Objektorientierte Programmierung

Dokumentation mit Javadoc

Prof. Dr. Ulrike Hammerschall
Fakultät für Informatik und Mathematik

Javadoc

- Javadoc ist ein Werkzeug des Java Development Kits zur Generierung der Dokumentation für ein Programm.
- Basis der Generierung sind Javadoc-Kommentare zu allen Elementen einer Klasse (Member), z.B.
  - vollständigen Klasse
  - Objektvariablen, Klassenvariablen, Konstanten
  - Methoden
- Syntax der Javadoc-Kommentare unterscheidet sich leicht zur Syntax normaler Kommentare.
- Javadoc-Kommentare werden je nach Kommentartyp um sogenannte Tags ergänzt.

Syntax-Rahmen

- Variablen und Konstanten:

```
/**
 * Fachliche Beschreibung der Variable bzw. Konstante
 */
```

- Klassen, Methoden und Konstruktoren:

```
/**
 * Zusammenfassung in einem Satz.
 * Ausführliche Beschreibung
 *
 * Tags (erkennbar am Zeichen @) für spezielle Ergänzungen zu
 * erwarteten Werten für Parametern, Ergebnis der Methode etc.
 */
```


Eine Auswahl der wichtigsten Tags abhängig von Elementen

- Klassen
  - @author <Name des Autors der Klasse>
  - @version <Versionsnummer>
- Methoden, Konstruktoren
  - @param <Parametername und Beschreibung>
  - @return <Rückgabewerte und ihre Bedeutung>
  - @throws <mögliche Exceptions>
  - @see <Verweis auf andere Methodendefinition>

Wie dokumentieren?

- Pro Klasse einen Javadoc-Kommentar
  - nach package-Klausel und import-Klauseln, direkt vor der class Definition mit den entsprechenden Tags
- Pro Variable / Konstante einen Javadoc Kommentar
  - Nur Text, keine Tags
- Pro Methode / Konstruktor einen Javadoc Kommentar
  - immer Kurzbeschreibung mit Punkt am Ende!
  - ggf. nachfolgende Langbeschreibung. Mehrere Sätze möglich.
  - @param und @return falls erforderlich
  - ggf. @throws und @see
- Javadoc-Kommentare stehen NIEMALS innerhalb einer Methode. In Methoden sind nur einfache Java-Kommentare zulässig.

Was dokumentieren bzw. kommentieren?

- Das Wichtigste vorweg:
  - In die Dokumentation gehören keine technischen Details zur Umsetzung. Es ist eine fachliche Beschreibung der Schnittstelle.
  - Sie dokumentieren nicht (nur) für sich, sondern
    - Für jemanden, der ihre Klassen verwenden möchte => **Blackbox View**
    - Für jemanden, der ihre Klassen erweitern / anpassen / ändern möchte => **Whitebox View**
- Blackbox View
  - Realisiert durch Javadoc-Kommentare mit daraus generierter Dokumentation.
  - Beschreibt, was eine Methode erwartet, was sie liefert und wie sie ggf. im Fehlerfall reagiert (das nach außen sichtbare Schnittstellenverhalten).
- Whitebox View
  - Realisiert durch einfache Java-Kommentare in den Methoden.
  - Beschreibt, wie die Methode das Schnittstellenverhalten tatsächlich umsetzt.
- Whitebox View und Blackbox View sollten sich ergänzen, nicht inhaltlich duplizieren.