Stilfibel

1 Festlegungen für die Quelltexte:

Generell gilt die Java Code Convention, d.h. es kann unter anderem mit `javadoc` eine Dokumentation erstellt werden. Ausnahmen und Zusätze werden hier beschrieben. (siehe auch Anlage)

1. Für jede Klasse wird eine extra Datei verwendet. Ausnahmen sind innere Klassen.
2. Das Dateiformat ist Unix kompatibel. In Dateinamen werden keine Leerzeichen verwendet. Leerzeichen können durch Unterstriche ersetzt werden.
3. Die maximale Zeilenlänge beträgt 79 Zeichen.

4. Geschweifte öffnende Klammern werden nach dem entsprechendem Wort und einem Leerzeichen geschrieben.
5. Für Einrückungen werden 2 Leerzeichen verwendet.
6. Zwischen den Methoden werden 2 Leerzeilen eingefügt.
7. Alle `case`-Anweisungen in einem `switch`-Block werden mit einem `break` versehen. In Ausnahmefällen darf davon abgewichen werden, jedoch nur wenn dieser Fall **ausführlich** dokumentiert ist.
8. Variablen werden initialisiert.
9. Rechenoperationsabkürzungen (z. B. `i++`) sollen nur bei einzelnen Laufvariablen akzeptiert. Seiteneffekte dürfen nicht genutzt werden.

10. Klassennamen beginnen mit einem Großbuchstaben.
11. Methodennamen und Variablennamen beginnen mit einem Kleinbuchstaben.
12. Konstanten werden komplett groß geschrieben.
13. Namen werden nicht mehrfach vergeben. Ausnahme sind Laufvariablen bzw. lokale Variablen.

14. Kommentare werden in deutsch geschrieben.
15. Umlaute sind **nur** in Kommentaren erlaubt.
16. @return in der Methodenbeschreibung.
17. @param Variable in der Methodenbeschreibung.
18. In den Beschreibungen der Methoden Klassen können HTML Tags verwendet werden, wie z.B. <font>, <p>,
, Links. Diese sollten aber nur ergänzend genutzt werden.
19. Generell sollen Interfacedokumentationen ausführlicher sein als die Implementierungsdokumentationen. Diese enthalten speziellere Hinweise.
20. Variablen werden ausführlich kommentiert.
21. Alle schließenden geschweiften Klammern werden der besseren Übersicht wegen kommentiert.

Anlage:

Ein Beispielprogramm:

Thomas Runge (trunge@informatik.tu-cottbus.de) Rolf Ginter (rginter@informatik.tu-cottbus.de)
Michael Vogel (mvogel@informatik.tu-cottbus.de) Ray Wojciechowski (rwojciec@informatik.tu-cottbus.de)

```
/**
 * In der 1. Zeile steht eine Zusammenfassung dessen, was in der Klasse
 * passiert.
 *                                                          14
 * Hier wird die Klasse genauer beschrieben.
 *
 * @version aktuelles Datum
 * @author
 *   <b><a href="mailto:rginter@informatik.tu-cottbus.de">Rolf
 *              Ginter</a></b><br>
 *   <b><a href="mailto:trunge@informatik.tu-cottbus.de">Thomas
 *              Runge</a></b><br>
 *   <b><a href="mailto:rwojciec@informatik.tu-cottbus.de">Ray
 *              Wojciechowski</a></b><br>
 *   <b><a href="mailto:mvogel@informatik.tu-cottbus.de">Michael
 *              Vogel</b></a>
 */                                  17

public class Example {
  private static int classAttribute;   // eine Klassenvariable
  private final int CONSTANT;           // eine Konstante
12
  /**
   * Kurzbeschreibung der Methode.
   *
   * Ausführliche Beschreibung der Methode, mit Hinweisen auf eventuelle
   * Probleme.
   *
16 * @param  param     Beschreibung der Variable
   * @return Beschreibung des Rückgabewertes
15
  public void methodName(int param) {
    int methodAttribute;         // ein Attribut der Methode
    String methodAttribute2;     // ein weiteres Attribut
    boolean condition;           // noch ein Methodenattribut

    for (int i=0; i<methodAttribute; i++) {
      //ACHTUNG! nicht erlaubt sind solche Konstrukte:
      methodAttribute = (i++)* methodAttribute;
    } // for (int i=0; i<methodAttribute; i++)
    if (condition) {
      // tue etwas
    } else {
      // tue etwas anderes
    } // if (condition)
    switch methodAttribute {
      case 1: // hier erfolgt die Erklärung
              methodAttribute++;
              break;
      case 2: // Erklärung
              methodAttribute--;
              break;
      case 3: // Erklärung
              methodAttribute + 3;
              // GENAUE Erklärung, warum break weggelassen wird
      case 4: // Erklärung                                          7
              methodAttribute * 5;
              break;
      default: // Erklärung
    } // switch methodAttribute;
  } // methodName
} // class Example
20
```