Otto-Friedrich-Universität Bamberg
Lehrstuhl für Medieninformatik
Prof. Dr. Andreas Henrich
Leon Martin, Tobias Hirmer, Martin Bullin

VAWi Web-Technologien, Sommersemester 2022

Studienleistung 2

Hinweise

- Die vollständige und korrekte Bearbeitung einer Aufgabe ergibt die volle Punktzahl dieser Aufgabe; in Ausnahmefällen werden halbe Punkte verteilt.
- Für diese Studienleistung erwarten wir einen **Bearbeitungsaufwand von insgesamt etwa 60 Stunden**, da wir von drei Personen pro Team ausgehen. Dies ist als Richtwert zu verstehen und keinesfalls als Unter- oder Obergrenze.
- In Verdachtsfällen behalten wir uns vor, Plagiate von der Benotung auszunehmen. Wir werden in einem solchen Fall Kontakt mit den Betroffenen aufnehmen.
- *Programmieraufgaben* sollen in nachvollziehbar kommentierter Form im Quelltext abgegeben werden. Verwenden Sie keine anderen als die angegebenen Bibliotheken, soweit sie nicht zum Standardsprachumfang gehören. Nicht lauffähige Programme können nicht bewertet werden. Fügen Sie gegebenenfalls eine README-Datei hinzu, die beschreibt, wie man Ihre Lösung kompilieren bzw. ausführen kann.

Abgabe der Lösungen

- *Abgabetermin* **05. Juni 2022, 23:55 Uhr**
- *Abgabemodus* Einreichung auf der Online-Lernplattform im entsprechenden Kurs; der Server ist erreichbar unter `http://lms.vawi.de/vawi/`.
- *Dateiformat der Abgabe* Die Abgabe erfolgt durch den Upload eines ZIP-Archivs, das alle notwendigen Dateien beinhaltet. Der Dateiname des ZIP-Archivs sollte das Namensschema **nachname1_nachname2_nachname3_02.zip** einhalten.
- *Update der Lösung* Bis zur oben angegebenen Deadline können Sie Ihre Lösung beliebig oft durch eine neue (korrigierte) Fassung ersetzen. Bitte beachten Sie, dass dabei die zuvor hochgeladene Lösung überschrieben wird. Wir haben keine Möglichkeit, alte Fassungen wiederherzustellen!

Wir wünschen Ihnen viel Erfolg
bei der Bearbeitung der Studienleistung!

Aufgabe: Erstellen eines One-Pagers zur Visualisierung von Schlagzeilen (45 Punkte)

Aufgrund der vielen parallel stattfindenden Ereignisse auf der Welt wollen Sie sich einen Überblick über aktuelle Schlagzeilen verschaffen. Bei der Suche nach einer passenden Webseite fällt Ihnen auf, dass sich die in WebT erlernten Inhalte bestens dafür eignen, selbst einen solchen Service zu erstellen. Bei der weiteren Recherche sind Sie auf einen API-Endpunkt gestoßen (`https://web-t.l0e.de/tl2/news/`), mit dem Sie die Schlagzeilen eines bestimmten Tages abrufen können. Sie entschließen sich dazu eine Webseite zu erstellen, welche die Schlagzeilen (zunächst nur für einen bestimmten Tag, nämlich den 10.05.2022) von dieser API abfragt und ansprechend präsentiert. Während der Vorüberlegungen zur Webseite haben Sie sich das folgende Mockup überlegt:

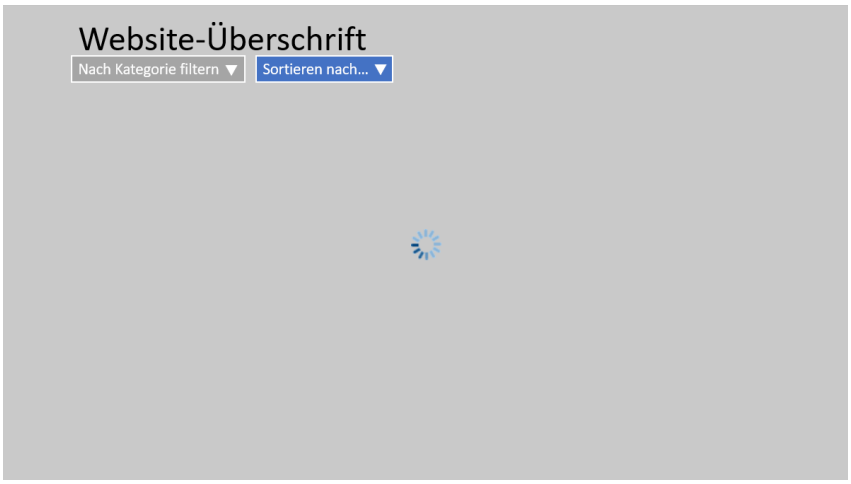

Abbildung 1: Immer wenn Daten von der API geladen werden, soll auf der Webseite ein Lade-Symbol angezeigt werden.

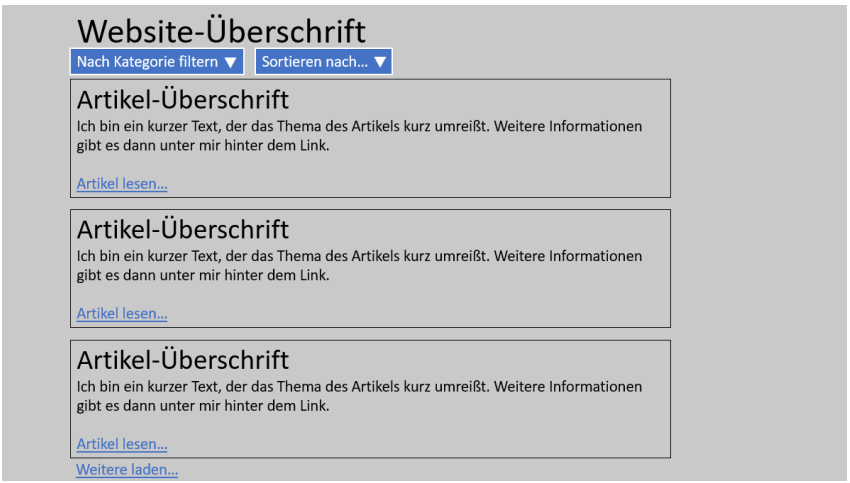

Abbildung 2: Das Aussehen der Webseite, nachdem die Daten geladen wurden.

API-Dokumentation

Die gegebene API, erreichbar unter `https://web-t.l0e.de/tl2/news/`, stellt folgende für die Bearbeitung der Teilleistung wichtige Endpunkte bereit:

`/`	Alle Schlagzeilen des 10.05.2022
`/tags`	Liste aller Kategorien
`/tag/{tag}`	Alle Schlagzeilen zu einer bestimmten Kategorie

Dateiübersicht

Für die Teilleistung wird die Datei `tl2.zip` mit dem folgenden Inhalt im Virtuellen Campus bereitgestellt:

`js`-**Ordner**	Der Ordner, in dem sich Ihre JavaScript-Dateien befinden sollen. Beinhaltet schon die Klasse `ApiService`, die implementiert und genutzt werden soll.
`css`-**Ordner**	Der Ordner, in dem sich Ihre CSS-Dateien befinden sollen.
`index.html`-**Datei**	Das HTML-Template für Ihre Webseite.

Vorgehen und Anforderungen

- Verwenden Sie das vorgegebene Template und behalten Sie die Ordnerstruktur bei. Die Erstellung zusätzlicher HTML-Dateien ist nicht erlaubt.
- Zur Gestaltung soll ausschließlich eigener CSS-Code verwendet werden. Die oben dargestellten Mockups dienen lediglich zur Veranschaulichung des Aufbaus der Webseite. Bei der sonstigen Gestaltung der Webseite (Farben, Fonts, Schriftgrößen etc.) sind Sie frei. Achten Sie jedoch darauf, dass Ihre Webseite optisch ansprechend und gut nutzbar ist.
- Ihre Webseite soll wie folgt strukturiert sein: Ganz oben soll sich eine Überschriften befinden. Darunter sollen sich die Filter- und Sortierschaltflächen befinden. Danach kommen die einzelnen Artikel und anschließend die Schaltfläche, um weitere Artikel zu laden bzw. anzuzeigen.
- Die Überschrift und die oberen Schaltflächen sollen stets “mitscrollen”, also fixiert sein.
- Das Anfordern der Daten soll mit asynchronen HTTP-Requests per Fetch API umgesetzt werden. Berücksichtigen Sie auch den Fall, dass die angeforderte Datei bzw. die angefragte Route nicht zur Verfügung steht und geben Sie dann eine Fehlermeldung aus.
- Beim Aufruf der Webseite soll ohne Nutzerinteraktion der Request zum Holen der Daten der Artikel und der verfügbaren Kategorien ausgeführt werden. Während auf die Antwort gewartet wird, soll an Stelle der Artikelübersicht ein Spinner angezeigt werden (vgl. Abb. 1). Dieser soll ausschließlich mit HTML/CSS implementiert werden.
- Sobald die Antwort des Servers verfügbar ist, sollen die neuesten 10 Artikel dargestellt werden.

- Mit einem Klick auf die Schaltfläche “Weitere Laden” sollen 10 weitere Artikel an die Liste angehängt werden. Dafür ist bei richtiger Datenhaltung kein weiterer Request verfügbar.
- Der Nutzer soll über das erste Drop-Down-Menü eine der Kategorien oder “Keine” auswählen können. Durch einen weiteren Request sollen nun die Artikel der Kategorie (oder im Falle von “Keine” alle Artikel) geladen und auch wieder die ersten 10 angezeigt werden.
- Der Nutzer soll über das zweite Drop-Down-Menü die bereits abgerufenen Daten sortieren können. Hierbei darf kein neuer Request gestartet werden. Es soll möglich sein, die Nachrichten nach der Uhrzeit, dem Titel und dem Tag **absteigend** zu sortieren. Standardmäßig sind die Daten, die vom Server kommen, nach der Uhrzeit sortiert.
- Pro Artikel sollen die Überschrift und der Text dargestellt werden. Den Link unter der Schaltfläche “Artikel lesen...” entnehmen Sie ebenso der API.

Webserver

Um lokale AJAX-Requests durchführen zu können und dabei keine Fehlermeldungen zu erhalten, benötigen Sie einen Webserver. Verwenden Sie die Live-Server-Extension von Visual Studio Code. Alternativ können Sie auch das Apache-Modul der neuesten Version von XAMPP als Webserver verwenden. XAMPP ist für alle gängigen Betriebssysteme unter `https://www.apachefriends.org/de/index.html` verfügbar. Geben Sie in Ihrer README-Datei an, wie Ihre Lösung ausgeführt werden kann.

Weitere Hinweise

- Nutzen Sie die Möglichkeiten der Konsolenansicht in den Entwicklungswerkzeugen von Firefox und Chrome zum Debuggen Ihres JavaScript-Codes.
- Erstellen Sie je eine CSS- und JavaScript-Datei für Ihre Abgabe. Nutzen Sie zudem die vorgegebene `index.html`.
- Verwenden Sie keine JavaScript-Bibliotheken (auch kein jQuery!).
- Testen Sie das von Ihnen implementierte Event Handling auf möglich Fehler, die durch ungewöhnliche Nutzerinteraktionen entstehen könnten.
- Kapseln Sie zusammengehörige Funktionalität in Klassen, um Ihren Code besser zu strukturieren und die Wiederverwendbarkeit von Code-Ausschnitten zu erhöhen.
- Geben Sie in einer README-Datei an, ob Sie Ihre Umsetzung mit Firefox oder Google Chrome getestet haben. Dokumentieren Sie in dieser README-Datei bitte auch weitere Besonderheiten, z.B. wenn Ihre Umsetzung unvollständig ist oder sich unter bestimmten Umständen fehlerhaft verhält.
- Ihr Abgabe-ZIP-Archiv muss alle zur Ausführung notwendigen Dateien beinhalten, nicht nur solche, die Sie neu erstellt haben.