CMTPRG03-7, Werkcollege – Week 2

Theorie

- Abstract Classes vs Interfaces
  https://www.youtube.com/watch?v=2aQ9Y7bumts

Opdrachten

Opdracht 1
Maak een interface Vehicle met een `drive()`, `setSpeed(int)` en int `getSpeed()`.
Bedenkt 3 implementaties (bijv Bus, Car, Bike) en teken een class diagram

```
Broem, ik ben een auto en ik rijd 120
Tring, tring, ik ben een fiets en ik rijd 15
BROEM ik ben een bus en ik rijd 55
```

Opdracht 2
Programmeer en test je code door een `ArrayList<Vehicle>` te maken en daar verschillende voertuigen in te stoppen. Maak eerst een object aan, zet er daarna de juiste snelheid in, en plaats het daarna in de lijst.
In drive schijf je een geluid van het voertuig en de snelheid naar de console.

Opdracht 3

"Identify the aspects of your application that vary and separate them from what stays the same"

---

setSpeed en getSpeed lijken niet handig in de interface… Los dit op met een abstract class

Opdracht 4
Waarschijnlijk implementeer je vaker een interface, dan dat je er één schrijft, er zijn namelijk veel standaard interfaces in frameworks en programmeertalen.
Als je gebruik maakt van die interface begrijpt het framework precies waar jouw class voor is, en wat hij ermee kan doen.
Bijv. Collections.sort(). Hiermee kan je een lijst van objecten sorteren als deze objecten de interface Comparable implementeren.

Neem opdracht 3 en laat de voertuigen nu op volgorde van snelheid zetten.
Gebruik uiteraard Comparable en Collections.sort() bij deze uitwerking

Voorbereiding Theorieles

Zoek uit wat de Android Lifecycle is

Extra
Vanaf volgende week gaan we Android apps maken. Als je vooruit wilt werken: les 1 t/m 7 van onderstaande gratis cursus behandelen de onderwerpen uit de module.
https://eu.udacity.com/course/new-android-fundamentals--ud851