



Java Essentials

Hoofdstuk 3

Eenvoudige klassen: Deel1

DE HOGESCHOOL MET HET NETWERK

Hogeschool PXL – Elfde-Liniestraat 24 – B-3500 Hasselt
www.pxl.be - www.pxl.be/facebook



Inhoud

1. De klasse Math
2. De klasse Random
3. De klasse String
4. De klasse Character
5. De klasse StringBuilder



Doel:

- eenvoudige klassen uit de Java API gebruiken
- Java API documentatie leren gebruiken



1. De klasse Math

= utility-klasse

= klasse die hulpmethoden aanreikt

→ Bevat enkel klassevariabelen en klassemethoden

→ Je kan er geen objecten van maken



- De klasse Math bevat 2 klassevariabelen:
Math.PI het getal π
Math.E het getal e
- De klasse Math bevat groot aantal klassemethoden voor wiskundige berekeningen:
 - Goniometrische functies
 - Machtsverheffing
 - Worteltrekking
 - Exponentiële functies
 - ...

Voorbeeld

```
cosinus = Math.cos(angle);
```



Enkele nuttige methoden

Methode	Omschrijving van de methode
<code>static int abs(int x)</code> <code>static double abs(double x)</code>	berekent de absolute waarde van het argument
<code>static double sqrt(double x)</code>	berekent de vierkantswortel van het argument
<code>static double pow(double x, double y)</code>	berekent x^y
<code>static int round(float x)</code> <code>static long round(double x)</code>	berekent het dichtst bijzijnde geheel getal

Voorbeelden

1. Verbeter, indien nodig:

Programma 1

```
int x;  
double y;  
y = Math.sqrt(2);  
x = Math.sqrt(4);
```

Programma 2

```
int x;  
int y;  
x = Math.pow(2, 3);  
y = Math.round(4.7);
```

2. Wat is het verschil tussen `Math.abs(-1)` en `Math.abs(-1.0)`?



Opdracht 1

Werken met de klasse Math

Schrijf een programma om x^3 te berekenen. Hierbij is x geheel en het resultaat moet ook geheel zijn. x moet via het toetsenbord ingegeven worden.



Opdracht 2

Schrijf een programma om de diameter van een cirkel te berekenen als de oppervlakte van de cirkel gegeven is.

Formule oppervlakte cirkel: $\pi \cdot r^2$

r = de straal van de cirkel

Rond het resultaat af op 2 decimalen.

Opdracht 3

Herneem de opdracht met de klasse Klas uit hoofdstuk2.
Rond het gemiddeld aantal studenten per klas af op 1 decimaal.



2. De klasse Random

De klasse *Random* bevindt zich in het pakket *java.util*
➔ volledige naam van de klasse = *java.util.Random*.

Creatie van een Random object

```
java.util.Random rand = new java.util.Random();
```

genereert random getallen

= aanroep van een **constructor**



```
java.util.Random rand = new java.util.Random();
```

of

```
import java.util.Random;  
public class Oef1 {  
    ...  
    Random rand = new Random();  
    ...  
}
```

Volledige klassenaam
importeren (om niet telkens
java.util te moeten herhalen).



Enkele nuttige methoden

Methode	Omschrijving van de methode
<code>int nextInt()</code>	genereert een willekeurig geheel getal
<code>int nextInt(int bound)</code>	genereert een willekeurig geheel getal ≥ 0 en $< \text{bound}$
<code>double nextDouble()</code>	genereert een willekeurig floating point getal ≥ 0 en < 1

Voorbeeld

```
int value = rand.nextInt(5);
```

Bovengrens niet inclusief

→ 0, 1, 2, 3 of 4

Opdracht 4

Werken met de klasse Random

- a. Maak een programma dat een random geheel getal genereert. Druk het getal af op het scherm.
- b. Pas het programma aan zodanig dat er 20 gehele getallen gegenereerd worden tussen 0 en 10 (10 incl.). Druk ze af op het scherm.
- c. Pas het programma aan zodanig dat er 10 getallen gegenereerd worden tussen 20 en 30 (30 incl.).
- d. Genereer een willekeurige kleine letter.



3. De klasse String

Behoort tot het pakket *java.lang* (wordt standaard geïmporteerd)

Creatie van String objecten

```
String text = new String("Hello World");
```



Vermits Java elke string automatisch omzet in een object van de klasse *String*, kan men een string ook als volgt declareren.

```
String text = "Hello World";
```



Bij de creatie van het object wordt een initiële waarde toegekend die nadien **niet meer kan veranderd worden!** (strings zijn immutable)

```
String text = new String("Hello World");
```

yyy

text

HelloWorld

```
text = "nieuwe tekst";
```

nieuwe tekst



Opdracht 5

Java API Documentation

- Ga in JavaDoc naar de klasse *String*.
- Kijk of je de constructoren vindt.
- Zoek de methode *toUpperCase()*. Wat doet deze methode? Test onderstaande code in IntelliJ.

```
String str = "abc";  
str.toUpperCase();  
System.out.println(str);
```

Waarom werkt deze code niet? Pas de code aan zodanig dat de code werkt.



Enkele nuttige methoden

Methode	Omschrijving van de methode
<code>int length()</code>	levert het aantal karakters van de huidige string
<code>char charAt(int index)</code>	levert het karakter op dat zich op de positie index bevindt. Opgepast men begint te tellen vanaf 0
<code>boolean equals(String s)</code>	bepaalt of de huidige string gelijk is aan s
<code>int compareTo(String s)</code>	vergelijkt (alfabetisch) de huidige string met s. Returnwaarde: neg voor <, nul voor =, pos voor >
<code>boolean equalsIgnoreCase(String s)</code>	bepaalt of de huidige string gelijk is aan s, hierbij worden hoofdletters en kleine letters als identiek beschouwd.
<code>String toLowerCase()</code>	levert de string op gelijk aan de huidige, maar waarin alle hoofdletters vervangen zijn door kleine letters
<code>String toUpperCase()</code>	levert de string op gelijk aan de huidige, maar waarin alle kleine letters vervangen zijn door hoofdletters



Methode	Omschrijving van de methode
boolean startsWith(String s)	bepaalt of de huidige string begint met de string s
boolean endsWith(String s)	bepaalt of de huidige string eindigt op de string s
String substring(int beginIndex, int endIndex)	levert een deel van de huidige string op met de gegeven begin- en eindpositie (exclusief)
String substring(int beginIndex)	levert een deel van de huidige string op met de gegeven beginpositie
int indexOf(String s)	levert de (eerste) positie op waar s binnen de huidige string voorkomt of -1 als hij daar niet voorkomt.
int indexOf(String s, int vanAf)	idem, maar er wordt pas gezocht vanaf de positie vanAf
int lastIndexOf(String s)	zoals overeenkomstige indexOf maar het zoeken gebeurt van achter naar voor.
String replace(char oud, char nieuw)	levert een nieuwe string op waarbij in de huidige string het karakter oud veranderd is in karakter nieuw
String trim()	levert de string op die verkregen wordt door in de huidige alle spaties aan het begin en het eind weg te laten.



Voorbeeld

Karakter op een welbepaalde plaats opvragen

```
String text = "Hello World";  
char c = text.charAt(8);  
System.out.println(c); // r
```

= positie 0!!!

Belangrijke opmerking

Als je de inhoud van 2 strings wil vergelijken, moet je de methode ***equals()*** gebruiken.

```
String s1 = "abcdef";  
String s2 = new String("abcdef");  
boolean eq = s1.equals(s2);  
System.out.println(eq); // true
```

Strings vergelijken met `==` is uit den boze tenzij men zeer bewust de referentie wil vergelijken.



Opdracht 6

Werken met String-objecten

Vervolledig onderstaande + leg uit wat er gebeurt.

str stelt een string voor

Python	Java	uitleg
str[3]	str.charAt(3)	teken op positie 3 opvragen
str[2:5]	str.substring(2, 5)	substring van de 2 ^{de} tot aan de 5 ^{de} positie
len(str)	str.length()	lengte van str
str.find('x')	str.indexOf('x')	plaats van de eerste x in de str
str.split()	str.split(" ")	split de tekst op basis van een spatie naar een list/array van strings
str.split(',')	str.split(",")	split de tekst op een , naar een list/array van strings
str + str	str + str str.concat(str)	samenvoegen van 2 strings tot 1 string
str.strip()	str.trim()	spaties vooraan en achteraan verwijderen

Opdracht 7

Werken met String-objecten

- a. Maak een programma met een regel tekst. Druk de tekst en de lengte van de tekst af.
- b. Vervang de letters 'a' door de letters 'o' en druk het resultaat af.
- c. Druk het aantal letters 'e' af?
- d. Maak 2 strings met verschillende inhoud en ga na of de inhoud gelijk is. Druk false of true af.
- e. Schrijf een programma om de middelste of de 2 middelste letters van een woord in hoofdletters te zetten.



4. De klasse Character

= klasse die hulpmethoden aanreikt om te werken met karakters

→ Bevat hoofdzakelijk klassevariabelen en klassemethoden

→ De klasse *Character* bevindt zich in het pakket *java.lang* . De volledige naam van de klasse = *java.lang.Character*



Enkele nuttige methoden

Methode	Omschrijving van de methode
<code>static boolean isDigit(char c)</code>	geeft de waarde true als het karakter een cijfer is en false als dit niet zo is
<code>static boolean isLetter(char c)</code>	geeft de waarde true als het karakter een letter is en false als dit niet zo is
<code>static boolean isUpperCase(char c)</code>	geeft de waarde true als het karakter een hoofdletter is en false als dit niet zo is
<code>static boolean isLowerCase(char c)</code>	geeft de waarde true als het karakter een kleine letter is en false als dit niet zo is
<code>static boolean isLetterOrDigit (char c)</code>	geeft de waarde true als het karakter een cijfer of een letter is en false als dit niet zo is
<code>static boolean isWhitespace(char c)</code>	geeft de waarde true als het karakter whitespace is Dit zijn o.a. de karakters spatie (' '), tab ('\t')
<code>static char toLowerCase(char c)</code>	maakt van het karakter c een kleine letter
<code>static char toUpperCase(char c)</code>	maakt van een karakter c een hoofdletter



Voorbeeld

Karakter naar kleine letter converteren

```
char c1 = 'A';  
char c2 = Character.toLowerCase(c1);
```



Opdracht 8

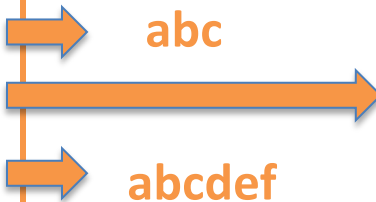
Werken met de klasse Character

- a. Schrijf een programma om te controleren of een artikelcode juist is. De artikelcode moet bestaan uit 2 letters gevolgd door 2 cijfers. Druk een gepaste foutmelding af.
- b. Plaats de eerste letter van de artikelcode in hoofdletters en druk de nieuwe artikelcode af.



5. De klasse StringBuilder

```
String s1 = "abc";  
System.out.println(s1);  
s1 += "def";  
System.out.println(s1);
```



s1 blijft ongewijzigd
(strings zijn immutable)



hier wordt een nieuw
object aangemaakt en
het oude object wordt
weggegooid!

Stel je voor dat `s1 += "def";` in een for-lus zou staan die 100x doorlopen wordt, dan wordt er 100x een nieuw object aangemaakt en 100x een object weggegooid → = CPU-tijd die verloren gaat → oplossing = StringBuilder

```
StringBuilder s1 = new StringBuilder("abc");  
s1.append("def");  
System.out.println(s1);
```



→ abcdef

hier “bouwt” (StringBuilder) men een string!!! De bestaande string “abc” wordt behouden en “def” wordt toegevoegd.

Indien je `s1.append("def");` in een lus zet die 100x doorlopen wordt, dan wordt er telkens gebruik gemaakt van dat ene object (er worden geen 100 objecten aangemaakt en vernietigd).

Om een StringBuilder object terug om te zetten naar een String gebruik je de toString() methode.

```
StringBuilder text = new StringBuilder("Hello World");  
text.append("met uitbreiding");  
String tekst = text.toString();
```

De klasse StringBuilder behoort tot het pakket *java.lang* (wordt standaard geïmporteerd).

Verdere methoden van de klasse StringBuilder vind je terug in de java API documentatie.



Opdracht 9

Werken met de klasse `StringBuilder`

Herneem opdracht8b:

plaats de eerste letter van de artikelcode in hoofdletters, de tweede letter in kleine letters en druk de nieuwe artikelcode af.

Maak hierbij gebruik van een `StringBuilder` om de code te veranderen.

