Programmieren in Java

Vorlesung 01: I/O und einfache Operationen

Prof. Dr. Peter Thiemann

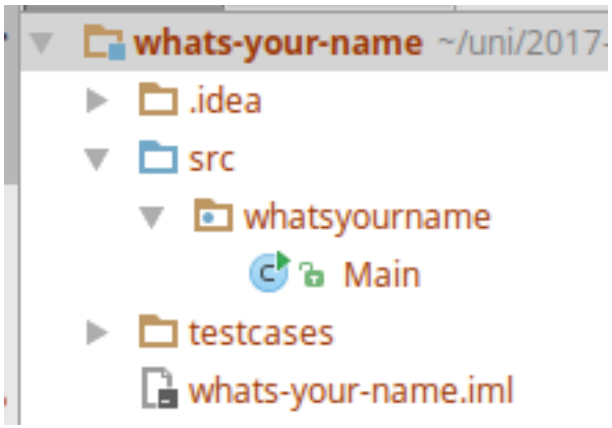
Albert-Ludwigs-Universität Freiburg, Germany

SS 2017

Aufgabe w01/2: whats-your-name

(let's have a look)

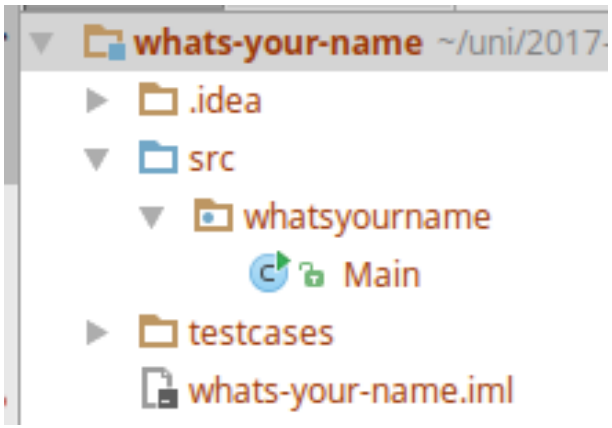
Projekte, Packages




```
Project: whats-your-name
Package: whatsyourname
```

- **Projekte** sind Ordner und Dateien mit fester Struktur
- Enthalten alle Informationen
  zum Übersetzen und Ausführen eines Programms

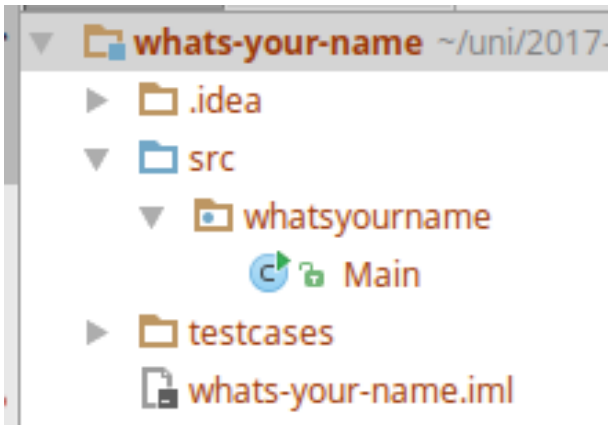
Projekte, Packages




```
Project: whats-your-name
Package: whatsyourname
```

- **Projekte** sind Ordner und Dateien mit fester Struktur
- Enthalten alle Informationen
  zum Übersetzen und Ausführen eines Programms
- **Packages** sind Teil eines Java-Projekt und ebenfalls Ordner.
- Sie dienen der Organisation von Java-Quellcode.

Projekte, Packages

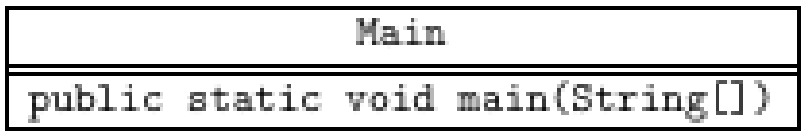


`Project: whats-your-name`
`Package: whatsyourname`

- **Projekte** sind Ordner und Dateien mit fester Struktur
- Enthalten alle Informationen
  zum Übersetzen und Ausführen eines Programms
- **Packages** sind Teil eines Java-Projekt und ebenfalls Ordner.
- Sie dienen der Organisation von Java-Quellcode.
- Wir benutzen hier immer nur eine Package.
  - Diese muss am richtigen Platz stehen `whats-your-name/src`
  - und alle Quellcode-Dateien enthalten (z.B. `Main.java`)

Implementierung von Klassen

Klassen

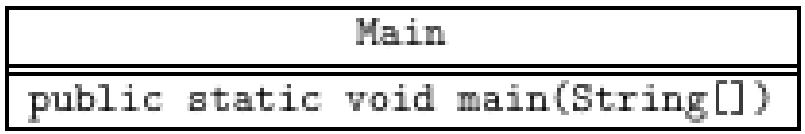


- Klassen gehören zu einer Package; diese steht am Beginn der Datei

```
package whatsyourname;
```

Implementierung von Klassen

Klassen



- Klassen gehören zu einer Package; diese steht am Beginn der Datei

```
package whatsyourname;
```

- Klassen enthalten **Methoden**
- In Java stehen **Instruktionen immer** in Methoden.

Implementierung von Klassen

Klassen

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

- Klassen gehören zu einer Package; diese steht am Beginn der Datei

```
package whatsyourname;
```

- Klassen enthalten **Methoden**
- In Java stehen **Instruktionen immer** in Methoden.
- . . . (also lassen Sie uns ein paar Instruktionen schreiben)

Implementierung von Klassen (2)

Was fällt auf?

- Alle Variablen müssen mit ihrem **Typ** deklariert werden.
- Oft sind Typen Klassen. (Ausnahme: ,,primitive Typen" wie `int`)
- Um andere Klassen zu verwenden müssen sie **importiert werden**.
- Semicolons, Klammern, ...

Kommandozeilenprogramme

- (siehe Systeme I)
- Byteströme:
  - Standardeingabe (`stdin`)
  - Standardausgabe (`stdout`)
  - Standardfehlerausgabe (`stderr`)
- Das Programm liest Daten aus **sequenziell** aus dem Eingabestrom, führt auf ihnen Berechnungen aus und schreibt sie dann auf einem Ausgabestrom.
- (. . . später: Kommandozeilenargumente)

Ein- und Ausgabe in Java

Zum lösen von `whatsyourname` wird benötigt:

- Klasse `java.io.Scanner` und `System.in`
  Ein Scanner-Objekt erlaubt lesen von Werten aus `stdin`.
  getrennt durch **Whitespace**, also Leerzeichen, Tabs, Newlines
- `System.out` und Klasse `PrintStream`
  Erlauben das Schreiben von Werten nach `stdout`
- Der Operator +
  konkateniert (führt zusammen) zwei Zeichenketten (`Strings`)

Compilierung, Classfiles

- Java Quellcode (d.h. Klassen) müssen in sog. **Classfiles** übersetzt werden, bevor sie ausgeführt werden können.
- `> javac src/whatsyourname/*.java -d classes`
- IDE sollte das automatisch tun.

Compilierung, Classfiles

- Java Quellcode (d.h. Klassen) müssen in sog. **Classfiles** übersetzt werden, bevor sie ausgeführt werden können.
- `> javac src/whatsyourname/*.java -d classes`
- IDE sollte das automatisch tun.

Ausführen:

- `> java -cp classes whatsyourname.Main`

Exkurs: APIs in Java

- `https://docs.oracle.com/javase/8/docs/api/`
- API: Application Programming Interface
- Achtung: es gibt **überladene** Methoden
  d.h. Methoden, die sich nur in den Parametertypen unterscheiden.

Primitive Datentypen in Java

- boolean, char, byte, short, int, long, float, double
- Einzelheiten siehe Tutorial über primitive Datentypen
  `https://docs.oracle.com/javase/tutorial/java/nutsandbolts/datatypes.html`
  `https://docs.oracle.com/javase/tutorial/java/nutsandbolts/operators.html`

Laufzeitfehler

- Wenn etwas schiefgeht (Division durch 0, Datei nicht gefunden) werfen Java-Methoden oft `Exceptions`.
- Die Fehler sollten verständlich sein (sonst $\rightarrow$ Forum).
- Später mehr dazu ...