# Softwareentwicklung I (IB)

# Blatt 2

Prof. Dr. Oliver Braun

Fakultät für Informatik und Mathematik
Hochschule München

Letzte Änderung: 02.11.2017 15:59

**Wollen Sie einen Schein?**

Gehen Sie in das Wiki zur Veranstaltung und überprüfen Sie

1. ob ihr GitHub-Account aufgelistet ist und
2. ob ihr GitHub-Account bei der korrekten Teilgruppe aufgelistet ist.

Wenn Ihr GitHub-Account dort nicht aufgelistet ist, werden Ihre Abgaben von uns **nicht** angesehen und Sie haben keine Möglichkeit den Schein zu bekommen.

**Abgabe aller Aufgaben auf diesem Blatt**: bis 15.11.17 10:00 Uhr durch Pushen in das zur Aufgabe gehörende GitHub-Repository.

## Und noch ein Tool: Gradle

> Wir nutzen ab sofort Gradle bis auf Weiteres für alle folgenden Aufgaben(blätter).

Für die verschiedenen Arbeiten beim Entwickeln (compilieren, ausführen, mit Checkstyle überprüfen, Tests durchführen, …) und das automatische Herunterladen von Libraries oder anderen Tools (wie z.B. Checkstyle), gibt es eine Reihe von sogenannten *Build-Werkzeugen*. In der Java-Welt sind die verbreitetsten Ant, Maven und Gradle. Den *modernsten* Ansatz verfolgt dabei Gradle.

Ein sehr angenehmer Aspekt von Gradle ist, dass es nicht installiert werden muss, sondern mit einem einfachen *Wrapper* automatisch herunter geladen und installiert wird. Sie finden dieses in den Repositories ab jetzt.

Statt nun extra `javac`, `java` und `checkstyle` zu starten, nutzen Sie nur noch das Kommando `gradlew` unter Windows bzw. `./gradlew` unter macOS, Linux oder einem anderen Unix.

Compilieren Sie ab sofort mit dem Kommando

```
gradlew build
```

bzw.

```
./gradlew build
```

Das führt neben dem Compilieren auch gleich die Checkstyle-Überprüfung durch. Wenn Sie nur Compilieren ohne Checkstyle wollen, nutzen Sie `gradlew compileJava` bzw. `./gradlew compileJava`. Für das Überprüfen mit Checkstyle nutzen Sie `gradlew check` bzw. `./gradlew check`.

`build`, `compile` und `check` sind übrigens **Tasks** in der Gradle-Welt. Mit dem Task `clean` können Sie alles was Gradle gebaut hat, wieder löschen. Mit dem Task `run` können Sie das Programm ausführen. Die Tasks bauen zum Teil aufeinander auf, d.h. mit einem `gradlew run` bzw. `./gradlew run` machen Sie automatisch ein `build` vor dem `run`.

Einen kleinen Nachteil hat Gradles `run`-Task: Auf der Kommandozeile können Sie einem Programm `Hello` einen Kommandozeilenparameter direkt übergeben in dem Sie ihn dahinter angeben, z.B.

```
java Hello Hans
```

Bei Gradle können Sie `Hans` nicht einfach dahinter schreiben, z.B.

```
./gradlew run Hans // funktioniert nicht wie gedacht
```

Sie müssen den `Hans` (oder was auch immer Sie übergeben wollen) als Schlüsselwertpaar übergeben. Ich habe die Projekte jeweils so konfiguriert, dass der Schlüssel `exec.args` ist, also z.B.

```
./gradlew run -Dexec.args="Hans"
```

Mehrere Argumente werden einfach durch Leerzeichen getrennt, also z.B.

```
java Bmi Erna 123 33
```

bzw.

```
./gradlew run -Dexec.args="Erna 123 33"
```

Wenn Ihnen die Version mit `java` besser gefällt, so öffnen Sie zum Ausführen doch einfach ein zweites Terminal/cmd-Fenster und navigieren Sie innerhalb der Projektes in das Verzeichnis `build` so tief hinein, bis Sie dort sind wo die compilierte Java-Klasse liegt. Sie finden Sie natürlich erst, wenn Sie einmal erfolgreich mit `compileJava` compiliert haben.

Ab diesem Blatt sind auch Tests enthalten, die Sie selbst lokal ausführen können. Der Jenkins macht das gleiche.

# Aufgabe 1

Das Repository für diese Aufgabe bekommen Sie unter https://classroom.github.com/a/UL8EkJAE.

In dem Repository finden Sie ein Gradle-Projekt. Für uns ist dabei im Moment nur das Verzeichnis `src/main/java` bzw. `src\main\java` interessant. In diesem liegt die Java-Datei, die Sie für diese Aufgabe bearbeiten sollen. Für die folgenden Aufgaben wird das genauso sein.

Um die Gradle-Kommandos ausführen zu können, müssen Sie im Terminal bzw. im cmd-Fenster ganz oben im Projektverzeichnis sein.

Schreiben Sie ein Programm mit dem Namen `Bmi` das den BMI berechnet. Der Name, das Gewicht in Kilogramm und die Größe in Zentimeter sollen als Kommandozeilenparameter übergeben werden. Ein Beispielaufruf könnte folgendermaßen aussehen:

```
java Bmi Erna 152 51
BMI: 22,07
```

bzw. gestartet durch

```
./gradlew run -Dexec.args="Erna 152 51"
```

wobei `152` die Größe in Zentimeter und `51` das Gewicht in Kilogramm sein soll.

Achtung: Je nach Einstellung kann es sein, dass ein Dezimalkomma oder ein Dezimalpunkt ausgegeben wird. Intern nutzt Java immer den Dezimalpunkt, wie im englischsprachigen Raum üblich.

Die Kommandozeilenparameter werden in einem Array mit dem Namen `args` übergeben. Den ersten Parameter, im Beispiel `Erna`, bekommen Sie mit dem Ausdruck `args[0]`, den zweiten mit `args[1]` und so weiter.

Ein Kommandozeilenparameter hat immer den Typ `String`. Sollten Sie einen `int` bzw. `double` benötigen, können Sie den `String` mit Hilfe der Funktion `Integer.parseInt()` bzw. `Double.parseDouble()` umwandeln.

Der BMI berechnet sich nach der Formel:

$BMI = \frac{mass(kg)}{(height(m))^2}$

Nutzen Sie zur Speicherung Variablen, denen Sie Namen geben, die den Konventionen entsprechen. Definieren Sie soviele Variablen wie möglich als `final`.

Gestalten Sie die Ausgabe so, dass sie **exakt** so aussieht wie im Beispiel. Nutzen Sie zur Ausgabe `System.out.printf()`. Diese Funktion hat als erstes Argument einen Format-String und als weitere Elemente beliebige Variablen oder Werte. Mit Hilfe von Platzhaltern werden die Variablen in die Ausgabe platziert. Informationen über die Formatsyntax finden Sie unter http://docs.oracle.com/javase/8/docs/api/java/util/Formatter.html#syntax.

Achten Sie insbesondere auch darauf, dass Ausgabe des exakten BMI auf 2 Stellen nach dem Komma erfolgt.

## Aufgabe 2

Das Repository für diese Aufgabe bekommen Sie unter https://classroom.github.com/a/pntOJxxR.

Ein Satellit funkt Zeitspannen als "Anzahl Sekunden" zur Erde. Schreiben Sie ein Programm `SatelliteTime`, das einen Sekundenbetrag auf der Kommandozeile akzeptiert und die Zeitspanne in der Form

```
d Tage h Stunden m Minuten und s Sekunden
```

wieder ausgibt, wobei gilt

`d` = Anzahl Tage,
`h` = Anzahl Stunden im Bereich 0 bis 23,
`m` = Anzahl Minuten im Bereich 0 bis 59,
`s` = Anzahl Sekunden im Bereich 0 bis 59.

Testen Sie das Programm beispielsweise mit dem folgenden Aufruf.

```
java SatelliteTime 10000
0 Tage 2 Stunden 46 Minuten und 40 Sekunden
```

bzw. mit

```
./gradlew run -Dexec.args="10000"
```

Nutzen Sie zur Berechnung den Modulo-Operator und speichern Sie die einzelnen Teile jeweils in einer Variablen. Nutzen Sie zur Ausgabe einen einzigen `System.out.printf()`-Aufruf. Definieren Sie soviele Variablen wie möglich als `final`. Letzteres sollten Sie, auch wenn es nicht explizit in der Aufgabe steht, in Zukunft **immer** machen.

## Aufgabe 3

Das Repository für diese Aufgabe bekommen Sie unter https://classroom.github.com/a/ASmKUeA4.

Die API[1]-Dokumentation der Klasse `Math` finden Sie unter http://docs.oracle.com/javase/8/docs/api/java/lang/Math.html.

Das, bereits im Repository vorhandene, Programm `MyMath` soll wie folgt funktionieren:

```
$ java MyMath 1.23
sin(1.230000) = 0.942489
cos(1.230000) = 0.334238
tan(1.230000) cos(1.230000) = 0.942489
e^1.230000 = 3.421230
log(e^1.230000) = 1.230000
```

bzw. mit

```
./gradlew run -Dexec.args="1.23"
```

Leider haben sich in den Quellcode einige Fehler eingeschlichen und er ist etwas unschön im Layout:

```
class MyMath { public static main(String[] args) { final input
= Double.parseDouble(args[0]); System.out.printf("sin(%f)
= %f\n", input, Math.sinus(input)); System.out.printf("cos(%f)
= %f\n", input Math.cos(input)); System.out.printf("tangens(%f)
cosinus(%f) = %f\n", input, Math.tan(input) Math.cos(input));
System.out.printf("e hoch %f = %f\n, input, Math.exp(input));
System.out.print("log(e hoch %f) = %f\n", input,
Math.log(Math.exp(input))); } }
```

Formatieren Sie den Quellcode und berichtigen Sie alle Fehler, so dass das Programm exakt wie oben dargestellt, funktioniert. Achtung: Unter Umständen müssen Sie auch etwas in der Ausgabe anpassen, so dass sie absolut identisch ist.

[1] Application Programming Interface