

# Klausur "Entwicklung Webbasierter Anwendungen", Sommersemester 2023

Prof. Dr. Ralf Hahn

## Hinweise:

- **Bitte schalten Sie Ihr Handy komplett aus, um das WLAN zu entlasten!**
- Bearbeitungszeit: 90 Minuten + 10 Minuten für Down- und Upload. Es gibt 100 Punkte in 5 Aufgaben.
- **Abgabe am 19.07.23 bis spätestens 17:45 Uhr über Git in Ihrem persönlichen Repository für die Prüfung.**
- Erlaubte Hilfsmittel: Ein Notebook pro Person und jede auf Papier gedruckte Quelle. Kommunikation mit anderen Menschen außer der Prüfungsaufsicht ist nicht erlaubt! Die Verwendung von Chats o.ä. zur Kommunikation mit anderen Personen wird als Täuschungsversuch gewertet.
- Lesen Sie zuerst die gesamte Aufgabenstellung bevor Sie mit der Lösung der Aufgaben beginnen.
- Bearbeiten Sie die Aufgabe im Ordner **src/Exam** des Repositories, das ihnen zugewiesen wurde. Legen Sie keine neuen Dateien an und benennen Sie die Dateien nicht um. Committen bzw. pushen Sie ihren Code via Git regelmäßig (ca. alle 10 Minuten) in Ihr zugeteiltes Repository. Die Abgabe erfolgt ausschließlich über Git. Commits mit einem Zeitstempel nach Ablauf der Bearbeitungszeit können nicht berücksichtigt werden.
- **Es wird nur bewertet, was sich in dem für Sie angelegten Gitlab-Repository in den vorbereiteten Dateien bis zum Abgabetermin unter src/Exam befindet.**
- Schreiben Sie standardkonformen Code, der zusätzlich die Regeln der Veranstaltung für ordentlichen Code und die Vorgaben durch die Seitenklassen berücksichtigt. Verwenden Sie Exceptions zur Behandlung von ungewöhnlichen Fehlern. Entwickeln Sie eine barrierefreie, sichere Anwendung.
- In den PHP-Seitenklassen ist die strenge Typüberprüfung aktiviert. Diese Überprüfung darf nicht abgeschaltet oder umgangen werden. Ändern Sie auch nicht die Signaturen der vordefinierten Methoden.
- Die Verwendung von Frameworks (z.B. zur Erleichterung der DOM-Operationen) ist nicht erlaubt.
- Sie können sowohl in Deutsch als auch in Englisch antworten (You may also answer in English).
- Diese Aufgabenblätter werden nicht abgegeben. Schreiben Sie also keine Lösungsteile darauf. Sie dürfen die Seiten natürlich auch trennen.

**Wichtig! Schreiben Sie zuerst den Code und versuchen Sie erst am Schluss den Code zum Laufen zu bringen!**

## Zulieferung:

In Ihrem GitLab-Projekt sind die Dateien, die Sie bearbeiten sollen, schon angelegt. Im Moodlekurs der Veranstaltung finden Sie eine ZIP-Datei mit vorbereiteten Inhalten für diese Dateien sowie einem SQL-Skript zum Erzeugen und Füllen der Datenbank. Kopieren Sie zunächst die enthaltenen Dateien in das Verzeichnis src/Exam und überschreiben Sie dabei vorhandene Dateien. Wichtig sind folgende Dateien:

- Page.php: Die bekannte Basisklasse der Seitenklassen (Sie müssen diese Datei nicht mehr bearbeiten!)
- Exam23.html - hier lösen Sie Aufgabe 1
- Exam23API.php - hier lösen Sie Aufgabe 2
- Exam23.js - hier lösen Sie Aufgabe 3
- Exam23.css - hier lösen Sie Aufgabe 4
- Exam23.sql - diese Datei müssen Sie als root (!!!) mit phpMyAdmin importieren, damit die neue Datenbank angelegt wird

**Nochmal! Schreiben Sie zuerst den Code und versuchen Sie erst am Schluss den Code zum Laufen zu bringen!**

## Aufgabe 1: HTML (24 Punkte)

Erstellen Sie in der Datei **Exam23.html** eine valide, statische und unformatierte HTML-Seite (keine PHP-Seite, keine Seitenklasse, kein Design und auch noch kein Layout!) mit den Inhalten aus nebenstehender Abbildung.

Strukturieren Sie die Baumarkt-Seite mit Hilfe geeigneter semantischer HTML(5)-Elemente! Der Hauptbereich des HTML-Dokuments besteht aus der Navigation sowie aus den Abschnitten „Summer Special“ und „Newsletter“. Der Text "Bei zwei linken..." soll immer ganz unten auf der Seite stehen.

Hinweise:

- Sie sollen für diese Seite wirklich nicht die Seitenklassen verwenden
- Der Browser-Tab soll "OPIs Baumarkt" anzeigen
- Die Navigationselemente sollen ohne echte Funktion auf die Seite Exam23.html verweisen
- Im Formularbereich sollen alle Eingabefelder Pflichtfelder sein. Verwenden Sie passende Formularelemente, um die Eingabe möglichst automatisch zu überprüfen.
- Beachten Sie, dass das Formular zwei Knöpfe beinhaltet, um die Daten abzuschicken. Der Knopf "Absenden" soll das Formular mit seinen Daten an Exam23API.php abschicken (siehe Aufgabe 2). Dadurch können Sie später das AJAX-Backend auch ohne JavaScript testen.
- Der Knopf "Absenden via AJAX" soll eine JavaScript-Methode `sendData()` aufrufen (welche in Aufgabe 3 die Daten via AJAX abschickt) ohne die Seite neu zu laden (Das HTML-Formular soll in diesem Fall nicht abgeschickt werden! Tipp: Verwenden Sie nicht `<button>`).

## Aufgabe 2: PHP (26 Punkte)

Implementieren Sie den serverseitigen Teil zur Verarbeitung der Newsletter-Anmeldung als Seitenklasse in PHP in der Datei **Exam23API.php**. Die zugelieferte Datei Exam23API.php enthält das bekannte PageTemplate der Seitenklassen. Die zugelieferte Page.php beinhaltet bereits die üblichen, benötigten Methoden und die Datenbankanbindung (Diese Datei müssen Sie nicht mehr bearbeiten!).

Realisieren Sie den Service so, dass einkommende Newsletter-Anmeldungen sinnvoll verarbeitet werden. Die empfangenen Daten sollen in die Datenbank eingefügt werden. (Sie dürfen das Problem von Einträgen mit gleicher Mailadresse ignorieren). Falls eine Mailadresse bereits vorhanden ist, wird trotzdem ein neuer Eintrag erzeugt. Wenn die Adresse erfolgreich in die Datenbank eingetragen wurde, liefern Sie den Kundennamen und die Emailadresse als JSON-Antwort zurück. Ansonsten ignorieren Sie den AJAX-Aufruf.

Hinweise:

- Achten Sie wie üblich auf Sicherheit
- Prüfen Sie serverseitig, ob es sich wirklich um eine gültige Mailadresse handelt.  
Tipp: Die PHP-Funktion `filter_var($email, FILTER_VALIDATE_EMAIL)` liefert `true`, wenn `$email` eine standardkonforme Emailadresse enthält.
- Hinweis: Solange Aufgabe 3 noch nicht implementiert ist, kann das Abschicken des Formulars mit dem "Absenden"-Knopf (vgl. Aufgabe 1) geprüft werden. Die gesendeten JSON-Daten werden dann in Rohform angezeigt.

## Aufgabe 3: JavaScript (25 Punkte)

- a) Binden Sie die Datei **Exam23.js** ein und implementieren Sie die Newsletter-Anmeldung mit AJAX.  
Bei einem Klick auf den Button „Absenden via AJAX“ wird die JavaScript-Funktion `sendData()` aufgerufen (vgl. Aufgabe 1). Diese Funktion liest die Formulardaten aus den Eingabefeldern und schickt sie mittels AJAX an das PHP-Skript `Exam23Api.php` aus Aufgabe 2. Beachten Sie die zugelieferten Funktionen `sendRequest()` und `processData()`, die das Verarbeiten von AJAX-Requests erleichtern.
- b) Überprüfen Sie die Gültigkeit der Eingaben vor dem Abschicken des AJAX-Requests und schicken Sie den Request nur ab, wenn die Eingaben in Ordnung sind! Zur Überprüfung bietet JavaScript zwei Methoden, die auch beim "normalen" Abschicken eines Forms genutzt werden:
- `X.checkValidity()` prüft für den DOM-Knoten eines HTML-Eingabeelements X, ob die in HTML festgelegten Vorgaben für das Formularelement erfüllt sind und liefert entsprechend true bzw. false.
  - `X.reportValidity()` zeigt für X die bekannten Hinweis-Popups, falls die Eingabe nicht korrekt ist.
- c) Bei erfolgreicher Verarbeitung sollen clientseitig die zurückgelieferten JSON-Daten (NAME und EMAIL) zur Bestätigung angezeigt werden. Zeigen Sie dann statt des Formularbereichs den folgenden Block an:
- „Vielen Dank, NAME, dass Sie unseren Newsletter abonniert haben. Dieser wird an folgende Mailadresse geschickt werden: EMAIL“.

### Newsletter

Vielen Dank, Ewa Musterfrau, dass Sie unseren Newsletter abonniert haben.  
Dieser wird an folgende Mailadresse geschickt werden: test@ewa.com

## Aufgabe 4: CSS (15 Punkte)

Erstellen Sie in **Exam23.css** mit Hilfe von Media Queries ein responsives Layout der Baumarkt-Seite. Implementieren Sie einen Breakpoint bei 800px, der zwischen 2 Layout-Varianten umschaltet. Nutzen Sie **flexbox** zur Anordnung und setzen Sie das Layout gemäß der Abbildungen um.

- a) Das Layout für breite Displays soll 3 spaltig sein mit den Navigationselementen links als eigene Spalte.

- b) Das Layout für schmale Displays soll einspaltig sein (siehe Abbildung in Aufgabe 1). Die Navigationselemente liegen oben und nebeneinander.

Definieren Sie die abgebildeten Hintergrundfarben, Rahmen und Ausrichtungen. Sonstige Designelemente wie z.B. Abstände dürfen Sie ignorieren. Wenn Sie geeignete HTML-Elemente gewählt haben, sollte das Default-Design in Ordnung sein.

## Aufgabe 5: Lauffähigkeit / Vollständigkeit (10 Punkte)

**Warnung!** Es kann lange dauern, einen kleinen Fehler zu finden und es gibt nur wenige Punkte!

**Gehen Sie diese Aufgabe erst an, wenn Sie die übrigen Aufgaben nach bestem Wissen gelöst haben.**

- a) Prüfen Sie den HTML- und CSS-Teil (auch ohne Docker), indem Sie die HTML-Datei im Browser öffnen. Vergleichen Sie die angezeigten Informationen mit den Abbildungen. Überprüfen Sie, ob der Code valide und barrierefrei ist. Entfernen Sie Probleme, die der Linter anzeigt (Meldungen bis Schwierigkeitsgrad 2 (z.B. zu AccessKey) dürfen Sie ignorieren).
- b) Funktionieren die 2 Layouts mit CSS? Verkleinern Sie das Browserfenster bis das Layout umschaltet.
- c) Damit der JavaScript- und Ajax-Code laufen kann, müssen Sie die Docker-Container starten.
  - Schicken Sie das Formular auf Ihrer HTML-Seite mit dem Knopf "Absenden" ab. Wird eine Browserseite mit den empfangenen JSON-Daten angezeigt?
  - Schicken Sie das Formular auf Ihrer HTML-Seite mit dem Knopf "Absenden via AJAX" ab. Wird auf der Seite (ohne Neuladen) der Bestätigungstext angezeigt? Wird das Formular ausgeblendet?

## Abgabe

Laden Sie Ihr Endergebnis in Git hoch und prüfen Sie über GitLab (<https://code.fbi.h-da.de>), ob die gewünschten Dateien auch wirklich angekommen sind.