

## **Tutorium Programmieren**

Tut Nr.3: Checkstyle, Kontrollstrukturen Michael Friedrich | 12. / 14.11.2013

INSTITUT FÜR THEORETISCHE INFORMATIK



## **Outline/Gliederung**



- Checkstyle
- 2 Kontrollstrukturen
- 3 Getter / Setter
- 4 Einschub
- Tutoriumsaufgabe
- 6 Ende

Checkstyle



Kontrollstrukturen

Einschub

## Checkstyle



- Beinhaltet Konventionen, um Code einheitlich und damit leserlich zu machen
  - Wir bewegen uns im Spielraum, der uns der Compiler bietet
- Daher Pflicht ab diesem Übungsblatt
  - in 2 Stufen, vorerst nur Checkstyle 1
- Daher Pflicht ab diesem Übungsblatt
  - in 2 Stufen, vorerst nur Checkstyle 1
  - Sonst: Abzug!



Einschub

Ende

## Was beinhaltet Checkstyle?



- Zunächst, wie schon letzte Woche erwähnt
  - Whitespace verpflichtend um
    - = (Zuweisung)
    - +, + =, −, − =
    - **\ \{, \}**
    - ==,!=, <, <=, >, >=
    - **&&**,||
    - if, else, for, while, return
    - ...

## Was beinhaltet Checkstyle?



- Kein Whitespace nach
  - Whitespace verpflichtend um
    - (Bitweises Komplement)
    - ++ (Prefix-Inkrementierung, z.B. ++i;)
    - (Prefix-Dekrementierung, z.B. –i;)
    - . (Punkt)
    - (Unäres Minus, z.B. -5)
    - + (Unäres Plus, z.B. +4)
    - ...

Kontrollstrukturen



12. / 14.11.2013

## Wie benutzt man Checkstyle?



- Kein Whitespace nach
  - Whitespace verpflichtend um
    - (Bitweises Komplement)
    - ++ (Prefix-Inkrementierung, z.B. ++i;)
    - (Prefix-Dekrementierung, z.B. –i;)
    - . (Punkt)
    - (Unäres Minus, z.B. -5)
    - + (Unäres Plus, z.B. +4)
    - ...

Kontrollstrukturen



## Wie benutzt man Checkstyle?



- Spätestens der Praktomat prüft ob ihr euch an die Richtlinien hält
  - nur konforme Lösungen werden angenommen (ca. 3ÜB)
  - Um einer stressigen Abgabe vorzubeugen, solltet ihr Checkstyle in eure IDE einbinden
  - .xml Dateien auf http://baldur.iti.uka.de/programmieren/

 $\rightarrow$  Demo für Eclipse



12. / 14.11.2013

Ende



#### Wofür?

Kontrollstrukturen nutzen wir, um den linearen Programmfluss zu steuern.

- ermöglicht Fallunterscheidungen if
- verhindert redundanten Code while



Checkstyle



## if-then-else - Verzweigungen

```
if (boolean1) {
//code to run if boolean1 is true
} elseif (boolean2) {
//code to run if boolean2 is true AND boolean1 is false
} else {
//code to run otherwise
}
```

Hinweis: boolean kann auch ein zusammengesetzter Ausdruck sein (Operatoren dazu siehe letztes Tut)



Kontrollstrukturen



#### while-Schleifen

```
while (boolean) {
//code to run while boolean is true
```

→ Bedingung vor erstem Schleifendurchlauf erfüllt

#### do-while - Schleifen

```
do{
//code to run while boolean is true
} while (boolean)
```

Kontrollstrukturen

→ Was ist hier anders?





#### for-Schleifen

```
for (var ; condition; var modification) {
//code to run in loop
```

## Beispiel:

```
for (int i = 0; i < n; i++) {
  gool
```

Alternativ kann "i" auch vorher initialisiert werden, die Zuweisung muss aber hier geschehen. Vorteile?





#### switch

```
switch (var) {
case 1:
// code if var is 1
break;
case 2:
case 3:
// code if var is 2 or 3
break;
default:
// code to run if none of the above conditions hold
break;
}
```

### Switch über primitive Datentypen möglich

Wie letztes Mal erwähnt ist String ein Sonderfall - Seit Java 1.7 auch (offiziell) möglich



12. / 14.11.2013

Checkstyle

## Getter / Setter



 Methoden, die kontrollierten Zugriff auf Attribute ermöglichen, statt Zugriff mit Punkt

#### Getter

```
returntype getVar(){
return this.var;
}
```

#### Setter

Checkstyle

```
void setVar(varType param) {
this.var = param;
}
```



### **Getter / Setter**



- Nutzen?
  - Prinzip der Kapselung

Modifier	Class	Package	Subclass	World
public	✓	✓	✓	✓
protected	✓	✓	✓	×
no modifier*	✓	✓	×	×
private	<b>4</b>	×	×	×



Kontrollstrukturen

Checkstyle

Einschub

### final und static



Es gibt 2 weitere Modifikatoren

#### final - definiert eine Konstante

final int MAX\_VALUE;

#### static - definiert eine Klassenvariable

final static int NUMBER\_OF\_WHEELS;

Kontrollstrukturen

Klassenvariable bedeutet, dass dieses Attribut nicht an ein bestimmtes Objekt gebunden ist.



# Tutoriumsaufgabe



 Wir wollen das bisher Gelernte umsetzen und uns einen eigenen Datentyp bauen
 Die rationalen Zahlen

 $\rightarrow$  Brainstorming



## **Tutoriumsaufgabe**



- Was brauchen wir also alles?
  - Ausgabe (toString)
  - Addition
  - Substraktion
  - Beides mit int (Hinweis: Überladen)
  - Kapselung

Kontrollstrukturen

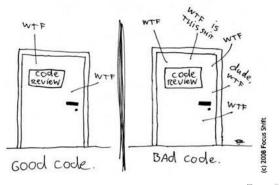


## Das war's für heute



#### Danke für eure Aufmerksamkeit!

The ONLY VACID MEASUREMENT OF Code QUALITY: WTFS/minute





12. / 14.11.2013

Checkstyle