are " + anotherArray.length + " chapters.");
        System.out.println(anotherArray);

    }

    public boolean someLibraryMethod() {
        return true;
    }
}
```

```
Table of Content
    • Chapter 1: Basic Java
    • Chapter 2: Advanced Java
    • Chapter 3: Guru Java
```

Step 1.4: Edit Book.java from the package comp3111.lab2.ex1. You could start your Book.java with the following program fragment and add your own code to complete it.

Book.java

```java
package comp3111.lab2.ex1;

public class Book {
    private String chapters[];
    private static final int DEFAULT_CHAPTERS = 10;

    public Book() {
        chapters = new String[DEFAULT_CHAPTERS];
        for (int i =0; i < chapters.length; i++)
            chapters[i] = "n/a";
    }

    public Book(String argument[]) {
        /* construct the object with an array of chapter names */
        /* ADD YOUR CODE HERE */
    }

    public String getChapter(int i) {
        /* return the chapter by the given index */
        /* ADD YOUR CODE HERE */
    }

    public String[] getChapters() {
        return chapters;
    }
}
```

Verify that you have **build.gradle** properly configured as follows:

build.gradle

```
plugins {
    id 'java'
    id 'application'
}

repositories {
    jcenter()
}

dependencies {
    // Use JUnit test framework
    testImplementation 'junit:junit:4.12'
}

mainClassName = 'Lab2.Library'

```

On execution, the program is expected to give the following results:

```
> Task :compileJava
> Task :processResources NO-SOURCE
> Task :classes

> Task :run
The title of Chapter 2 is Advanced Java
There are 3 chapters.
[Ljava.lang.String;@1c20c684

BUILD SUCCESSFUL in 1s
2 actionable tasks: 2 executed
```

Step 1.5: When we print an array variable directly, it will only print out the type and its address. We can instead use some APIs like **java.util.Arrays** for help. In Library.java, replacing the line:

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

with

```
System.out.println(java.util.Arrays.toString(anotherArray));
```

The results of execution will then become:

```
> Task :compileJava
> Task :processResources NO-SOURCE
> Task :classes

> Task :run
The title of Chapter 2 is Advanced Java
There are 3 chapters.
[Basic Java, Advanced Java, Guru Java]

BUILD SUCCESSFUL in 1s
2 actionable tasks: 2 executed
```

*Note1: the API java.util.Arrays is from another package. Therefore, we need to either be verbose to refer the class together with the package name (like what we did) or to import the package at the top of the program (adding **import java.util.Arrays;**)*

Note2: to learn how other API works, you can refer to the documentation on Java 8 API
https://docs.oracle.com/javase/8/docs/api/java/util/Arrays.html

Exercise 2: Java Interfaces and Inheritance

Step 2.1: Create another package comp3111.lab2.ex2
Step 2.2: Create two classes: Computer and MobileComputer as below, so that MobileComputer inherits from Computer.

Book.java | Library.java | build.gradle | Computer.java | MobileComputer.java

```
1  package comp3111.lab2.ex2;
2
3  public class Computer {
4      protected String secret;
5      public Computer() {
6          secret = "computer secret";
7      }
8      public void work() {
9          System.out.println("A computer is working");
10     }
11 }
12
```

Book.java | Library.java | build.gradle | Computer.java | MobileComputer.java

```
1  package comp3111.lab2.ex2;
2
3  public class MobileComputer extends Computer {
4      private int battery;
5      public MobileComputer() {
6          secret = "MobileComputer secret";
7          battery = 5;
8      }
9      @Override
10     public void work() {
11         if (battery > 0) {
12             System.out.println("It is working on my lap.");
13             battery--;
14         } else
15             System.out.println("Running out of battery");
16     }
17     public void charge() {
18         if (battery < 10)
19             battery++;
20     }
21 }
22
```

*Note: We use the keyword **extends** to inherit a base class.*
Note: @Override is an annotation. This annotation explicitly tells the compiler that we are overriding the parent's method (or member function in C++ terminology).

Step 2.3: Change the driver program Library.java as below. Run the program to verify your result.

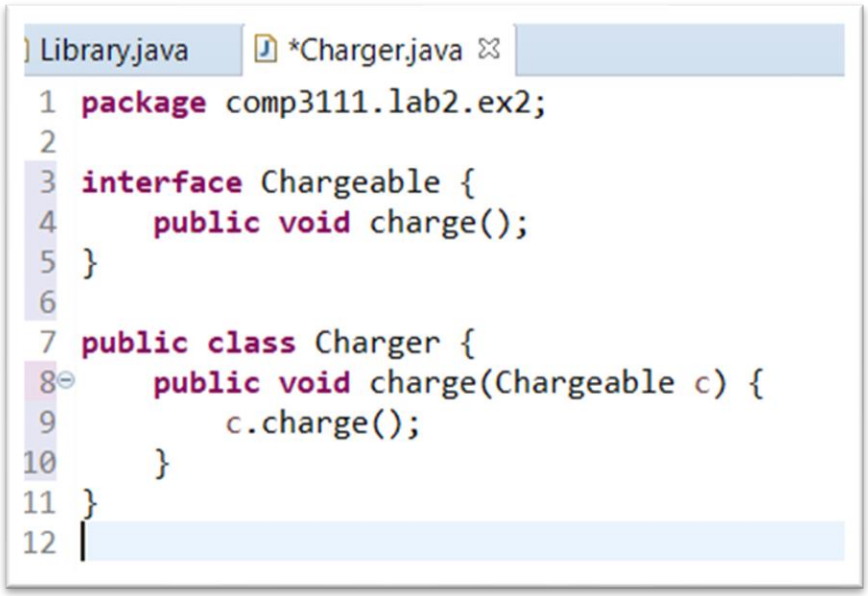
Library.java

```
1  package Lab2;
2
3  import comp3111.lab2.ex1.Book;
4  import comp3111.lab2.ex2.*;
5
6  public class Library {
7      /* Add this function */
8      public static void main(String arg[]) {
9          final String array[] = {"Basic Java", "Advanced Java", "Guru Java"};
10         Book b = new Book(array);
11         int k = 2;
12         System.out.println("The title of Chapter " + k + " is " + b.getChapter(k - 1));
13         String anotherArray[] = b.getChapters();
14
15         System.out.println("There are " + anotherArray.length + " chapters.");
16         System.out.println(java.util.Arrays.toString(anotherArray));
17
18         /* ex2: Add the following */
19         Computer a = new MobileComputer();
20         for (int i = 0; i < 10; i++)
21             a.work();
22     }
23 }
```

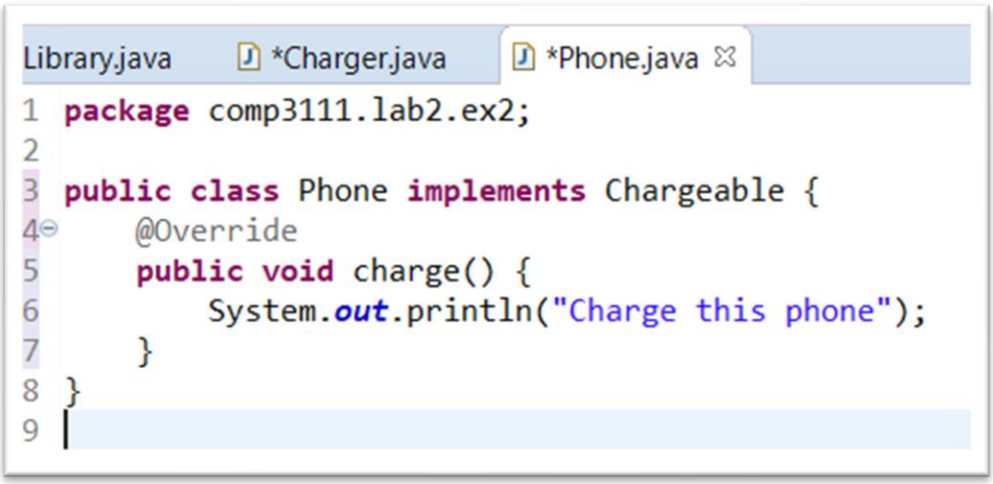
The results of execution should look like:

```
> Task :compileJava
> Task :processResources NO-SOURCE
> Task :classes

> Task :run
The title of Chapter 2 is Advanced Java
There are 3 chapters.
[Basic Java, Advanced Java, Guru Java]
It is working on my lap.
It is working on my lap.
It is working on my lap.
It is working on my lap.
It is working on my lap.
Running out of battery
Running out of battery
Running out of battery
Running out of battery
Running out of battery

BUILD SUCCESSFUL in 1s
2 actionable tasks: 2 executed
```

Step 2.4: Create a class called Charger that has one function. Also create the interface Chargeable inside this Charger.java



Step 2.5: Create another class Phone as below.



Note: Using C++ terminology, an interface in Java has only pure-virtual functions. Unlike an abstract class which is allowed to have non pure-virtual functions, an interface cannot contain any implementation.

Step 2.6. Then change the driver program Library.java as:

Library.java

```
1  package Lab2;
2
3  import comp3111.lab2.ex1.Book;
4  import comp3111.lab2.ex2.*;
5
6  public class Library {
7      /* Add this function */
8      public static void main(String arg[]) {
9          final String array[] = {"Basic Java", "Advanced Java", "Guru Java"};
10         Book b = new Book(array);
11         int k = 2;
12         System.out.println("The title of Chapter " + k + " is " + b.getChapter(k - 1));
13         String anotherArray[] = b.getChapters();
14
15         System.out.println("There are " + anotherArray.length + " chapters.");
16         System.out.println(java.util.Arrays.toString(anotherArray));
17
18         /* ex2: Add the following */
19         Computer a = new MobileComputer();
20         for (int i = 0; i < 10; i++)
21             a.work();
22
23         Charger c = new Charger();
24         Phone p = new Phone();
25         MobileComputer m = new MobileComputer();
26
27         c.charge(p);
28         c.charge(m);    // this does not work without fixing MobileComputer
29     }
30 }
```

Note1: An error at line 28 is expected on compiling Library.java. You need to identify the cause of the error and try to fix it as part of the lab assignment.

Note2: Tutorial on Interfaces and Inheritance: https://docs.oracle.com/javase/tutorial/java/IandI/index.html

Lab Assignment and Submission

Lab Assignment

Work out the solutions for the following tasks:

- Complete the missing code in comp3111.lab2.ex1.Book.java
- Fix comp3111.lab2.ex2.MobileComputer.java to make it work

Submission

Submit your answers through the Text Entry box:

- Identify the error in the initial version of MobileComputer.java
- Describe how you have fixed the problem and explain why your solution works