Inleveropdacht 1

Verslag

Joris Jansen

ITV2E

**Inleiding**

Samen met Joeri Hofman heb ik inleveropdracht één gemaakt. In dit verslag bespreek ik het resultaat van de code die door Bettercodehub is gehaald. De uitkomst bestaat uit tien onderdelen:

1. Write short units of code
2. Write simple units of code
3. Write code once
4. Keep unit interfaces small
5. Seperate concerns in modules
6. Couple architecture components loosely
7. Keep architecture components balanced
8. Keep your codebase small
9. Automate tests
10. Write clean code

**Resultaten**

Hieronder bespreek ik per guideline de resultaten van Bettercodehub. Bij de categorieën die we niet hebben gehaald bespreek ik hoe dat kan.

**Short units of code**

De eerste categorie hebben we gehaald. Vanaf het begin hebben we geprobeerd alles netjes de programmeren. Dat is gelukt; als een methode te lang wordt hebben we een “sub-functie” gemaakt.

**Simple units of code**

Ook deze categorie hebben we gehaald.

**Write code once**

We hebben zo min mogelijk code dubbel geschreven. Natuurlijk zijn er wel enkele dubbele lines, maar we hebben geprobeerd dat zo veel mogelijk te beperken. Dat is de reden dat wij ook de categorie voldoende hebben gehaald.

**Unit interfaces small**

Een guideline voor deze categorie is om niet meer dan vier parameters per unit te hebben. Het hoogste aantal parameters Het hoogste aantal parameters voor een methode is twee. Wij voldoen dus ruim aan de guideline.

**Seperate concerns in modules**

Voor deze opdracht hebben we gebruik gemaakt van het MVC-model. (Model-View-Controller) De model verzorgt al het rekenwerk, in het geval van deze opdracht dus de sorteeralgoritmen. De view zorgt voor de GUI en de controller zorgt dat de interactie met de gebruiker goed verloopt.

Op deze manier hebben we alle “concerns” gescheiden gehouden.

**Couple architecture components loosely**

De componenten zijn erg goed object georiënteerd. Op deze categorie hebben we dus voldoende gescoord.

**Keep architecture components balanced**

Dit is één van de twee categorieën waar we niet voldoende op hebben gescoord. Hij heeft één architecture compontent gevonden. En dat klopt. Het is geen grote applicatie en het was dus niet heel erg nodig.

**Automate tests**

Dit is de tweede categorie die we niet hebben gescoord. De meeste tests hebben we tijdens heb programmeren gedaan. Het is dus zeker niet zo dat we niet getest hebben.

**Write clean code**

De laatste categorie hebben we wel gehaald. Deze categorie lijkt een beetje op de eerste twee categorieën. Het verschil is dat deze meer gaat over commentaar en dergelijke. Als er geen grote lappen comments in staan is deze categorie voldoende. En dat is gelukt.

Met acht van de tien categorieën goed kunnen we erg tevreden zijn met het resultaat. Volgens dit resultaat is het programma dus goed te onderhouden en te debuggen.