Java Coding Standards

Dateiorganisation and Kommentare

✓ Übliche Reihenfolge der Elemente in einer Java-Sourcedatei

① Datei-header

② ‚package' Anweisung

③ ‚import' Anweisungen

④ Hauptklasse(‚public class')

⑤ Klassenvariablen(‚static'), geordnet nach Sichtbarkeit, angefangen mit 'public'

⑥ Instanzvariablen (ohne 'static'), wieder geordnet nach Sichtbarkeit

⑦ 'main()'-Methode (falls vorhanden)

⑧ Konstruktoren

⑨ Methoden (wichtige zuerst, aber thematisch zusammenhängende Methoden beieinander)

⑩ Weitere interne Klassen (falls vorhanden)

✓ Ein möglicher Datei-Header

```
/*
 * Project: MeinProjektname
 * $Header: $
 * Author:  MeinName
 * Last Change:
 *    by:   $Author: $
 *    date: $Date:   $
 * Copyright (c): MeineFirma, Jahr
 */
```

✓ Zeilen- und Dateilänge

Java-Sourcecodezeilen sollten nur ein Statement beinhalten und nicht länger als 120 Zeichen sein. Java-Sourcedateien sollten nicht länger als 2000 Zeilen sein

Einrueckung, Klammern und Leerzeichen

✓ Tabulator

Stellen Sie Ihren Editor so ein, dass er Tabulatorzeichen durch Leerzeichen ersetzt (siehe z.B. Eclipse).Die Tabulatorbreite sollte projekteinheitlich auf entweder zwei, drei oder vier Zeichen eingestellt werden.

✓ Leerzeilen

Innerhalb von Methodendefinitionen wird nicht mehr als eine einzelne Leerzeile verwendet, um Blöcke zu separieren.

Zwischen Methoden werden eine oder zwei Leerzeilen gesetzt. Wenn nach der Hauptklasse noch eine weitere Klasse definiert wird, wird sie mit zwei Leerzeilen abgetrennt.

✓ Geschweifte Klammern

Die öffnende geschweifte Klammer ('{') kann am Ende der vorherigen stehen. Bei Bedingungen und Schleifenkonstrukten ('if', 'for', 'while') wird die öffnende Klammer meistens an das Ende der Bedingungs- oder Schleifenkonstruktionszeile gesetzt. Die schließende geschweifte Klammer ('}') steht immer in einer eigenen Zeile, außer es folgt ein zur Bedingung gehörendes Konstrukt wie 'else', 'while', 'catch' oder 'finally'. Der Teil zwischen den Klammern wird eingerückt.

Beispiel

```
public class MeineKlasse {
    public static void main(String[] args) {
        int[] array = new int[10];
        for(int i=0; i < array.length; i++){
            array[i] = i;
        }
    }
}
```

✓ Leerzeichen bei runden Klammern

Bei Bedingungen und Schleifenkonstrukten ('if', 'for', 'while') sowie bei Methoden gehen die Meinungen auseinander, ob und wo bei den runden Klammern Leerzeichen gesetzt werden sollen.

Sun hat in seinen Code Conventions vorgeschlagen, bei Bedingungen und Schleifenkonstrukten ein Leerzeichen vor die öffnende Klammer zu setzen und bei Methoden keine Leerzeichen zu setzen

```
public void meineMethode(String meinParm1, String meinParm2)
{
  if (null != getString(meinParm1)) {
    procede(meinParm2);
  }
  for (int i=0; i<n; i++) {
    doSomething(i);
  }
}
```

Naming Contenctions, Java-Typen

✓ package, import

'package'-Namen werden komplett in Kleinbuchstaben und ohne Unterstriche geschrieben. Die Namen der eigenen Pakete beginnen mit dem umgekehrten Domainnamen ('de.meinefirma.<projekt>.<kontext>').

Die 'package'-Struktur und -Hierarchie sollte möglichst der Softwarearchitektur entsprechen (z.B. Aufteilung in Präsentations-, Geschäftslogik- und Persistenzschicht). Falls die Klassen eines 'packages' einem Pattern entsprechen, sollte das 'package' so heißen (z.B. 'dao').

Interfaces und Superklassen sollten in der 'package'-Hierarchie entweder parallel, gleich oder höher angeordnet sein, aber nicht tiefer als die abgeleiteten Unterklassen.

<Beispiel>

```
package de.meinefirma.myprojectxy.mysubpackage;

import java.io.BufferedInputStream;
import de.meinefirma.myprojectxy.businesslogic.MeineEntityKlasse;
```

✓ class, interface

Klassennamen (und Interface-Namen) beginnen mit einem Großbuchstaben und werden weiter mit Kleinbuchstaben geschrieben, aber bei zusammengesetzten Wörtern beginnt jedes interne Wort wieder mit einem Großbuchstaben ('Pascal'-Schreibweise). Sie enthalten keine Unterstriche.

Klassennamen sollten möglichst aus Substantiven gebildet werden

<Beispiel> : “class ImageCoverter” , “class ComparatorImpl”, “class BusinessEntity”

✓ Methoden

Methoden werden klein geschriebenen, aber bei zusammengesetzten Wörtern beginnt jedes interne Wort wieder mit einem Großbuchstaben. Sie enthalten keine Unterstriche. Methodennamen sollten mit Verben beginnen.

<Beispiel> : getColor() , setAttribute() , createInstance() ...usw

✓ Attribute, Variablen

Die Schreibweise von Variablen ist wie die der Methoden. Sie sollten nicht mit '_' oder '$' beginnen und normalerweise auch nicht mit einem den Typ spezifizierenden Präfix.

Variablennamen sollten die Bedeutung verdeutlichen, möglichst als Substantiv. Einbuchstabige Variablennamen sollten nur in sehr kurzen temporären Blöcken verwendet werden (z.B. 'for(int i=0; i<n; i++) ...'). In solchen Fällen sollten i, j, k, m, n für Integers, c für Characters und s1, s2 für Strings bevorzugt werden.

- ✓ **Konstanten**

  Konstanten werden komplett in Großbuchstaben geschrieben. Interne Wortbestandteile werden durch Unterstriche ('_') voneinander getrennt. Konstanten sollten nicht direkt im Code zum Beispiel als Zahl eingefügt werden (keine 'Magic Numbers'), sondern im Klassenkopf als 'static final'-Konstante definiert werden.

```
static final int    WIDTH_MIN = 100;
static final int    WIDTH_MAX = 999;
static final String MEIN_NAME = "Torsten";
```

● Bedingungen

✓ If, if-else, if else-if Bedingung

if-Bedingungen nie ohne geschweifte Klammern verwendet werden sollen.

Diese Bedingung soll die folgende Form folgen :

```
if (condition) {
    statements;
}

if (condition) {
    statements;
} else {
    statements;
}

if (condition) {
    statements;
} else if (condition) {
    statements;
} else {
    statements;
}
```