

Week 1 - oefening 3: Collecties in Java

Doelstelling

Introductie tot de programmeertaal Java
Gebruik van de java Collections Framework

Algemene beschrijving

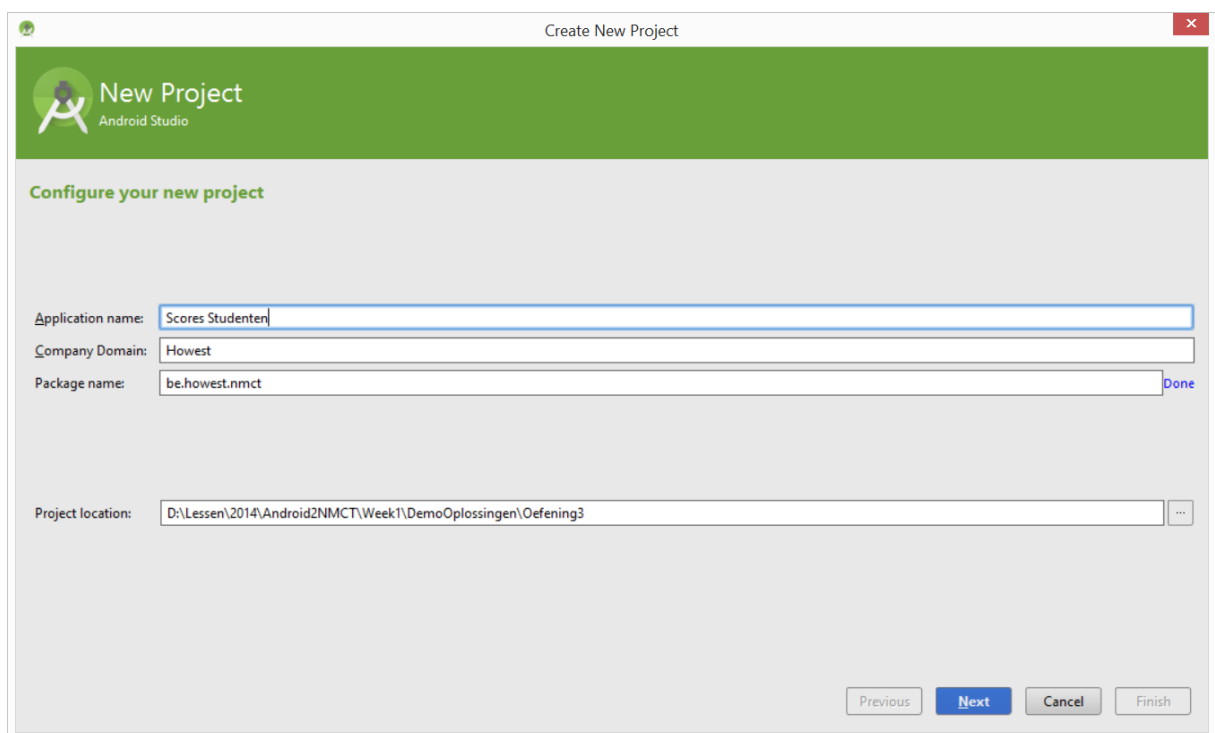
We maken een console Java-applicatie dat in staat is van verschillende studenten hun scores voor hun deelgenomen module-examens bij te houden.

Naast elke afzonderlijke score wordt ook een totaal percentage berekend. Hierbij wordt rekening gehouden met het gewicht van elke module.

Het totaal percentage laat ons ook toe de studenten te ordenen.

Voorbereiding

Maak een nieuw Java-project aan met als naam 'Student Exam'. mMaak onder de src-directory de namespace 'be.howest.nmct.admin aan. Codeer onderstaande klassen.



Create New Project

New Project
Android Studio

Configure your new project

Application name: Scores Studenten

Company Domain: Howest

Package name: be.howest.nmct Done

Project location: D:\Lessen\2014\Android2NMCT\Week1\DemoOplossingen\Oefening3 ...

Previous Next Cancel Finish

Code

Maak een klasse **ModulePunt** aan met volgende attributen:

- moduleNaam: String
- aantalStudiePunten: int
- score: double

Voorzie volgende methodes:

- voor elk attribuut een get- en set-methode.
- constructor(en)
- de toString-methode dat alle info terug geeft.
- equals- & hashCode-methode zodat de gelijkheid van twee objecten op basis van de moduleNaam kan geverifieerd worden.

Maak de klasse **Student** aan met volgende attributen:

- emailStudent: String
 - scoresStudent: HashMap<String, ModulePunt>
- Over het gebruik van de collectieklasse HashMap kan je hier meer info vinden:
<http://docs.oracle.com/javase/tutorial/collections/interfaces/map.html>
<http://docs.oracle.com/javase/7/docs/api/java/util/HashMap.html>

Voorzie volgende methodes:

- voor het attribuut 'emailStudent' een get- en set-methode.
- constructor(en)
- de methode 'voegScoreToe' met parameters sModuleNaam(String), score (double): deze methode maakt intern een object van de klasse ModulePunt aan en voegt deze toe aan de HashMap.
- De methode 'getTotaleScoreStudent' dat het totaal percentage van de student berekent. Het totaal percentage wordt als volgt berekend:
 - Bepaal de totale som van alle studiepunten van de deelgenomen studiepunten
 - Overloop alle scores. Vermenigvuldig elke score met het gewicht van zijn module. Het gewicht van de module is het aantal studiepunten van deze module gedeeld door de totale som van alle aantal studiepunten. Tel alle verkregen producten op om het totaal percentage te verkrijgen.
- de toString-methode dat alle info terug geeft.
- Laat de klasse de interface Comparable implementeren. Meer info over deze interface (en zijn resp. methode(s) die geïmplementeerd moeten worden, vindt u op: <http://docs.oracle.com/javase/7/docs/api/java/lang/Comparable.html>
 Deze interface laat ons later een collectie van objecten van de klasse Student te ordenen. <http://docs.oracle.com/javase/tutorial/collections/interfaces/order.html>
 Zorg ervoor dat de studenten geordened worden op hun totale score.

Voorzie ook volgende static methodes in de klasse Student:

- De static methode getScoresModule krijgt als parameters een List-collectie met objecten van de klasse Student én één modulenaam (string) binnen. De methode geeft een list met alle resultaten over alle studenten heen voor deze module terug.


```
public static List<Double> getScoresModule(List<Student> studenten,
      String moduleNaam) {
```
- De static methode getGemiddeldeScoreModule krijgt als parameters een List-collectie met objecten van de klasse Student én één modulenaam (string) binnen. De methode geeft het gemiddelde score van deze module over alle studenten heen terug.

```
public static double getGemiddeldeScoreModule(List<Student> studenten,
String moduleNaam) {
```

- De static methode sorteerStudenten krijgt als parameter een List-collectie met objecten van de klasse Student binnen. Deze methode sorteert deze list.

```
public static void sorteerStudenten(List<Student> studenten) {
```

Maak een main-klasse '**RunAdmin**' aan waarin we de klassen testen:

- maak via de constructor een aantal objecten van de klasse Student aan. Voeg aan elke student een aantal scores (objecten van de klasse ModulePunt) toe.
- Test nu elke static-methode uit de klasse Student uit.
-



```
"C:\Program Files\Java\jdk1.7.0_51\bin\java" -Didea.launcher.port
Gemiddelde van module Design: 14.333333333333334
De scores voor de module Design zijn:
16.0
19.0
8.0
Sorteert de studenten volgens hun totale score
max.pauwels@howest.be - Totale score: 8.600000000000001
stijn.walcarius@howest.be - Totale score: 14.0
steven.verborgh@howest.be - Totale score: 15.48
```

Info

Wanneer men in een klasse de equals-methode opneemt (en dus deze van de klasse Object overridt), dan dient men in feite ook de hashCode()-methode op te nemen! Deze methode geeft een int-waarde ('hashCode') van het object terug en ondersteunt de werking van een Hashtable (java.util.Hashtable).

Hierbij geldt dat:

- Een object gedurende zijn levensduur binnen een applicatie altijd dezelfde hashCode moet teruggeven.
- Als twee objecten gelijk zijn op basis van de equals-methode, dan moeten ze ook dezelfde hashCode teruggeven.
- Als twee objecten niet gelijk zijn op basis van de equals-methode dan hoeven ze geen verschillende hashCode te hebben.

Android Studio kan naast de equals-methode ook de hashCode-methode automatisch laten genereren.