HBO-ICT Code Conventions voor de propedeuse

Alle broncode die bij opdrachten of projecten wordt gemaakt moet voldoen aan de HBO-ICT Code Conventions (coding standards). Door allemaal dezelfde Code Conventions te hanteren verbetert de leesbaarheid en wordt broncode makkelijker en beter te begrijpen en onderhouden. In de praktijk gaat 80% van de kosten van software naar het onderhoud ervan (zoals uitbreiding en verbeteringen). Opleveren van leesbare code is daarom van essentieel belang.

De HBO-ICT Code Conventions zijn een opleidingsbrede standaard waaraan elke medewerker en student zich dient te houden. De regels zijn afgeleid van de Sun Code Conventions (zie referenties).

Layout

1. Naam van de auteur en het doel van het programma staan boven de hoofdklasse van het programma.

2. Layout van de code is zodanig dat de structuur zowel bij afdruk als op scherm goed te zien is. De NetBeans standaard “format” optie volstaat:
   - Indentatie 4 spaties
   - Sluitaccolade “}” op dezelfde kolom als 1e letter van de regel met open-accolade “{“
   - Regels niet langer dan 120 tekens, liefst korter dan 100

Naamgeving

3. Namen van klassen en variabelen zijn volgens de taalstandaard:
   - Java: klassen beginnen altijd met hoofdletter, variabelen en methoden nooit.
   - Bij langeNamenZoalsDit geen streepjes of underscores gebruiken, maar CamelCase.
   - Constanten moeten in hoofdletters worden geschreven met underscores, bijvoorbeeld: `final int MAX_HOOGTE.`

4. Naamgeving van methoden, variabelen en klassen dient de werking te verduidelijken (selfexplainable code):
   - niet `int s; // s is studentnummer`. Gebruik dan liever: `int studentnummer;`
   - methoden beginnen met een werkwoord (get, set, do, move, sort, enz.)
   - de naam van een klasse is een zelfstandig naamwoord

5. Magic Numbers in programmacode voorkomen of vervangen door (standaard) constanten.
   - Dus niet: `omtrek = diameter * 3.1415;`
     maar `omtrek = diameter * Math.PI;`
   - Gebruik bijvoorbeeld `public static final String HOSTNAAM = "oege";` om stringconstanten met een duidelijke naam te definiëren en gebruik deze dan in plaats van de letterlijke string.

Commentaar

6. Elke methode, klasse of attribuut moet voorafgegaan worden door een toelichting:
   - Wat is het doel?
   - Wat levert de methode op? (javadoc @return)
   - Wat betekenen de parameters van de methode? (javadoc @param)

7. Commentaar moet het waarom toelichten en niet herhalen wat er al staat
   - Fout voorbeeld: `index++; // verhoog index met 1`
   - Goed voorbeeld: `index++; // verwijs naar volgende student in de rij`

Dode code

8. Verwijder statements die weg kunnen en maak je programma “schoon”. Dit geldt ook voor code die “commented-out” is. NetBeans herkent ongebruikte importregels en variabelen maar kan niet altijd “zien” of een stuk code helemaal weg kan. Zorg ervoor dat je programma leesbaar en begrijpelijk is voor beginners.

Scoping

9. Variabelen worden (alleen) gedeclareerd op de plek waar ze gebruikt worden.
   - Klasse-attributen mogen niet worden gebruikt als vervanging van methode parameters of lokale variabelen van een methode.
   - Static attributen mogen niet worden misbruikt als globale variabelen of als vervanging van een associatie tussen twee klassen.
   - Attributen van een klasse zijn private. Uitzonderingen toelichten.

Referenties

a. Sun Code conventions for the Java Programming Language: *http://www.oracle.com/technetwork/java/javase/documentation/codeconventions-139411.html*

b. How to write Doc Comments for Javadoc: *http://www.oracle.com/technetwork/java/javase/documentation/index-jsp-135444.html#javadocdocuments*

c. Camel Case uitgelegd: http://en.wikipedia.org/wiki/CamelCase

d. Wat zijn Magic Numbers (Unnamed Numerical Constants): *https://en.wikipedia.org/wiki/Magic_number_(programming)#Unnamed_numerical_constants*

Voorbeeld voor de layout.

```
/*
 * Licentie: deze software is exclusief bedoeld voor onderwijsdoeleinden
 * aan de Hogeschool van Amsterdam (HvA).
 */
package nl.hva.dmci.general;

/**
 * Het is bedoeld als voorbeeld voor de coderingstandaard. En met name om aan te
 * geven waar in een Java source file het doel van het programma en de naam
 * van de auteur staan.
 *
 * @author N.J. Tromp
 */
public class CodingStandards {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
    }

}
```