Java Code Style

**Namen:** Chiem Stevens, Joram van Driel, Mark Baaij, Pascal de Bruijn, Matthijs Vrolijks, Kenley Strik  
**Groep:** 23TIVT1B2  
**Datum:** 08-11-2018  
**Tutor:** Paul de Mast  
**Senior:** Peter Kailuhu

Inhoudsopgaven

[Inleiding 2](#_Toc529449605)

[Declaraties - naam conventies 2](#_Toc529449606)

[Algemeen 2](#_Toc529449607)

[Variabele en Objecten namen 2](#_Toc529449608)

[Methoden en Klassen namen 2](#_Toc529449609)

[Voorbeeld 2](#_Toc529449610)

[Uitlijning - conventies 3](#_Toc529449611)

[Algemeen 3](#_Toc529449612)

[Spaties en Tabs 3](#_Toc529449613)

[Witregels en accolades 3](#_Toc529449614)

[Commentaar 4](#_Toc529449615)

[Voorbeeld 5](#_Toc529449616)

# Inleiding

In dit document worden de conventies met betrekking tot de Java source code beschreven. Er worden onderwerpen behandeld zoals de taal van de namen in de code, uitlijning, naamgeving en andere zaken die met de opmaak van de code te maken hebben.

# Declaraties - naam conventies

## Algemeen

Alle namen in de code zullen in het Engels worden geschreven en moeten goed beschrijven wat ze betekenen. De namen moeten lang worden uitgeschreven geen afkortingen, dus bijvoorbeeld: **personID** in plaats van **perID.** Dit mag alleen als met de afkorting nog steeds erg duidelijk is wat de variabele inhoud zoals bijvoorbeeld: **maxAmount** in plaatst van **maximumAmount** en alleen als er meer dan drie karakters kunnen worden weggelaten. De namen mogen alleen met letters worden beschreven en mogen geen speciale tekens bevatten.

## Variabele en Objecten namen

Alle variabelen en objecten namen die geen constante of enumeratie waarde omschrijft wordt geschreven in camelCase, dus bijvoorbeeld: **carTire** in plaatst van **cartire** of **CarTire.** Constante waarden worden daarentegen in PascalCase geschreven, dus bijvoorbeeld: **CarTire** in plaats van **cartire** of **carTire**. En enumeratie waarden worden volledig in hoofdletters geschreven, dus bijvoorbeeld: **CARTIRE** in plaats van **cartire**, **carTire** of **CarTire**.

## Methoden en Klassen namen

Alle methode namen worden geschreven in camelCase, dus bijvoorbeeld: **getMaximum** in plaats van **GetMaximum** en moet duidelijk aangeven wat de methode doet. Voor methodes geld dus vaak als naamgeving (werkwoord + zelfstandig naamwoord) (**get** + **Maximum**). Alle klassen namen worden geschreven in CamelCase, dus bijvoorbeeld: **Person** in plaats van **person**.

## Voorbeeld

class Person  
{  
 private String name;  
 private int personAge;  
 private final int MaxAge;  
  
 public Person(String name, int personAge)  
 {  
 this.name = name;  
 this.personAge = personAge;  
 this.MaxAge = 100;  
 }  
  
 private enum Directions  
 {  
 *NORTH*,  
 *EAST*,  
 *SOUTH*,  
 *WEST*;  
 }  
}

# Uitlijning - conventies

## Algemeen

In elke scope moet de code met één tab naar recht worden verplaatst zodat huidelijk blijft in welke scope de code zich bevind.

## Spaties en Tabs

Bij alle operatoren behalve unary operatoren moet er standaard een spatie voor en achter de operator zijn. Het enigste geval waar dit niet van toepassing is, is bij de **++** of **--** operatoren. Bijvoorbeeld: **count++** en **count--**. Bij unary operatoren worden altijd met de operand samen geschreven zonder spaties.

Bij komma’s en puntkomma’s moet er altijd een spatie achter worden geplaatst maar nooit ervoor behalve bij puntkomma’s die aan het einde van de regel staan, dus bijvoorbeeld:   
**getMaximum(number1, number2);** in plaats van **getMaximum(number1 ,number2) ;** of **getMaximum(number1,number2) ;**.

Binnen de haken van de methode, loop, if of else if declaraties moet voor de eerste en achter de laatste parameter nooit een spatie staan, dus bijvoorbeeld:   
**getMinimum(number1, number2);** in plaats van **getMinimum ( number1, number2 );  
for(int i = 0, i < 10; i++)** in plaats van **for( int i = 0, i < 10; i++ )  
if(number1 <= number2)** in plaats van **if( number1 <= number2 )  
else if(number1 <= number2)** in plaats van **else if ( number1 <= number2 )**

## Witregels en accolades

Accolades moeten altijd op dezelfde kolom worden gezet waardoor de begin accolade altijd exact boven de eind accolade staat, dus bijvoorbeeld:  
**if(number1 >= number2)** in plaats van **if(number1 >= number2){  
{  
 }  
}**

Na elke scope van een methode moet er als er een volgende methode volgt altijd een witregel worden geplaats. De inhoud van elke scope moet altijd worden gezet tussen twee accolades. Hieronder staan voorbeelden van de verschillende scopes. Bij de do-while loop moet de while volgens syntax conventie achter de eind accolade worden geplaatst.

## Commentaar

Commentaar moet zo dicht mogelijk bij de bijhorende code worden gezet. Bij elk soort loop methode of klassen moet het commentaar er altijd direct boven staan zonder een witregel ertussen, dus bijvooreeld:  
**//Give back the largest of the two number**  
**public int getMaximum(number1,number2)  
{  
  
}**

Verder wordt het plaatsen van commentaar gedaan achter de code als deze uit één regel kan worden opgebouwd. Als dit meer dan één regel omvat wordt het commentaar direct boven het stuk code gezet zonder een witregel ertussen en wordt er een witregel boven het stuk commentaar geplaatst, dus bijvooreeld:  
**int number = 10; //Sets the number variable to 10**of **//Sets the number variable to 10  
int number = 10;**

## Voorbeeld

//Klasse:  
class Person()  
{  
 //Methode:  
 public int getMaximum(int number1, int number2)  
 {  
 //For-loop:  
 for(int i = 0; i < 10; i++)  
 {  
  
 }

//Foreach-loop:  
 for(int i : numbers)  
 {  
  
 }

//While-loop:  
 while(number1 > number2)  
 {  
  
 }

//DoWhile-loop:  
 do  
 {  
  
 } while(number1 > number2);  
   
 //If-statement:  
 if(number1 >= number2)  
 {  
 number1 += number2;  
 }  
 else if(number1 <= number2)  
 {  
 number1 -= number2;  
 }  
 else  
 {  
 number1 \*= number2;  
 }  
 }  
}