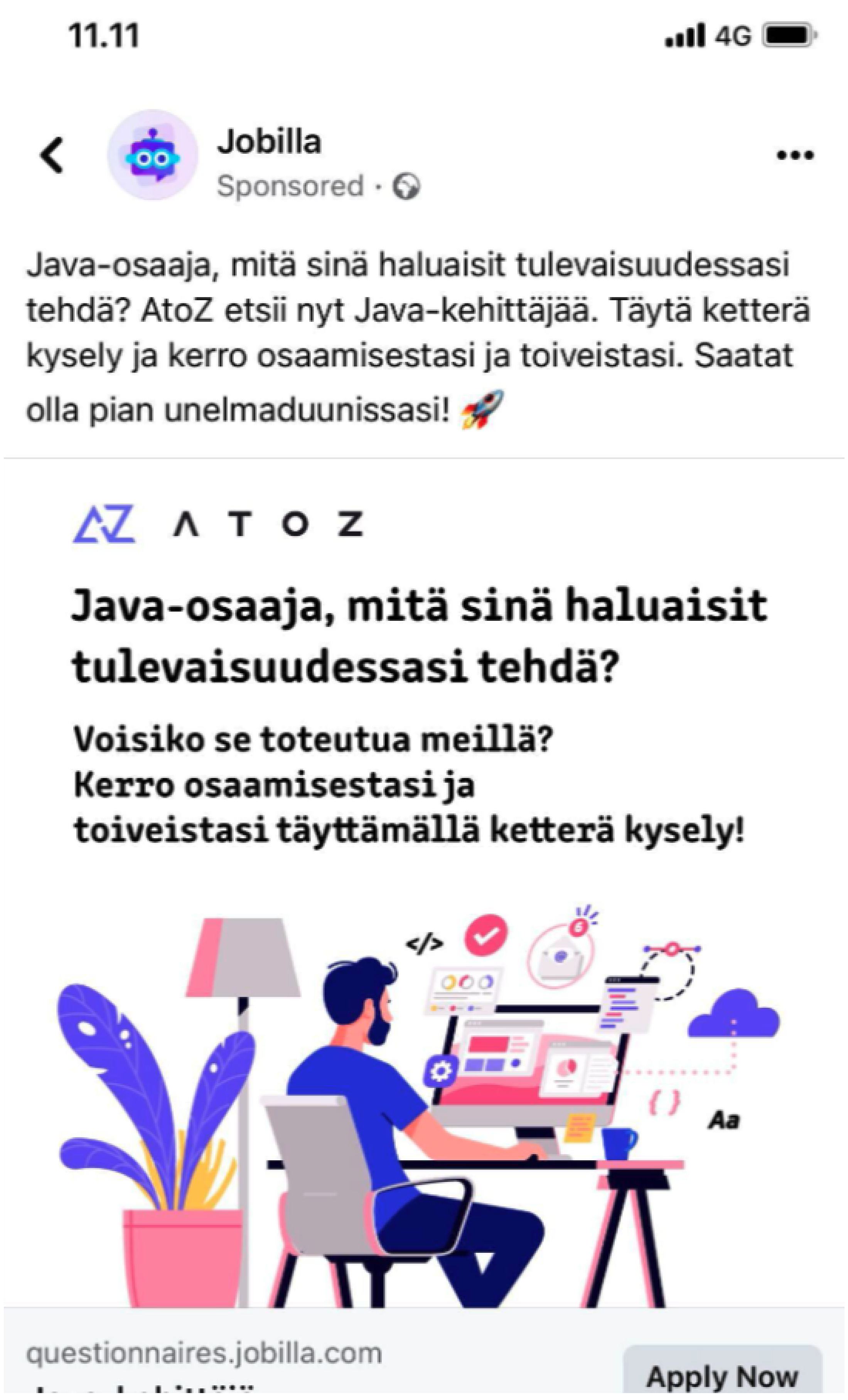
Week 02 - Hello Java

Louis Botha, `louis.botha@tuni.fi`

Java is a general-purpose, class-based, object-oriented programming language designed for having lesser implementation dependencies. It is a computing platform for application development. Java is fast, secure, and reliable, therefore. It is widely used for developing Java applications in laptops, data centers, game consoles, scientific supercomputers, cell phones, etc.

The goal is to write one application and have it work on whatever device. The application will run there as long as you have a JVM installed.

Yes, there is also Java Developer jobs available, advertisement on my Facebook recently



11.11 4G

Jobilla
Sponsored ·

Java-osaaja, mitä sinä haluaisit tulevaisuudessasi tehdä? AtoZ etsii nyt Java-kehittäjää. Täytä ketterä kysely ja kerro osaamisestasi ja toiveistasi. Saatat olla pian unelmaduunissasi! 🚀

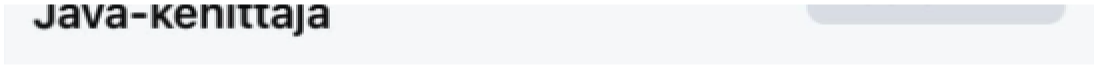
ATOZ

Java-osaaja, mitä sinä haluaisit tulevaisuudessasi tehdä?

Voisiko se toteutua meillä?
Kerro osaamisestasi ja toiveistasi täyttämällä ketterä kysely!

questionnaires.jobilla.com
[illegible]

Apply Now



Java Features

Here are some important Java features:

- It is one of the easy-to-use programming languages to learn.
- Write code once and run it on almost any computing platform.
- Java is platform-independent. Some programs developed in one machine can be executed in another machine.
- It is designed for building object-oriented applications.
- It is a multithreaded language with automatic memory management.
- It is created for the distributed environment of the Internet.
- Facilitates distributed computing as its network-centric.

Java Development kit (JDK)

JDK is a software development environment used for making applets and Java applications. The full form of JDK is Java Development Kit. Java developers can use it on Windows, macOS, Solaris, and Linux. JDK helps them to code and run Java programs. It is possible to install more than one JDK version on the same computer.

Why use JDK?

Here are the main reasons for using JDK:

- JDK contains tools required to write Java programs and JRE to execute them.
- It includes a compiler, Java application launcher, Appletviewer, etc.
- Compiler converts code written in Java into byte code.
- Java application launcher opens a JRE, loads the necessary class, and executes its main method.

Java Virtual Machine (JVM)

Java Virtual Machine (JVM) is an engine that provides a runtime environment to drive the Java Code or applications. It converts Java bytecode into machine language. JVM is a part of the Java Run Environment (JRE). In other programming languages, the compiler produces machine code for a particular system. However, the Java compiler produces code for a Virtual Machine known as Java Virtual Machine.

Why JVM?
Here are the important reasons of using JVM:

- JVM provides a platform-independent way of executing Java source code.
- It has numerous libraries, tools, and frameworks.
- Once you run a Java program, you can run on any platform and save lots of time.
- JVM comes with JIT (Just-in-Time) compiler that converts Java source code into low-level machine language. Hence, it runs faster than a regular application.

Java Runtime Environment (JRE)

JRE is a piece of software that is designed to run other software. It contains the class libraries, loader class, and JVM. In simple terms, if you want to run a Java program, you need JRE. If you are not a programmer, you don't need to install JDK, but just JRE to run Java programs.

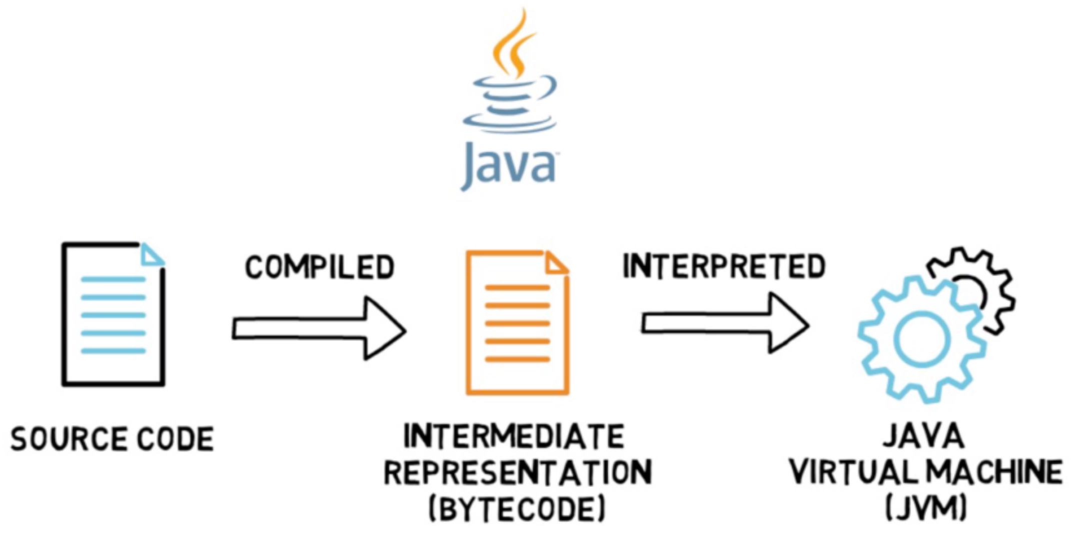
Why use JRE?
Here are the main reasons of using JRE:

- JRE contains class libraries, JVM, and other supporting files. It does not include any tool for Java development like a debugger, compiler, etc.
- It uses important package classes like math, swing, util, lang, awt, and runtime libraries.
- If you have to run Java applets, then JRE must be installed in your system.

How is Java Platform Independent?

The Java compiler does not produce native executable code for a particular machine. Instead, Java produces a unique format called. It executes according to the rules laid out in the virtual machine specification. Therefore, Java is a platform-independent language.

Bytecode is understandable to any JVM installed on any OS. In short, the java source code can run on all operating systems.


Java
COMPILED
INTERPRETED
SOURCE CODE
INTERMEDIATE REPRESENTATION (BYTECODE)
JAVA VIRTUAL MACHINE (JVM)

Compiled Programming Languages – Java

My First App

Verify that the Java tools are installed and available.

:protip: Checking the JDK, JRE and VM

```
java --version
// JDK
openjdk 17.0.2 2022-01-18
// JRE
OpenJDK Runtime Environment Temurin-17.0.2+8 (build 17.0.2+8)
// Virtual Machine
OpenJDK 64-Bit Server VM Temurin-17.0.2+8 (build 17.0.2+8, mix
ed mode, sharing)


javac --version
javac 17.0.2
```

Compiling and Running Java Programs

- Open the directory where you cloned the 1st GitHub classroom repository

- Create a directory in the repository called `src` and changed into it.

- Create a file and call it `MyFirstApp.java`, in any way you like.

- Open `MyFirstApp.java` in :vscode:VS Code for editing

- In a new file we need to start by defining the class. We need to give it a name, the class name and the filename should be the same and in the same case as well. We will learn more about classes later on.

```
class MyFirstApp {
}
```

> *:attention: Notice the curly braces always need to open and close*

- Inside the class, create a method called main. We will soon learn more about methods.

```
class MyFirstApp {
    public static void main(String[] args) {
    }
}
```

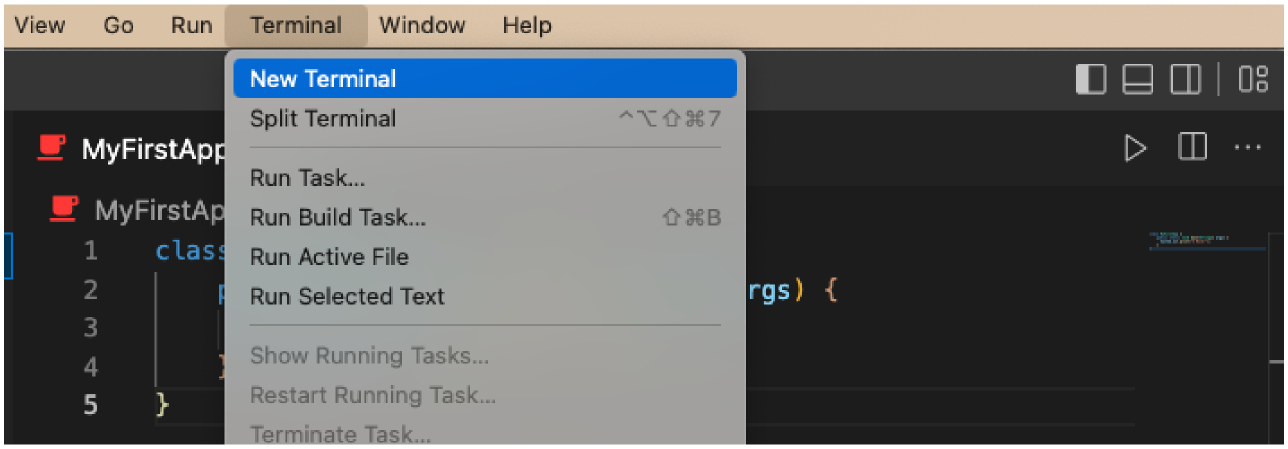
> *:attention: Notice again that the curly braces need to open and close*

- Now that we have a class and a **main** method, we can actually make the program do something. We can use an inbuilt class of Java called **System** to print something to screen.

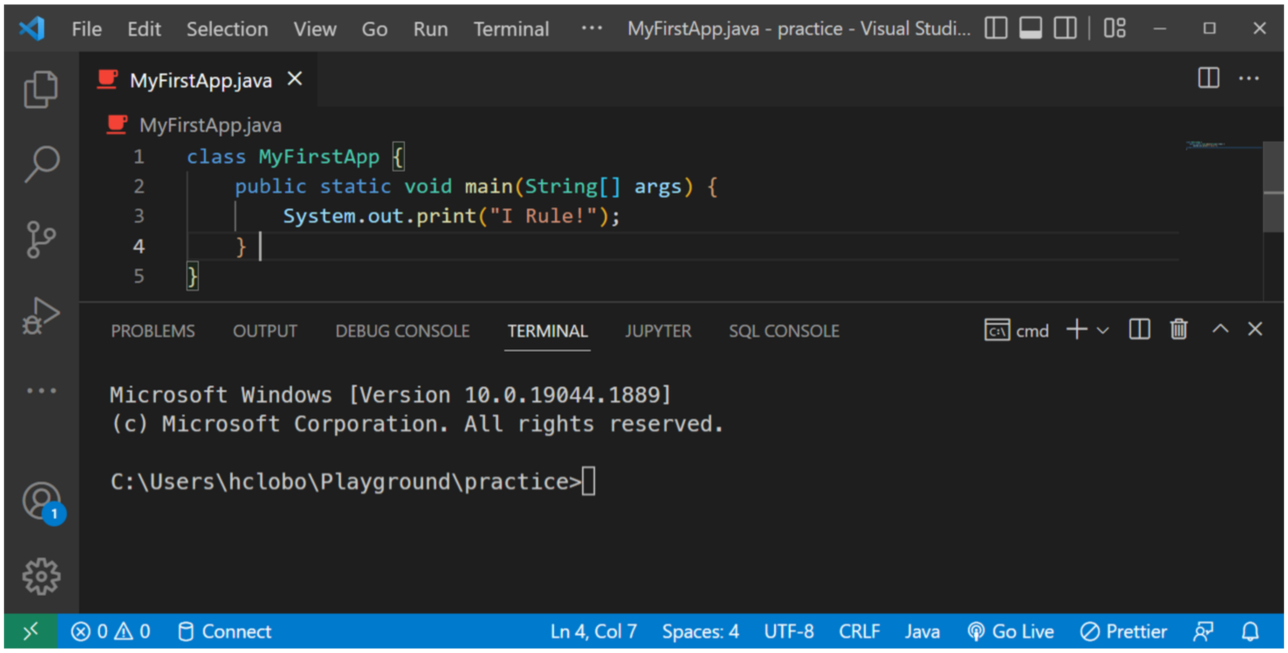
```
class MyFirstApp {
    public static void main(String[] args) {
        System.out.println("I Rule!");
    }
}
```

> *:attention: Notice the indentation*

- Save the file, you can use the keyboards shortcuts `ctrl + s` or `cmd + s`.
- Open the directory where the `MyFirstApp.java` file is located in a terminal. You can also open a terminal inside :vscode:. On Windows use the Command Prompt terminal for now.



On Windows use the Command Prompt terminal for now.


File Edit Selection View Go Run Terminal ··· MyFirstApp.java - practice - Visual Studi...
MyFirstApp.java
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER SQL CONSOLE
cmd
0 0 Connect Ln 4, Col 7 Spaces: 4 UTF-8 CRLF Java Go Live Prettier

```java
class MyFirstApp {
    public static void main(String[] args) {
        System.out.print("I Rule!");
    }
}
```

```
Microsoft Windows [Version 10.0.19044.1889]
(c) Microsoft Corporation. All rights reserved.

C:\Users\hclobo\Playground\practice>
```

- Compile the code

```
javac MyFirstApp.java
```

- List the contents of the directory, `dir` (Win) or `ls` (Mac/Linux)

```
> ls

MyFirstApp.class        MyFirstApp.java
```

Notice that the compiler has created .class file of the .java file.

- Run the code

```
java MyFirstApp


I Rule!
```

You don’t need to write the .class at the end to run the code.

- Change the code, to print out **Hello Java!**, save the file and run it

```
java MyFirstApp


I Rule!
```

Notice that the output of the application is still the same.

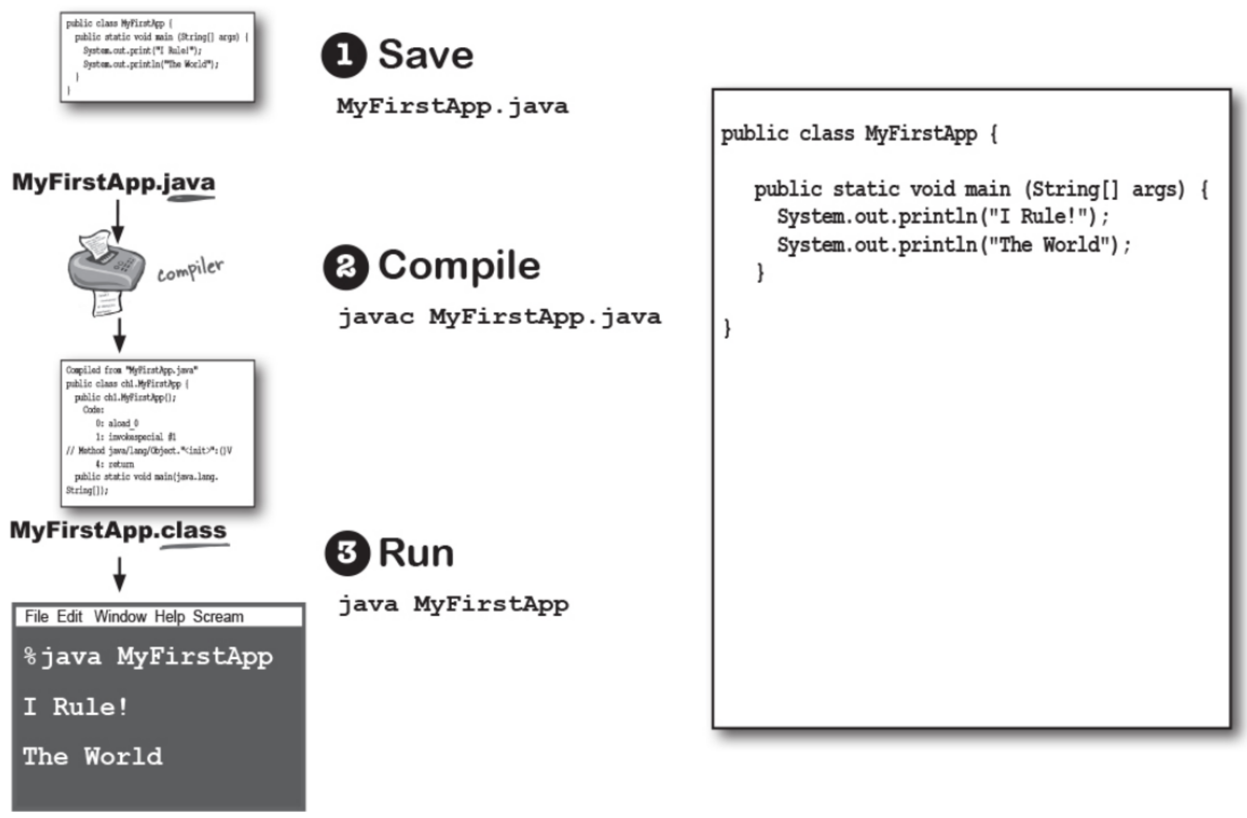
- Compile the code

```
javac MyFirstApp.java
```

- Run the code

```
java MyFirstApp


Hello Java!
```

> *:attention: For changes in the code to be visible you have to recompile the code!*



```
public class MyFirstApp {
  public static void main (String[] args) {
    System.out.print("I Rule!");
    System.out.println("The World");
  }
}
```

1 Save

```
MyFirstApp.java
```

2 Compile

```
javac MyFirstApp.java
```

```
Compiled from "MyFirstApp.java"
public class ch1.MyFirstApp {
  public ch1.MyFirstApp();
    Code:
       0: aload_0
       1: invokespecial #1
// Method java/lang/Object."<init>":()V
       4: return
  public static void main(java.lang.
String[]);
```

3 Run

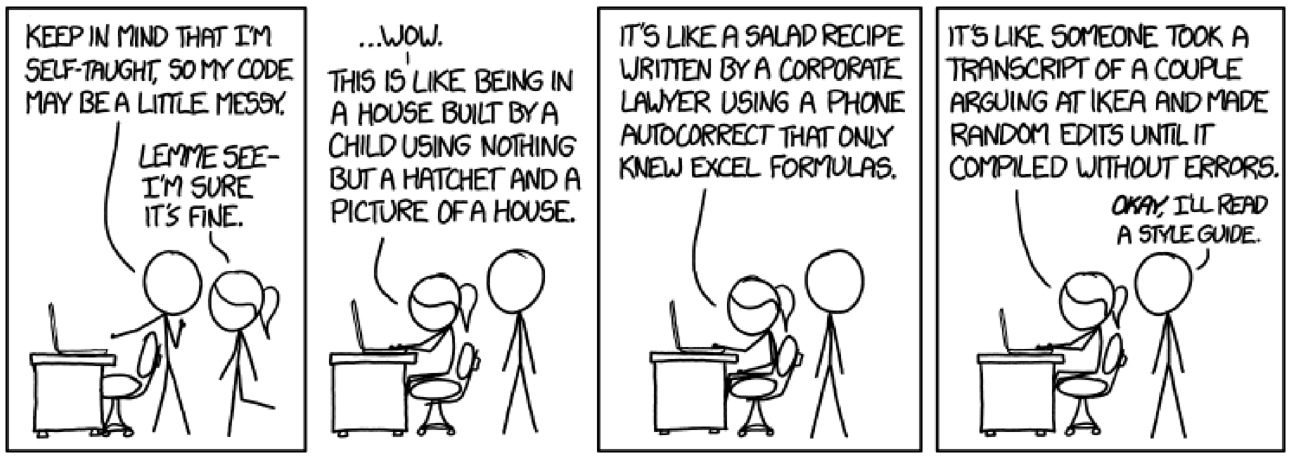
```
java MyFirstApp
```

```
%java MyFirstApp
I Rule!
The World
```

```
public class MyFirstApp {

   public static void main (String[] args) {
     System.out.println("I Rule!");
     System.out.println("The World");
   }

}
```

Head First Java

Java Style Guide


KEEP IN MIND THAT I'M SELF-TAUGHT, SO MY CODE MAY BE A LITTLE MESSY.
LEMME SEE— I'M SURE IT'S FINE.
...WOW.
THIS IS LIKE BEING IN A HOUSE BUILT BY A CHILD USING NOTHING BUT A HATCHET AND A PICTURE OF A HOUSE.
IT'S LIKE A SALAD RECIPE WRITTEN BY A CORPORATE LAWYER USING A PHONE AUTOCORRECT THAT ONLY KNEW EXCEL FORMULAS.
IT'S LIKE SOMEONE TOOK A TRANSCRIPT OF A COUPLE ARGUING AT IKEA AND MADE RANDOM EDITS UNTIL IT COMPILED WITHOUT ERRORS.
OKAY, I'LL READ A STYLE GUIDE.

XKCD #1513: Code Quality

For now, check:

- that you indent your code with **4 spaces** and not a **TAB** (In your editor set the default that tab insert 2 or 4 spaces).
- that the curly braces is opening on the same line as the statement and closing on its own line.
- that each statement ends with **;**.
- that the class names are in UpperCamelCase.
- that method names are in camelCase.
- that you use comments `// A comment` and `/* A longer multiline comment */`
  (Comments is a good way to take notes straight in code).
- that a line of code is not too wide, between 80 and 100 characters (Basically, you don't want to scroll sideways to read a line of code).

There will be more items as time goes by, but let's start from here.

FYI, there is as many Code Styles Guides as there is Lead Software Developers in the world. Some companies like Google has made their code style available for others to use.

Don't go search for the style guides just now as there is still too much terms that you will be unfamiliar with and it would make no sense, but we will get there 🙂

Exercise

The challenge is to use what you know so far and change the `MyFirstApp` that it will print out the following:

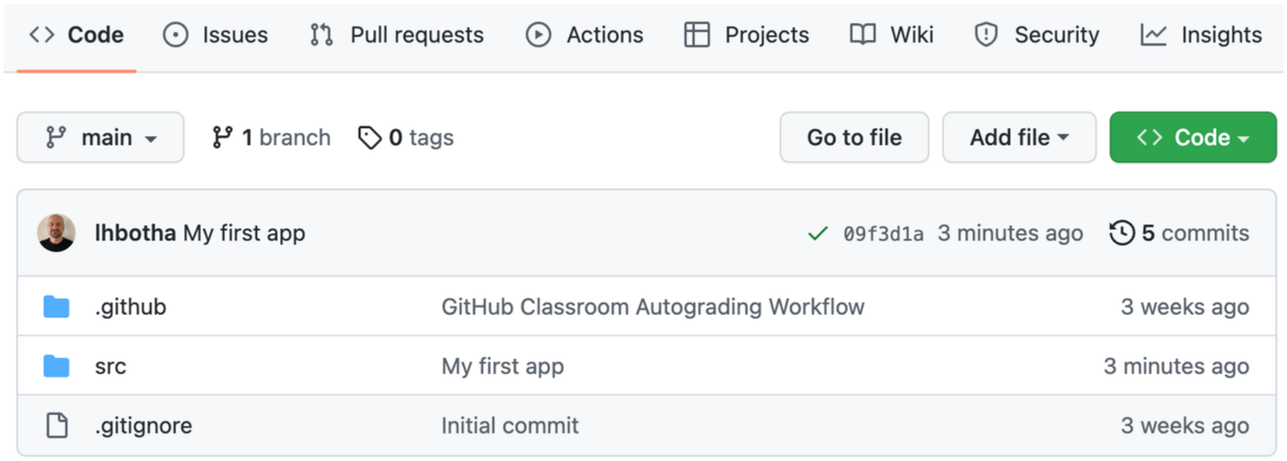
```
// Expected output
***************
* Java Rocks! *
***************
```

Next push the code to the GitHub repository:

```
// Check for changed files
❯ git status
On branch main
Your branch is up to date with 'origin/main'.
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working d
irectory)
        modified:   src/MyFirstApp.java
no changes added to commit (use "git add" and/or "git commit -
a")

// Stage changes for commit
❯ git add .

// Commit the changes to the local repository
❯ git commit -m "My first app"
[main 09f3d1a] My first app
 1 file changed, 1 insertion(+), 1 deletion(-)

// Push the changes to the GitHub
❯ git push origin
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 16 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (4/4), 333 bytes | 333.00 KiB/s, done.
Total 4 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), completed with 2 local o
bjects.
```

```
To github.com:se-5G00DL97/2023-wk02-getting-started-with-githu
b-lhbotha.git
   9afe0d3..09f3d1a  main -> main
```

Check that the file are in the remote repository and that the test past. The test is passed when there is a GREEN checkmark next to the commit id.



lhbotha My first app ✓ 09f3d1a 3 minutes ago 5 commits

.github	GitHub Classroom Autograding Workflow	3 weeks ago
src	My first app	3 minutes ago
.gitignore	Initial commit	3 weeks ago

You can mark Week 02 - 6 as done on the correct exercise and attendance :google_sheets: sheet