Qualitätssicherungskonzept

27.05.2010

Inhaltsverzeichnis

1 Dokumentationskonzept

1.1 Bezeichner

Für alle Variablen, Methoden, Klassen und Konstanten gilt, dass nur sprechende Namen zur Bezeichnung verwendet werden um die Bedeutung des jeweiligen Objektes für den Leser besser deutlich zu machen. Hierfür wird als einheitliche Bezeichnungssprache Englisch festgelegt. Weiterhin gelten folgende Regeln für die Schreibweise:

- **Klassenbezeichner** beginnen mit einem Großbuchstaben. Sollten sie mehrere Wörter enthalten, so wird der Anfangsbuchstabe jedes neuen Wortes groß geschrieben.
- **Variablen- und Methodenbezeichner** beginnen mit Kleinbuchstaben. Für mehrere Wörter gelten die Regeln analog zu Klassennahmen.
- **Konstantenbezeichner** enthalten nur Großbuchstaben. Sollten sie mehrere Wörter enthalten, so werden diese durch Unterstriche getrennt.

1.2 Einrückung

Um eine gute Lesbarkeit zu gewährleisten wird jeder Unterbefehl (also wenn er im Rumpf einer Klasse oder einer Funktion steht) um vier Leerzeichen eingerückt. Tabulatoren werden nicht verwendet. Außerdem wird jeder neue Befehl auf eine extra Zeile geschrieben. Dabei ist eine Zeilenlänge von 80 Zeichen einzuhalten. Nur in Ausnahmen zur Verbesserung der Lesbarkeit darf die Zeilenlänge auch bis zu 120 Zeichen betragen.

In Funktionsköpfen kommen außer nach Kommas keine Leerzeichen zum Einsatz, auch nicht vor oder hinter den runden Klammern. Öffnende und schließende geschweifte Klammern des Funktionsrumpfes stehen auf seperaten Zeilen. Zwischen den Zeilen mit den Klammern des Funktionsrumpfes und den Zeilen des enthaltenen Codes wird keine extra Zeile frei gelassen. Beispiel:

```
nameDerFunktion(ersterParameter, zweiterParameter, ...,letzterParameter)
{
    code;
}
```

1.3 Kommentare

Um einen verständlichen Quelltext zu erstellen, werden druchgängig Kommentare eingefügt. Wichtige Algorithmen sind ebenfall zu dokumentieren um später eine gute Nachvollziehbarkeit zu garantieren. Ist ein Kommentar nicht länger als eine Zeile, so wird `//` zur Kennzeichnung verwendet. Bei mehrzeiligen Kommentaren wird der Beginn mit `/*` und das Ende mit `*/` gekennzeichnet. Dabei beginnt jede neue Zeile innerhalb dieses Kommentars mit `*`. Beispiel:

```
// Dies ist ein einzeiliger Kommentar.

/* Dieser Kommentar, der dazu dient mehrzeilige
 * Kommentare zu demonstrieren, ist mehrzeilig.
 * Um genau zu sein: er ist drei Zeilen lang.
 */
```

1.4 PHP-Code

PHP-Code wird immer mit `<?php` eingeleitet. In Dateien die nur PHP-Code enthalten darf jedoch kein schließender Tag `?>` verwendet werden.

1.5 Kontrollstrukturen

Nach dem Kontrollstrukturbefehl (`if`, `while` oder `for`) folgt ein Leerzeichen und dann in runden Klammern die Bedingung. Auf der gleichen Zeile durch ein Leerzeichen getrennt, steht dann noch die öffnende Klammer `{` des Rumpfes. Die schließende Klammer des Rumpfes `}` steht in einer seperaten Zeile auf Höhe des ersten Zeichens der Kontrollstruktur. Beispiel:

```
if (var == 0) {
    code;
}
```

1.6 Dokumentation

Zur Dokumentation ist phpdoc zu verwenden. Dabei sind in jedem Dokumentenkopf folgende Standard-Tags zu verwenden: `@copyright`, `@license`, `@package`, `@author`, `@version`. Darüber hinaus soll zu jeder Funktion eine Beschreibung vorhanden sein, sowie die Tags `@param` und `@return`

2 Testkonzept

2.1 Allgemeines

Für die Gewähleistung der Funktionalität der Software wird hauptsächlich mit Selenium getestet um ein Testen der Erweiterung aus Nutzersicht zu ermöglichen. Die Test laufen in verschieden Phasen ab:

- Komponententest
- Integrationstest
- Systemtest

Bei auftretenden Problemen wird die verursachende Komponente ausfindig gemacht, und der Fehler analysiert. Zur Behebung des Fehlers wird wie bei "Organisatorische Festlegung"beschrieben verfahren. Alle wichtigen Tests werden in einer Testsuite gesammelt. Die Testbeispiele werden von den festgelegten Programmiererteams erstellt.

2.2 Selenium

Selenium ist ein Testframework für Webanwendungen. Für das Projekt wird Selenium IDE verwendet, da dies,in Form eines Mozilla-Firefox-Add-on, die Möglichkeit bietet durch Interaktion mit einer Webanwendung Testfälle aufzuzeichnen und bei Bedarf schrittweise abzuspielen. Weiter können Überprüfungen mittels verify und assert eingebunden werden.

2.3 Komponententest

Zu Testen sind hierbei die einzelnen Klassen und Methoden. Hierfür werden parallel zur Implementierung der einzelnen Stories Testbeispiele festgelegt um die Klassen und Methoden hinsichtlich ihrer Funktionalität zu prüfen. Hierfür kann auch das Testframework phpunit verwendet werden, da dies besonders für des automatisierte Testen einzelner Klassen und Methoden geeignet ist.

2.4 Integrationstest

Der Integrationstest überprüft das Zusammenspiel der einzelnen Komponenten und setzt somit auch einen erfolgreichen Komponentest der Stories voraus. Getestet wird regelmäßig zu einem festgesetzen Zeitpunkt mit dem aktuellen Stand der Erweiterung.

2.5 Systemtests

Es wird das Verhalten der Erweiterung bestehend aus allen Komponenten aus Nutzersicht bezüglich der festgelegten Anforderungen überprüft. Da mit Selenium IDE bereits aus Anwendersicht getestet wird, wird der Systemtest ebenfalls regelmäßig zu einem bestimmten Zeitpunkt durchgeführt.

3 Organisatorische Festlegungen

Alle Gruppenmitglieder, die am Code des Projektes arbeiten, stehen für ihren erarbeiteten Code in der Verantwortung. Sie müssen sich vorher mit den festgelegten Dokumentationsrichtlinien vertraut machen und diese bei der zu leistenden Arbeit umsetzen. Vor jeder gruppeninternen Abgabe ist die Umsetzung der Richtlinien durch den Programmierer selbst zu prüfen. Eine weitere Kontrolle zur Garantie der Einhaltung findet durch den Verantwortlichen für Dokumentation & Qualitätssicherung statt, welcher durch ein weiteres Gruppenmitglied unterstützt werden kann.

Auftretende Fehler während der Entwicklung, werden mit einer kurzen Beschreibung und der (möglichen) Ursache dokumentiert. Somit sollen ähnliche Fehler vermieden bzw. schneller erkannt werden. Für die Behebung des Fehlers ist der Programmierer selbst verantwortlich, jedoch kann mit Rücksprache auch eine Fehlerbehebung durch ein anderes Gruppenmitglied erfolgen.