# collections

### Voorbeeld

### Te gebruiken bij:

-

### Eigenschappen:

-

### Info

#### If statments

Een if-statement wordt gebruikt om een bepaalde codeblok uit te voeren als een bepaalde voorwaarde waar is.

Syntax in tekst: "Als (voorwaarde) dan voer de volgende code uit."

Voorbeeld in Java-code:

java

int getal = 10;

if (getal > 0) {

System.out.println("Het getal is positief.");

} else if (getal < 0) {

System.out.println("Het getal is negatief.");

} else {

System.out.println("Het getal is nul.");

}

#### Switch-statements:

Een switch-statement wordt gebruikt om verschillende codeblokken uit te voeren op basis van de waarde van een expressie.

Syntax in tekst: "Schakel over (expressie) en voer codeblokken uit op basis van de overeenkomstige waarden."

Voorbeeld in Java-code:

java

int dag = 1;

String dagNaam;

switch (dag) {

case 1:

dagNaam = "Maandag";

break;

case 2:

dagNaam = "Dinsdag";

break;

case 3:

dagNaam = "Woensdag";

break;

// ...

default:

dagNaam = "Ongeldige dag";

break;

}

System.out.println("Dag: " + dagNaam);

#### Loop-statements:

##### for-loop

Loop-statements worden gebruikt om een bepaald stuk code herhaaldelijk uit te voeren.

Er zijn verschillende soorten loop-statements in Java, zoals de for-loop, de while-loop en de do-while-loop.

Syntax in tekst: "Herhaal de volgende code zolang de voorwaarde waar is / voor een bepaald aantal iteraties."

Voorbeeld in Java-code:

java

// for-loop

for (int i = 1; i <= 10; i++) {

System.out.println(i);

}

// while-loop

int j = 1;

while (j <= 10) {

System.out.println(j);

j++;

}

// do-while-loop

int k = 1;

do {

System.out.println(k);

k++;

} while (k <= 10);

import java.util.ArrayList;

##### enhanced for-loop

public class Main {

public static void main(String[] args) {

// Maak een ArrayList van strings

ArrayList<String> namen = new ArrayList<>();

namen.add("Alice");

namen.add("Bob");

namen.add("Charlie");

// Gebruik een for-each loop om door de ArrayList te itereren

for (String naam : namen) {

System.out.println(naam);

}

}

}

In dit voorbeeld wordt een ArrayList met de naam "namen" gemaakt en vervolgens worden er enkele namen aan toegevoegd. Daarna wordt de for-each loop gebruikt om door de ArrayList te itereren. Bij elke iteratie wordt de huidige naam toegewezen aan de variabele "naam" en wordt deze afgedrukt met behulp van de "System.out.println()" methode. Het resultaat van het bovenstaande programma zal zijn:

#### Lists

##### ArrayList:

Een ArrayList is een dynamische array die elementen van dezelfde soort kan bevatten.

Syntax in tekst: "Maak een ArrayList en voeg elementen toe / haal elementen op / verwijder elementen / controleer de grootte."

Voorbeeld in Java-code:

java

import java.util.ArrayList;

// ArrayList maken en elementen toevoegen

ArrayList<String> namen = new ArrayList<>();

namen.add("Alice");

namen.add("Bob");

namen.add("Charlie");

// Elementen ophalen

String eersteNaam = namen.get(0);

System.out.println("Eerste naam: " + eersteNaam);

// Elementen verwijderen

namen.remove("Bob");

// Grootte controleren

int grootte = namen.size();

System.out.println("Aantal namen: " + grootte);

##### HashMap:

Een HashMap is een collectie van key-value paren, waarbij elk element een unieke sleutel heeft.

Syntax in tekst: "Maak een HashMap en voeg key-value paren toe / haal waarden op met behulp van de sleutels / controleer of een sleutel aanwezig is."

Voorbeeld in Java-code:

java

import java.util.HashMap;

// HashMap maken en key-value paren toevoegen

HashMap<String, Integer> leeftijden = new HashMap<>();

leeftijden.put("Alice", 25);

leeftijden.put("Bob", 30);

leeftijden.put("Charlie", 35);

// Waarden ophalen met behulp van de sleutels

int leeftijdVanBob = leeftijden.get("Bob");

System.out.println("Leeftijd van Bob: " + leeftijdVanBob);

// Controleren of een sleutel aanwezig is

boolean bevatAlice = leeftijden.containsKey("Alice");

System.out.println("Bevat Alice: " + bevatAlice);

##### LinkedList:

Een LinkedList is een verzameling van elementen waarbij elk element verwijst naar het volgende element in de lijst.

Syntax in tekst: "Maak een LinkedList en voeg elementen toe / haal elementen op / verwijder elementen / controleer de grootte."

Voorbeeld in Java-code:

java

import java.util.LinkedList;

// LinkedList maken en elementen toevoegen

LinkedList<String> steden = new LinkedList<>();

steden.add("Amsterdam");

steden.add("Rotterdam");

steden.add("Utrecht");

// Elementen ophalen

String eersteStad = steden.get(0);

System.out.println("Eerste stad: " + eersteStad);

// Elementen verwijderen

steden.remove("Rotterdam");

// Grootte controleren

int grootte = steden.size();

System.out.println("Aantal steden: " + grootte);

##### Jump

###### break statement:

Het break statement wordt gebruikt om een lus of schakelverklaring (switch statement) vroegtijdig te beëindigen.

Wanneer een break statement wordt uitgevoerd binnen een lus, wordt de controle onmiddellijk overgedragen aan de eerstvolgende instructie buiten de lus.

In een switch statement zal een break statement de uitvoering van de switch statement beëindigen en de controle overdragen aan de eerstvolgende instructie buiten de switch.

Voorbeeld van break statement in een lus:

java

for (int i = 1; i <= 10; i++) {

if (i == 5) {

break; // beëindigt de lus als i gelijk is aan 5

}

System.out.println(i);

}

Uitvoer:

1

2

3

4

Voorbeeld van break statement in een switch statement:

java

Copy code

int dag = 3;

switch (dag) {

case 1:

System.out.println("Maandag");

break;

case 2:

System.out.println("Dinsdag");

break;

case 3:

System.out.println("Woensdag");

break; // beëindigt de switch statement nadat "Woensdag" is afgedrukt

default:

System.out.println("Ongeldige dag");

}

Uitvoer:

Woensdag

###### continue statement:

Het continue statement wordt gebruikt om de huidige iteratie van een lus over te slaan en door te gaan met de volgende iteratie.

Wanneer een continue statement wordt uitgevoerd binnen een lus, worden de resterende instructies binnen de huidige iteratie overgeslagen en wordt de controle overgedragen aan het begin van de volgende iteratie.

Voorbeeld van continue statement in een lus:

java

for (int i = 1; i <= 5; i++) {

if (i == 3) {

continue; // slaat de rest van de instructies in de huidige iteratie over

}

System.out.println(i);

}

Uitvoer:

1

2

4

5

###### return statement:

Het return statement wordt gebruikt om de uitvoering van een methode te beëindigen en een waarde (of geen waarde) terug te geven aan de oproepende code.

Wanneer een return statement wordt uitgevoerd, wordt de resterende code in de methode overgeslagen en wordt de controle overgedragen aan de oproepende code.

Voorbeeld van return statement in een methode:

java

public int som(int a, int b) {

return a + b; // geeft de som van a en b terug

}

In dit voorbeeld wordt de methode som beëindigd zodra het return statement wordt uitgevoerd, en de som van de twee ingevoerde getallen wordt teruggegeven aan de oproepende code.