Wikilenium

Selenium Test Projekt für SWT

Melissa Beck

Ivar Contessoto

Olivia Kaufmann

Rahel Lanz



Inhalt

[2 Einleitung 3](#_Toc5389638)

[2.1 Philosophen-Wiki 3](#_Toc5389639)

[3 Wikilenium 4](#_Toc5389640)

[3.1 Technologien 4](#_Toc5389641)

[3.1.1 Selenium 4](#_Toc5389642)

[3.1.2 Allure 4](#_Toc5389643)

[3.2 Systemanforderungen 4](#_Toc5389644)

[3.3 Annahmen 4](#_Toc5389645)

[3.4 Implementation 4](#_Toc5389646)

[3.4.1 Wikilenium-core 4](#_Toc5389647)

[3.4.2 Wikilenium-Ui 5](#_Toc5389648)

[4 Entwicklungsstrategie 5](#_Toc5389649)

[4.1 Auftragsanalyse 5](#_Toc5389650)

[4.2 Prototyp implementieren 5](#_Toc5389651)

[4.3 Testapparat verbessern 5](#_Toc5389652)

[4.4 User Interface 5](#_Toc5389653)

[4.5 Dokumentation Annahmen 5](#_Toc5389654)

[4.6 Erweiterung für Test Batches und Excel File 5](#_Toc5389655)

# Einleitung

Das Projekt «Wikilenium» dient als Übungsprojekt für das Modul SWT. Dabei sollen sich die Studenten mit dem Testing-Tool «Selenium» einen Testapparat zur Testautomatisierung implementieren. Mit dem Testapparat soll das Spiel Philosophen-Wiki getestet werden.

## Philosophen-Wiki

Philosophen-Wiki ist ein Spiel, welches innerhalb der Wikipedia gespielt wird. Für das Spiel gelten die folgenden Regeln:

* Der Spieler startet auf einer vorgegeben Wikipedia Seite
* Der Spieler muss von dieser Seite aus auf eine vordefinierte Ziel-Seite gelangen.
* Es muss immer der erste Link des Artikels, der weder in Klammern noch kursiv geschrieben ist, angeklickt werden.

Innerhalb dieses Spiels gibt es verschiedene Spiel Varianten:

1. Die gegeben Zielseite muss innerhalb von n Clicks erreicht werden.
2. Zwei Mitspieler geben einander Start und Ziel an, derjenige mit weniger Clicks gewinnt.

# Wikilenium

Wikilenium ist eine Automatisierung des «Philosophen-Wiki», die mit Selenium implementiert wurde.

## Technologien

### Selenium

Die Automatisierung des Spiels erfolgt mit «Selenium». Das Testing-Team hat sich dazu entschieden, die Implementation mit Java zu realisieren und verwendet dazu die Java API für Selenium. Zur Versionskontrolle wurde ein Git Repository eingerichtet und zur Verwaltung der Abhängigkeiten und Automatisierung des Build-Prozesses wird Apache Maven verwendet. Der Testapparat wird vorerst nur für den Google Chrome Browser implementiert, soll aber einfach auf andere Browser erweiterbar sein.

### Allure

Im Projekt gibt es auch automatisierte Tests, welche das korrekte Arbeiten des Testapparats verifizieren sollen. Diese werden jeweils im Build-Prozess des Projektes ausgeführt. Aus den Ergebnissen lassen sich mit «Allure» Reports generieren.

## Systemanforderungen

Aus den oben erwähnten Technologien gehen die folgenden Systemanforderungen her:

* Java 8 oder höher
* Chrome und passender ChromeDriver: Das chromedriver.exe muss unter   
  «C:\Program Files (x86)[\WebDrivers\chromedriver.exe»](file:///\\WebDrivers\\chromedriver.exe) abgelegt werden.

## Annahmen

Das Testing-Team hat die folgenden Annahmen getroffen:

* Es wird nur der erste erwähnte Spielmodus implementiert.
* Die zu verwendende Sprache soll primär Deutsch sein. Der Apparat muss aber einfach auf andere Sprachen erweiterbar sein.
* Der erste gültige Link:
  + Muss im Hauptteil des Artikels sein
  + Darf nicht auf externe Ressourcen verweisen
  + Darf keine Verlinkung einer Ressource sein
  + Darf nicht innerhalb einer Klammer stehen
  + Darf kein Verweis auf eine Fussnote sein
  + Darf nicht kursiv geschrieben sein
* Tests sind erfolgreich, wenn spätestens nach n Clicks das Zeil erreicht ist
* Tests sind erfolgslos, wenn:
  + Die Ziels-Seite nicht nach n Clicks erreicht wurde
  + Eine Endlos-Schleife entdeckt wurde

## Implementation

### Wikilenium-core

Der Kernteil des Testapparats. Er liefert die Möglichkeit eine Startseite, eine Zielseite, die maximale Anzahl der Klicks sowie die Sprache zu definieren. Sofern die Eingaben alle gültig sind, wird der automatisierte Test gestartet. Innerhalb des Test Client Projektes existieren Fix definierte Tests, die als Unit-Tests ausgeführt werden.

Zudem wird das ausführen von Test-Batches mit einem Excel File zur Eingabe und Ausgabe von Testdaten unterstützt.

### Wikilenium-Ui

Wikilenium-Ui ist eine Grafische Oberfläche für Wikilenium-core, welche das Starten von einzelenen Testfällen und Test-Batches erlaubt.

# Entwicklungsvorgehen

## Auftragsanalyse

Der Auftrag wurde zu Beginn in der Gruppe diskutiert und Annahmen getroffen. Dokumentiert wurde noch nichts. Es wurde der Fokus darauf gelegt, zuerst einen lauffähigen Prototypen zu haben.

## Prototyp implementieren

Es wurde ein erster Prototyp mit Java und Selenium entwickelt, welcher auf Links in Wikipedia klicken kann, bis er die Seite Philosophie erreicht. Das Ziel dieses Prototyps war es, die Selenium API kennenzulernen und die API des Testapparates zu definieren.

Der Prototyp erlaubt es einen einzelnen Test auszuführen, mit den Input Daten: Startseite, Maximum Clicks, Zielseite.

## Testapparat verbessern

In einem weiteren Schritt wurde der Prototyp mit zufälligen Startseiten gefüttert und der Pfad geloggt, welchen er genommen hat. Die Pfade mussten manuell überprüft werden, um zu erkennen, ob der Prototyp die «richtigen» Links klickt. Bei erkennbarem Fehlverhalten, wurden automatisierte Testfälle für den Testapparat geschrieben, welche das korrekte Verhalten überprüfen. Dann wurden die Fehler behoben, so dass die automatisierten Tests alle passen.

Weiterhin wurde die API des Testapparats so erweitert, dass die Sprache als Parameter eingegeben werden kann.

## User Interface

Das Starten der Testfälle aus der Entwicklungsumgebung heraus, kam bei der ersten Präsentation nicht gut an. Deshalb wurde ein UI implementiert, über welches man Testfälle starten kann.

## Dokumentation Annahmen

Nach der ersten Präsentation wurde dem Team klar, dass eine Dokumentation der Annahmen fehlt. Das wurde mit diesem Dokument nachgeholt.

## Erweiterung für Test Batches und Excel File

Als eine neue Anforderung hinzugekommen ist, musste der Testapparat erweitert werden. Es wurde eine neue Funktionalität implementiert, um ganze Test Batches auszuführen und Testinputs und Outputs in einem Excel zu lesen bzw. zu speichern. Die Erweiterung war sehr einfach, weil die Code Basis gut modularisiert ist und saubere Schnittstellen hat.