Java

Introduction

Tobias Hanf, Manik Khurana

17. Oktober 2021

Java-Course

Overview

Proceeding

About this course

Requirements

- You know how to use a computer
- Please bring your computer with You

Proceeding

- There will be around 14 lessons
- Each covers a topic and comes with excercises

Question

What is your programming knowledge?

- None
- Tried it once or twice
- Basic knowledge
- Advanced
- 1337 hax0r

Some resources

- You can ask your tutor
- Book: Head first Java
  `https://katalog.slub-dresden.de/id/0-1680506722`
- StackOverflow, FAQs, Online-tutorials, ...
- Official documentation
  `https://docs.oracle.com/javase/8/`
- Material-Repository
  `https://github.com/t-hanf/java-lessons`

Communication

- Write an email to
  tobias.hanf@mailbox.tu-dresden.de
  tobias.hanf@mailbox.tu-dresden.de
- Use the ”Contact Teacher”button
- Write an email to the programming mailing list (all tutors)
  programmierung@ifsr.de
- Other options, maybe Discord, Matrix, Messenger??

About Java

Pros:

- Syntax like C++
- Strongly encourages OOP
- Platform-independent (JVM)
- Very few external libraries

  $->$ Easy to use and very little to worry about

About Java

Cons:

- A lot of unnecessary features
  in the JDK
- Slower than assembly
- No multi-inheritance
- Weak generics
- Mediocre support for other programming paradigms

$->$ Neither fast, small nor geeky

Your first program

Hello World

DEMO

Creating your Working Environment

Open the Terminal

```
      mkdir myProgram
      cd myProgram
      touch Hello.java
      vim Hello.java

```

Hello World!

This is an empty JavaClass. Java Classes always start with a capital letter

```java
public class Hello {

}

```

Hello World!

This is a small program printing *Hello World!* to the console:

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

```

How to run your program

save your program by pressing ’esc’, then ’:w’ exit vim by typing ’:q’ (and hit return) then:

```
      javac Hello.java
      java Hello

```

Hello World in an IDE

DEMO

Receive a copy of IntelliJ IDEA

Visual Studio Code is a extensible code editor

- You can download VSCode at `https://code.visualstudio.com/`
- Install the Java extension ”Language Support for Java(TM) by Red Hat”

Other code editors or IDEs are also fine but you have to know how to use them.

Basics

Programming

What is programming ?

- telling a computer what to do

Programming

What is programming ?

- telling a computer what to do
- different instructions

Programming

What is programming ?

- telling a computer what to do
- different instructions
- store a date (memory)

Programming

What is programming ?

- telling a computer what to do
- different instructions
- store a date (memory)
- do something with this date (compute)

Programming

What is programming ?

- telling a computer what to do
- different instructions
- store a date (memory)
- do something with this date (compute)
- list instruction in order

Code concepts

```java
public class Hello {
    // Calculates some stuff and outputs everything on the console
    public static void main(String[] args) {
        System.out.println(9 * 23);
    }
}

```

Code concepts

```java
public class Hello {
    // Calculates some stuff and outputs everything on the console
    public static void main(String[] args) {
        int x;
        x = 9;
        int y = 23;
        int z;
        z = x * y;

        System.out.println(z);
    }
}
```

Comments

```java
public class Hello {
    // prints a "Hello World!" on your console
    public static void main(String[] args) {
        System.out.println("Hello World!");
    }
}

```

You should always comment your code.
Code is read more often than it is written.

- // single line comment
- /* comment spanning
  multiple lines */

About the Semicolon

```java
public class Hello {
    // prints a "Hello World!" on your console
    public static void main(String[] args) {
        System.out.println("Hello World!") ;
    }
}

```

Semicolons conclude all statements.
Blocks do not need a semicolon.

Blocks

```java
    public class Hello {
        // prints a "Hello World!" on your console
        public static void main(String[] args) {
            System.out.println("Hello World!");
        }
    }

```

Everything between { and } is a *block*.
Blocks may be nested.

Naming of Variables

- The names of variables can begin with any letter or underscore. Usually the name starts with small letter.
- Compound names should use CamelCase.
- Use meaningful names.

```
public class Calc {
    public static void main(String[] args) {
        int a = 0; // not very meaningful
        float myFloat = 5.3f; // also not meaningfull
        int count = 7; // quite a good name

        int rotationCount = 7; // there you go
    }
}

```

Primitive data types

Java supports some primitive data types:

boolean	a truth value (either **true** or **false**)
int	a 32 bit integer
long	a 64 bit integer
float	a 32 bit floating point number
double	a 64 bit floating point number
char	an unicode character
void	the empty type (needed in later topics)

Calculating with *int* i

```java
public class Calc {
    public static void main(String[] args) {
        int a;
        a = 7;
        System.out.println(a);
        a = 8;
        System.out.println(a);
        a = a + 2;
        System.out.println(a);
    }
}
```

Calculating with *int* ii

```java
public class Calc {
    public static void main(String[] args) {
        int a; // declare variable a
        a = 7; // assign 7 to variable a
        System.out.println(a); // prints: 7
        a = 8;
        System.out.println(a); // prints: 8
        a = a + 2;
        System.out.println(a); // prints: 10
    }
}
```

After the first assignment the variable is initialized.

Calculating with *int* iii

```java
public class Calc {
    public static void main(String[] args) {
        int a = -9;
        int b;
        b = a;
        System.out.println(a);
        System.out.println(b);
        a++;
        System.out.println(a);
    }
}
```

Calculating with *int* iv

```
public class Calc {
    public static void main(String[] args) {
        int a = -9; // declaration and assignment of a
        int b; // declaration of b
        b = a; // assignment of b
        System.out.println(a); // prints: -9
        System.out.println(b); // prints: -9
        a++; // increments a
        System.out.println(a); // prints: -8
    }
}
```

Calculating with *int* v

Some basic mathematical operations:

Addition	`a + b;`
Subtraction	`a - b;`
Multiplication	`a * b;`
Division	`a / b;`
Modulo	`a % b;`
Increment	`a++;`
Decrement	`a--;`

Calculating with *float* i

```java
public class Calc {
    public static void main(String[] args) {
        float a = 9;
        float b = 7.5f;
        System.out.println(a); // prints: 9.0
        System.out.println(b); // prints: 7.5
        System.out.println(a + b); // prints: 16.5
    }
}

```

Calculating with *float* ii

```java
public class Calc {
    public static void main(String[] args) {
        float a = 0.1f;
        float b = 0.2f;

        System.out.println(((a + b) == 0.3));
    }
}
```

Calculating with *float* iii

```
public class Calc {
    public static void main(String[] args) {
        float a = 0.1f;
        float b = 0.2f;

        System.out.println(((a + b) == 0.3)); // false
        System.out.println((a + b));
    }
}
```

Float has a limited precision.
This might lead to unexpected results!

Mixing *int* and *float*

```java
    public class Calc {
        public static void main(String[] args) {
            float a = 9.3f;
            int b = 3;
            System.out.println(a + b); // prints: 12.3
            float c = a + b;
            System.out.println(c); // prints: 12.3
        }
    }

```

Java converts from **int** to **float** by default, if necessary.
But not vice versa.

Strings

A String is not a primitive data type but an object.
We discuss objects in detail in the next section.

```java
    public class Calc {
        public static void main(String[] args) {
            String hello = "Hello World!";
            System.out.println(hello); // print: Hello World!
        }
    }

```

Concatenation

```java
public class Calc {
    public static void main(String[] args) {
        String hello = "Hello";
        String world = " World!";
        String sentence = hello + world;
        System.out.println(sentence);
        System.out.println(hello + " World!");
    }
}

```

You can concatenate Strings using the +. Both printed lines look the same.

Strings and Numbers

```java
public class Calc {
    public static void main(String[] args) {
        int factorA = 3;
        int factorB = 7;
        int product = factorA * factorB;
        String answer =
            factorA + " * " + factorB + " = " + product;
        System.out.println(answer); // prints: 3 * 7 = 21
    }
}

```

Upon concatenation, primitive types will be replaced by their current value as *String*.