Programming in Java

Java source code files, layout, compilation and running

The layout of a Java program

The simplest Java program consists of:

- One file - a text file with source code in the Java programming language

This single source code file must do two things:

- Declare something called a “class”
- Inside the class, declare the so called “main method”

The source code file must be saved with a name whose first part is the same as the declared class

Show us the code!

```
public class Small {

  public static void main(String[]args) {

  }

}
```

Show us the code!

```
public class Small {  // ← Here is the class declaration

  public static void main(String[]args) { // ← main method

  }

}
```

This text is saved in a file called `Small.java`

What is a class?

A class is one of the basic building blocks in a Java program. We'll talk a lot about classes, what they are and how to use and create them further on!

For now, just learn that the so called "main method" must be declared inside one of these things called class.

This was the class declaration, by the way:

```
public class Small {
  //(other stuff, such as the main method)
}
```

OK, one file, one class?

Usually, there is one class declaration per source code file. In our extremely small program, there was only one file and one class.

If there is only one class in a file, the file name for the source code must be named after the class name (followed by “.java”). So our simple source file which declares the class Small must be saved in a text file called “Small.java”

What about that so called “main method”?

Any Java program must have a starting point where all the execution begins. This is formalized in the Java programming language as “one class declaring the main method”. OK?! But what is it, then?

It’s the part of the source code that read:

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

}
```

Do we have to write all that??!

Yes, unfortunately. Don’t worry, we’ll get into all that in later chapters!

This is the required part:

```
public static void main(String[]args)
```

(actually args may be called anything, but args is the normal name)

Just accept that for now. We’ll go through all those words in coming chapters!

In the example, what does the main method do?

The main method is confined between the { and the }

As you can see, there’s nothing in between. Consequently, this program starts in the main method (as all programs do) but since it is empty, it ends right away after doing nothing.

What’s the point?

The point is that we could show you the smallest program possible and focus on the two parts of the program: The declaration of a class, and, inside that class, the definition of the so called main method (the starting point of the program).

Exercise: Read this program out loud

First line: “Declare a class called Small”

```
public class Small {

  public static void main(String[]args) {

  }

}
```

Exercise: Read this program out loud

```
public class Small {
   Next line: “Declare the so called main method”
   public static void main(String[]args) {

   }

}
```

Pause!

We’ll continue soon!

Compiling and running the program

Let’s compile (turn the source code into something the java virtual machine can understand and execute) and run (let the java virtual machine execute the program) this program:

```
public class Small {

  public static void main(String[]args) {

  }

}
```

Compiling and running:

Open a terminal, navigate to the directory where the source code file is located, and enter the following to compile the program:

```
$ javac Small.java
```

The result is that the compiler, javac, creates a new file in the same directory, called Small.class

Now, let’s run our program (still in the same directory) as before:

```
$ java Small
```

The arguments differed!

To compile, we gave javac the argument of the source code file, Small.java

That was because the job of the compiler is to check the validity of the code according to the Java language rules (syntax, etc), and to produce a new file which the Java virtual machine can understand and run.

When we want to run the result of the compilation, we have to invoke the java virtual machine, using the “java” command. But the Java virtual machine is only interested in classes (using class names). The class with the main method was named “Small”. So, consequently, we give this class name as the argument to the java command:

```
$ java Small
```

Notes about this small program

There are a few things worth noting about this small program:

- Usually, there are instructions in the main method that kicks off the actual program execution
- Usually, the file with the class containing the main method, is located in a directory in a directory structure
- Usually, the instructions inside the main method, involves other classes defined in other files
- Usually, those other files are located in other directories according to some logical division of the files

We will look into this in the coming lecture