

Business Process Engineering

Wintersemester 2022/2023

Software Testing & Qualitätssicherung 3

Agenda

- Feedback Aufgabe 10
- Test Driven Development
- UI Tests
- Gauge + Taiko



Feedback Aufgabe 10

Negativbeispiel 1: Zu allgemein, keine Testschritte

3 555 - Test Bestellung abschicken	1 Bestellung abschicken	System/Anzeige	Die Bestellung war erfolgreich und wird vom Kunden an Floradise übergeben.
------------------------------------	-------------------------	----------------	--

Negativbeispiel 2: Zu allgemein, keine Testschritte

3 896 - Bestellung abschicken	1 Eine valide Bestellung wird abgeschickt.	geöffneter Strauß-Konfigurator, bereits konfigurierter Strauß im Warenkorb, valide Kundendaten eingegeben
3 896 - Bestellung abschicken	1 Eine invalide Bestellung wird abgeschickt.	geöffneter Strauß-Konfigurator, einmal mit Strauß im Warenkorb + invalide Kundendaten, einmal mit leerem Warenkorb

Die Bestellung wird erfolgreich abgeschlossen und eine Bestätigung wird gesendet.
Die Bestellung kann nicht abgeschlossen werden.

Feedback Aufgabe 10

Negativbeispiel 3 (schon besser): Zu wenig Details

1 555 - Test des Strauß-Konfigurators	1 Strauß-Konfigurator aufrufen	Website	Der Strauß-Konfigurator wurde erfolgreich aufgerufen.
1 555 - Test des Strauß-Konfigurators	2 Strauß konfigurieren (inkl. aller möglichen Konfigurationsmöglichkeiten)	Strauß-Konfigurator	Der Strauß wurde erfolgreich konfiguriert.
1 555 - Test des Strauß-Konfigurators	3 Strauß fehlerhaft nach erster Konfiguration bearbeiten.	Strauß-Konfigurator	Bei der Überarbeitung ihres Straußes fehlt mindestens eine Auswahlmöglichkeit.
1 555 - Test des Strauß-Konfigurators	4 Strauß nach erster Konfiguration bearbeiten	Strauß-Konfigurator	Der Strauß wurde erfolgreich überarbeitet.
1 555 - Test des Strauß-Konfigurators	5 Strauß in Warenkorb legen	Warenkorb	Der Strauß wurde in den Warenkorb gelegt.

Feedback Aufgabe 10

Musterlösung Teilaufgabe

External Id	Name	Step	Te	Result
2 333 - Strauß konfigurieren (ohne Voreinstellung)		1 Öffnen Sie die Website www.floradise.com		Es öffnet die Floradise-Website mit dem Online-Shop
2 333 - Strauß konfigurieren (ohne Voreinstellung)		2 Klicken Sie auf das Burgermenü auf der rechten Seite		Ein Menü mit verschiedenen Punkten wird auf der rechten Seite angezeigt
2 333 - Strauß konfigurieren (ohne Voreinstellung)		3 Klicken Sie auf 'Straußkonfigurator'		Es öffnet die Straußkonfigurator-Seite, bis auf einen Start-Button ist noch nichts sichtbar
2 333 - Strauß konfigurieren (ohne Voreinstellung)		4 Klicken Sie auf 'Start'		Es öffnet der Straußkonfigurator ohne Vorauswahl, es werden nun alle Felder und Bereiche angezeigt
2 333 - Strauß konfigurieren (ohne Voreinstellung)		5 Wählen Sie über den ^ Button neben der rosa Rose eine Anzahl von 3 aus		Es werden 3 rosa Rosen in der 3D-Vorschau angezeigt, der Preis aktualisiert sich auf 3€
2 333 - Strauß konfigurieren (ohne Voreinstellung)		6 Wählen Sie über den ^ Button neben dem Eukalyptus eine Anzahl von 5 aus		Es werden zusätzlich zu den 3 rosa Rosen in der 3D-Vorschau nun auch 5 Eukalytus-Zweige angezeigt, der Preis aktualisiert sich auf 6€
2 333 - Strauß konfigurieren (ohne Voreinstellung)		7 Klicken Sie im Bereich Vase auf 'nein'		Die 3D-Ansicht sowie der Preis bleiben unverändert, der Button 'nein' ist nun grün umrandet
2 333 - Strauß konfigurieren (ohne Voreinstellung)		8 Klicken Sie im Bereich Papiermanschette auf 'Keine Manschette'		Die 3D-Ansicht sowie der Preis bleiben unverändert, der Button 'Keine Manschette' ist nun grün umrandet
2 333 - Strauß konfigurieren (ohne Voreinstellung)		Klicken Sie im Bereich Karte auf 'ja' und geben Sie in das Feld Text die Worte		In der 3D-Ansicht wird die Vorschau einer Karte ergänzt, der Text wird in kleiner Schrift mit
2 333 - Strauß konfigurieren (ohne Voreinstellung)		9 'Herzlichen Glückwunsch zum Geburtstag' ein		angezeigt. Der Preis aktualisiert sich auf 9€
2 333 - Strauß konfigurieren (ohne Voreinstellung)		10 Klicken Sie im Bereich Karte auf die Karte mit den Luftballons		In der 3D-Ansicht wird die Vorschau der Karte zu der Karte mit den Luftballons geändert, der Text wird in kleiner Schrift weiterhin angezeigt. Der Preis bleibt bei 9€
2 333 - Strauß konfigurieren (ohne Voreinstellung)		11 hinzuzufügen		Neben dem Warenkorb-Symbol erscheint nun ein roter Kreis mit einer 1
2 333 - Strauß konfigurieren (ohne Voreinstellung)		12 Klicken Sie auf den Button Warenkorb anzeigen		Der Warenkorb wird angezeigt, darin befindet sich der zuvor konfigurierte Strauß

Test Driven Development

- Deutsch: Testgesteuerte Entwicklung
- Methode, die häufig bei der agilen Entwicklung eingesetzt wird
- Programmierer erstellt Softwaretests **vor** den zu testenden Komponenten

- Test First
 - Unit Tests werden vor dem eigentlichen Computerprogramm geschrieben
 - Mehrere fehlgeschlagene Unit Tests dürfen gleichzeitig existieren
 - Vorstufe der testgetriebenen Entwicklung



Test Driven Development nach Kent Beck

- Unit-Tests und deren Units werden parallel entwickelt
- Entwicklung in *Mikroiterationen* (wenige Minuten)
 - Drei Hauptteile (Red-Green-Refactor)
 - Red – Test für eine neue Funktionalität schreiben
 - Green – Programmcode mit möglichst niedrigem Aufwand ändern
 - Refactoring – Code „aufräumen“
- Schritte wiederholen, bis:
 - bekannte Fehler bereinigt sind
 - der Code die gewünschte Funktionalität liefert
 - dem Entwickler keine sinnvollen weiteren Tests einfallen
- Nachteil: hoher Aufwand



UI Tests

- Es werden folgende Aspekte geprüft:
 - Visuelles Design
 - Funktionalität
 - Usability
 - Performance
 - Compliance
- Prüfen wie die Applikation Anwenderinteraktionen ausführt
 - Mit Maus- und Tastatureingaben etc.
- Prüfen, ob visuelle Elemente richtig dargestellt werden und funktionieren

UI Tests spielen eine wichtige Rolle für die Produktivsetzung einer Applikation!

UI Tests

- Manuell
 - Ein User interagiert mit der UI, basierend auf definierten Testfällen.
 - Die Ergebnisse werden dokumentiert.
- Record and Playback Testing
 - Ein User führt einmal die Interaktionen mit der UI aus, welche aufgezeichnet werden.
 - Die Aufzeichnung wird in eine Skript umgewandelt, dieses kann automatisiert ausgeführt werden.
- Model based
 - Ein Modell für das System erstellen
 - Eingaben definieren
 - Erwartete Ergebnisse verifizieren
 - Tests ausführen
 - Tatsächliche Ergebnisse prüfen und abgleichen

UI Testing Checklist:

- Data type errors
- Feldlänge
- Navigationselemente
- Fortschrittsleisten
- Type ahead
- Table scrolling
- Error logging
- Menu items
- Working shortcuts
- Confirm action buttons
- ...

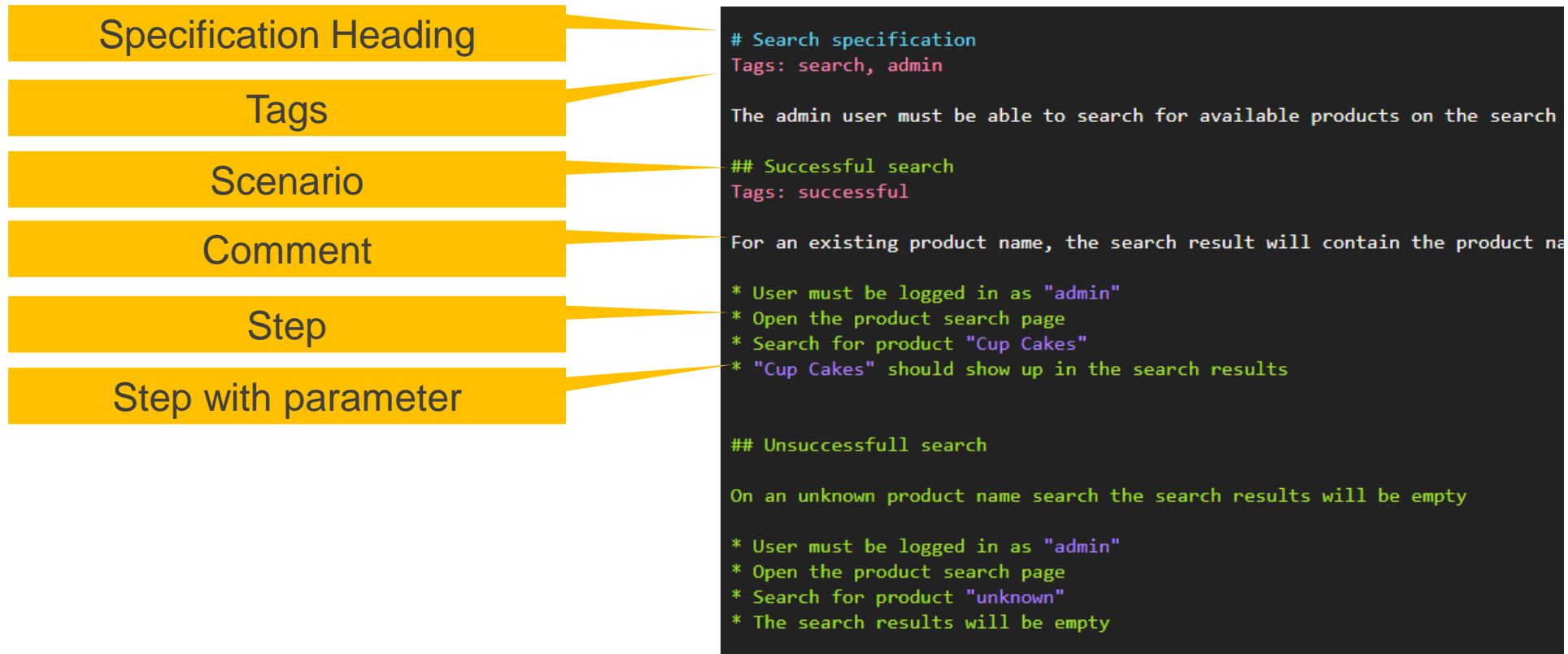
UI Testing Frameworks: Gauge



- Kostenloses Open-Source Framework zum schreiben und durchführen von Acceptance Tests
- Nutzt Markdown Syntax
- Plattform / Sprachübergreifend
- Tests können prinzipiell auch von “Nicht Entwicklern” geschrieben werden
- Tests können manuell ausgeführt werden → Testprotokoll

<https://gauge.org/>

Specification



Specification Heading

- Muss genau ein Mal vorhanden sein

```
# Spec Heading
```

oder

```
Spec Heading  
=====
```

Scenario

- Repräsentiert einen Workflow in einer Specification
- Es muss mindestens ein Scenario geben

```
## Scenario heading
```

oder

```
Scenario heading  
-----
```

Step

- Ausführbare Komponente einer Specification
- Kann in einem Scenario oder außerhalb existieren

```
* Search for product "Cup Cakes"  
* "Cup Cakes" should show up in the search results
```

Parameters



- Simple Parameters
- Dynamic Parameters
 - Werden als Platzhalter benutzt, wenn man sich auf eine Tabellenspalte bezieht
- Table Parameters
 - Werden benutzt, wenn ein Step für mehrere Werte einer Tabelle ausgeführt wird
- Special Parameters
 - So können auch Tabellen und Files zu Parametern werden

Parameters



- Simple Parameters
- Dynamic Parameters
 - Werden als Platzhalter benutzt, wenn man sich auf eine Tabellenspalte bezieht
- Table Parameters
 - Werden benutzt, wenn ein Step für mehrere Werte einer Tabelle ausgeführt wird
- Special Parameters
 - So können auch Tabellen und Files zu Parametern werden

```
* Check "product 1" exists
* Check "product 2" exists
```


Parameters



```
# Create projects

|id| name |
|--|-----|
|1| Alice|
|2| Bob  |
|3| Eve  |

## First scenario
* Say "hello" to <name>.

## Second scenario
* Say "namaste" to <name>.
```

- Simple Parameters
- Dynamic Parameters
 - Werden als Platzhalter benutzt, wenn man sich auf eine Tabellenspalte bezieht
- Table Parameters
 - Werden benutzt, wenn ein Step für mehrere Werte einer Tabelle ausgeführt wird
- Special Parameters
 - So können auch Tabellen und Files zu Parametern werden



Parameters

- Simple Parameters
- Dynamic Parameters
 - Werden als Platzhalter benutzt, wenn man sich auf eine Tabellenspalte bezieht
- Table Parameters
 - Werden benutzt, wenn ein Step für mehrere Werte einer Tabelle ausgeführt wird
- Special Parameters
 - So können auch Tabellen und Files zu Parametern werden

```
# Create projects

## First scenario

* Create the following projects
  |project name| username |
  |-----|-----|
  | Gauge java | Daredevil|
  | Gauge ruby | Iron Fist|
```



Parameters

```
* Verify email text is <file:email.txt>
```



- Simple Parameters
- Dynamic Parameters
 - Werden als Platzhalter benutzt, wenn man sich auf eine Tabellenspalte bezieht
- Table Parameters
 - Werden benutzt, wenn ein Step für mehrere Werte einer Tabelle ausgeführt wird
- **Special Parameters**
 - So können auch Tabellen und Files zu Parametern werden.
Syntax: <prefix:value>

Tags

- Helfen beim Filtern und Sortieren von Specifications und Scenarios

```
# Search specification
  Tags: search, admin

## Successful search
  Tags: successful
```

Comments

- Alles ohne Präfix ist ein Comment

```
# Search specification
Tags: search, admin

The admin user must be able to search for available products on the search

## Successful search
Tags: successful

For an existing product name, the search result will contain the product name
```

Step Implementierung



```
* Say "hello" to "gauge"
```

Gauge Step



```
step("Say <greeting> to <name>", function(greeting, name, done) {
  try {
    setTimeout(function () {
      // Code for step
      done();
    }, 1000);
  } catch(e) {
    done(e);
  }
});

// Handling errors in promises with done
step("Say <greeting> to <name>", function(greeting, name, done) {
  // Let promise1 be some promise we need to wait for.
  promise1
    .then(done)
    .catch(function(e) { done(e);});
});
```

Step Implementation in JavaScript

Gauge Live Demo

```
specs > example.spec > ...
Run Spec | Debug Spec
1 # Getting Started with Gauge
2
3 This is an example markdown specification file.
4 Every heading in this file is a scenario.
5 Every bulleted point is a step.
6
7 To execute this specification, use
8 + npm test
9
10 This is a context step that runs before every scenario
11 * Open todo application
12
13 Run Scenario | Debug Scenario
14 ## Display number of items
15 * Add task "first task"
16 * Must display "1 item left"
17 * Add task "second task"
18 * Must display "2 items left"
19
20 Run Scenario | Debug Scenario
21 ## Must list only active tasks
22 * Add tasks
23
24 -- |description|
25 -- |-----|
26 -- |first task |
27 -- |second task|
28 -- |third task |
29 -- |fourth task|
30 -- |fifth task |
```



gauge

		Failed	Passed	Skipped
TOTAL SPECS	1	0	1	0
TOTAL SCENARIO	2	0	2	0

Specifications

Name	Execution time
Getting Started with Gauge	00:00:15

Getting Started with Gauge 00:00:15

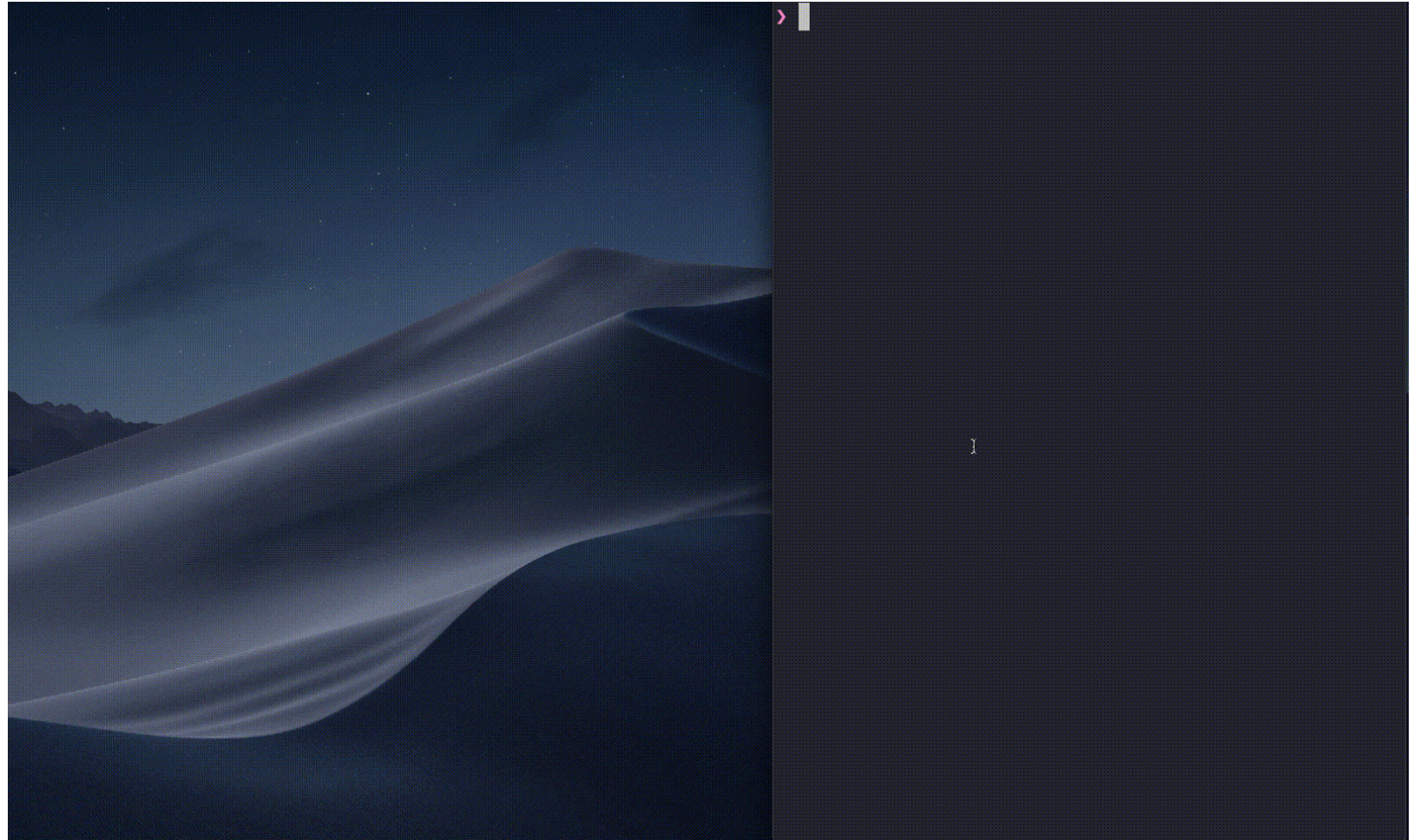
This is an example markdown specification file. Every heading in this file is a scenario. Every bulleted point is a step.
To execute this specification, use npm test
This is a context step that runs before every scenario

- Display number of Items** 00:00:05
 - Execution Time : 00:00:03
 - Open todo application
 - Execution Time : 00:00:01
 - Add task "first task"
 - Execution Time : 00:00:00
 - Must display "1 item left"
 - Execution Time : 00:00:01
 - Add task "second task"

Taiko

“Taiko is free and open source [Node.js](https://nodejs.org/) library with a simple API to automate Chromium based browsers (Chrome, Microsoft Edge, Opera) and Firefox.”

<https://taiko.dev/>



Quelle: <https://taiko.dev/>

Taiko

- API für „Blackbox Testing“
 - Kein Blick in den Seiten Quelltext nötig – meistens 😊
- Skripte die direkt auf einer Website ausgeführt werden
 - „Headless“ oder „Observed“

```
1  const { openBrowser, goto, write,  
2    click, closeBrowser, textBox } = require('taiko');  
3  
4  (async () => {  
5    await openBrowser();  
6    await goto("google.com");  
7    await write("taiko test automation");  
8    await press("Enter");  
9    await closeBrowser();  
10 })();
```



```
$ npx taiko tyko-demo.js  
[PASS] Browser opened  
[PASS] Navigated to URL http://google.com  
Error: Element focused is not writable  
    at C:\Users\asc\Desktop\gauge-test\uniks-bpe\node_modules\taiko\lib  
    at async waitAndGetActionableElement (C:\Users\asc\Desktop\gauge-te  
    at async write (C:\Users\asc\Desktop\gauge-test\uniks-bpe\node_modu  
    at async module.exports.write (C:\Users\asc\Desktop\gauge-test\unik  
    at async module.exports.<computed> (C:\Users\asc\Desktop\gauge-test  
    at async global.<computed> (C:\Users\asc\Desktop\gauge-test\uniks-b  
    at async C:\Users\asc\Desktop\gauge-test\uniks-bpe\tyko-demo.js:7:5  
[PASS] Browser closed
```

Taiko Live Demo



Taiko vs. Selenium

Taiko

- Chrome DevTools Protokoll
- Kein Support für WebKit basierte Browser (z.B. Safari)
- Kein Support für mobile Applikationen
- Ausschließlich JavaScript Unterstützung

Selenium

- WebDriver
- Support beliebiger Browser
- Support für viele Sprachen, z.B. Java, Python, Ruby, PHP etc.
- Support für GUI basiertes Recording



Aufgabe 11

- Gauge + Taiko (Testen der Applikation aus Woche 8/9)
- **Deadline:**
 - Montag, 31.01.2022, 14:00 Uhr (1. Abgabe)
 - Donnerstag, 03.02.2022, 14:00 Uhr (2. Abgabe)



Quellen

- <https://deepsources.io/blog/exponential-cost-of-fixing-bugs/>
- <https://blog.seibert-media.net/blog/2011/05/16/software-tests-notwendigkeit-arten/>
- <https://www.oliver-lampert.at/glossar/testarten/>
- <https://www.testingexcellence.com/seven-principles-of-software-testing/>
- <https://comquent.de/de/die-sieben-grundsaeetze-des-softwaretestens/>
- <https://www.youtube.com/watch?v=3bJcvBLJViQ>
- <https://www.youtube.com/watch?v=Wi75S5TTfQ0>
- <https://blog.christianposta.com/deploy/blue-green-deployments-a-b-testing-and-canary-releases/>
- <https://de.wikipedia.org/wiki/Datei:Kde-bugtracking-via-bugzilla-firefox-1.0.6-kde-3.4.2-de.png>
- <https://www.redmine.org/projects/redmine/wiki/RedmineIssueList>
- <https://trac-hacks.org/wiki/ChildTicketsPlugin/Examples>
- <https://taiko.dev>
- <https://gauge.org>



The background features a large, stylized cloud shape composed of overlapping translucent blue and white polygons. A network of white lines connects various points within and around the cloud, with some nodes highlighted in orange, green, and yellow. The background is split vertically into a dark blue left half and a light blue right half, with several small, glowing circular particles scattered across the scene.

Business Process Engineering

Wintersemester 2022/2023

Dr. Andreas Scharf