Programmierrichtlinien zur OOS Ergänzung 1

FH AACHEN UNIVERSITY OF APPLIED SCIENCES

Kommentierung mit javadoc

Die Kommentierung der Programme mit javadoc ist ab Praktikumsversuch Nr. 2 zwingend vorgeschrieben.

Hierbei ist folgendes als Dokumentationskommentar in den einzelnen Java-Programmen anzugeben:

1. **Kopf einer Klasse**

Im Kopf der Klasse wird die Funktionalität und die grobe Struktur der Klasse beschrieben.

Außerdem werden die Standardeinträge für Copyright, Organisation, Autor und Version ausgefüllt.

Bsp.:

```
/**
 * <p>Überschrift: Struktur von Benutzern </p>
 * <p>Beschreibung: Diese Klasse definiert die grundlegende Struktur von
 *                  Benutzern.
 *                  Die Struktur eines Benutzers setzt sich zusammen aus:
 *                  - der UserId und
 *                  - dem Passwort
 *                  Da es nur um eine Struktur geht, werden lediglich die
 *                  beiden Standardmethoden equals und toString
 *                  implementiert.</p>
 *
 * <p>Copyright: Heinz Faßbender Copyright (c) 2003</p>
 * <p>Organisation: FH Aachen, FB05 </p>
 * @author Heinz Faßbender
 * @version 1.0
 */

public class Benutzer { ...
```

2. **Bedeutung der public-Attribute**

Vor jedem public-Attribut wird die Bedeutung des Attributs beschrieben.

Bsp.:

```
/**
 *  Attribut zur Speicherung der UserID:
 */
  public  String userId;
```

Für alle anderen Attribute wird deren Bedeutung mit normalen Kommentaren beschrieben

3. **Bedeutung der public-Konstruktoren**

Die Konstruktoren müssen nur kommentiert werden, wenn sie außer der Initialisierung noch andere Funktionalität liefern.

Bsp.:

FH AACHEN
UNIVERSITY OF APPLIED SCIENCES

```
/**
 *  liefert noch Ausgabe, dass er in Standardkonstruktor ist
 */
  public Benutzer() {
    System.out.println("Bin in Standardkonstruktor!");
  }
```

4. **Bedeutung der public-Methoden**

Die Kommentierung der public-Methoden ist besonders wichtig, da diese die Dienste des Objekts liefern und somit die Schnittstelle implementieren. Deshalb wird hier besonderer Wert draufgelegt. Außerdem ist die Bedeutung der Parameter jeweils mit dem Tag @param zu beschreiben.

Bsp.:

```
/**
 *  Standardmethode
 *  @param benutzer liefert das Objekt, dessen Inhalte mit denen des
 *                  aktuellen Objekts verglichen werden sollen.
 */
  public boolean equals(Benutzer benutzer) {
    return (this.userId.equals(benutzer.userId) &&
            this.passWort.equals(benutzer.passWort));
  }

/**
 *   Standardmethode, die die Inhalte der beiden Attribute in der folgenden
 *   Form ausgibt: userId/passWort
*/

  public String toString() {
    return (userId + "/" + passWort);
  }
```

Für alle anderen Methoden wird deren Funktionalität mit normalen Kommentaren beschrieben.