E.Tutorial
www.et.ethz.ch

Programming with Java Module 0

Programming in Java

Theoretical Part

Contents

Keywords

Programming Language	Source Code	Syntax
Program	Compiler	
Programming Environment	Bytecode	Semantic
Editor	Class	
Algorithm	Comments	Command

Authors:

Lukas Fässler, Barbara Scheuner

E-Mail:

et@ethz.ch

Date:

27 April 2026

Version: 1.1

Hash: 282316a

The authors try the best to provide an error free work, however there still might be some errors. The Authors are thankful for your feedback and suggestions.

1 Module overview

The development of the computer has made it possible that machines perform computing work. However, the computer on its own is not able to solve any problems: we have to prescribe an approach (processing rules). This approach is relayed to the computer in the form of a **program**. This happens in a special language: the **programming language**. Processing rules for solving a task are called **algorithms**. We demand that an algorithm terminates its task and therefore does not run endlessly if it is executed and that it generates a useful output for each input. Hence, "algorithm" is an abstract term. We can understand, for example, a cake recipe or directions given to a tourist as an algorithm. Here, however, we only consider algorithms that have been formulated in a concrete way in a programming language.

2 Writing computer programs

If two people communicate, spoken words are accompanied, for example, by facial expressions and gestures. The answer *"fine"* to the question *"how are you?"* can be interpreted in different ways depending on the intonation, for example. People have an intellect that enables them to interpret a dialog and to put it in a context. Computers do not have this capability. We have to express ourselves very precisely to communicate with a computer. The computer doesn't know what we actually meant if we expressed ourselves in the wrong way. This was a very laborious work for the first computers because the language a computer understands is not intuitive at all for people. Therefore, so-called standard languages that are closer to our natural language have been developed. To implement algorithms as a computer program, you will use such a language in this course: **Java**.

2.1 Computer programs consist of data and instructions

A **computer program** is mainly a selection of data and a sequence of **instructions** which fulfill a certain function when they are executed. Each instruction performs a certain operation, for example a calculation. For a better understanding, you can imagine a cooking recipe as mentioned above. It first contains the quantities of the ingredients (data) and then the order of the steps (instructions) which have to be executed to cook a certain meal. The basic structure of a recipe is almost always the same: first the ingredients then the individual steps.

A computer program is very similar to this. Each program also follows a basic pattern. However, we do not speak of a pattern but of the **syntax of a programming language**, i.e. of the rules that have to be adhered to for the structure of a program. As already mentioned, there are some essential differences to the steps in a recipe. The instructions have to be formulated precisely. We will not find instructions such as "season to taste" here since the computer is not able to interpret them unambiguously.

The following line shows a very simple example for the programming language Java:

```
public class HelloWorld {
  public static void main(String[] args) {
    System.out.println("Welcome to programming with Java.");
  }
}
```

Our program contains in this example:

- a basic framework consisting of a class with the name "*HelloWorld*". The name of the class has to be identical to the name of the file in which the program is stored. Our code is thus stored in the file *HelloWorld.java*. The class contains a method which is called *main method*.
- an **instruction** in the main method (the print command `System.out.println()`)
- the **data** (in this case the text `Welcome to programming with Java.`).

When executing this program now, the following line is output on the screen:

```
Welcome to programming with Java.
```

What the program executes, i.e. its meaning, is called **semantics** of the program.

2.2 Programs have to be translated

Programs in a programming language such as Java are readable and understandable for humans. As already mentioned, a computer doesn't understand them directly but only after a conversion into instructions for its processor. They are not only difficult to understand for us, but also significantly simpler than the commands of a program in a high-level language as Java. This means that a single instruction of a program results in a sequence of several processor instructions.

Therefore, the commands of the program have to be translated into instructions of the computer so that the computer is able to execute our program. Special computer programs (so-called **compilers**) are required to translate programs from a programming language into a sequence of processor instructions. Hence, the process of translating is called **compiling**.

2.2.1 Writing and executing Java programs

Programs are stored in files. We need an **editor** to be able to edit and store these files. There are a wide range of editors and development environments for Java. After writing a program, the **source code** is stored. Files containing Java source code have the extension *.java*. In the next step, the compiler translates the source code into a format called **bytecode** which is not visible for the user. It gets the extension *.class*.

3 Statements

A **statement** is the smallest executable unit of a program. As in many other programming languages, a statement ends with a **semicolon** (`;`).

Notation:

```
Statement;
```

Example:

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

3.0.1 Comments

Comments are reading aids for humans. They serve to document the source code. Comments are skipped and completely ignored by the compiler. Any number of comments can be entered. It is necessary to specify where a comment begins and where it ends. In Java, comments can be written in two ways: either as **line comments** which cannot be longer than one line (i.e. without line breaks) or as **block comments** which can contain several lines and have a sign indicating the start and the end of it.

3.0.2 How to write a line comment:

In the following example, the lines 1 and 3 are ignored by the compiler, but the 2nd line is translated.

```
// This is a comment and is ignored by the compiler.
System.out.println("Line is translated by the compiler.");
// This is a comment and is ignored by the compiler.
```

How to write a block comment: In the following example, all three lines are ignored by the compiler.

```
/* This is a comment and is ignored by the compiler
This line is also ignored by the compiler.
This line is also igored by the compiler. */
```