Praktikum zu Web-Technologien/Web-Engineering, Wintersemester 2022/2023

Praktikum 9 (Objekte in JavaScript)

Im Rahmen des Praktikums entwickeln wir eine Web-Anwendung, welche wir Schritt für Schritt mit weiteren Anforderungen, Funktionen und Technologien erweitern.

Hinweis zu Bonuspunkten: Zur Vergabe der Bonuspunkte werden wir den von Ihnen produzierten Quellcode begutachten, sobald Sie bestimmte *Meilensteine* erreicht haben. Sobald ein Praktikum einen solchen Meilenstein markiert, finden Sie einen entsprechenden Hinweis auf dem jeweiligen Praktikumsblatt. Dieser Hinweis beinhaltet auch, wieviele Bonuspunkte für den jeweiligen Meilenstein erzielbar sind. Da die Praktika aufeinander aufbauen, enthält ein Meilenstein jeweils alle vorhergehenden Praktikumsaufgaben - zur Erlangung aller Bonuspunkte sind also alle Aufgaben zu bearbeiten. Um die Bonuspunkte zu erhalten, sprechen Sie uns einfach im Praktikum an, sobald Sie einen Meilenstein erreicht haben. Wir schauen uns dann gemeinsam Ihren Stand in einer kleinen Abnahme an. Dabei muss Ihre Gruppe, in welcher Sie die Aufgabe gelöst haben, vollständig anwesend sein. Insgesamt gibt es drei Meilensteine, und es sind maximal 10 Bonuspunkte über das Praktikum erreichbar. *Wichtig:* Abnahmen erfolgen nur bis maximal 20.01.23 (Ende der Vorlesungszeit)!

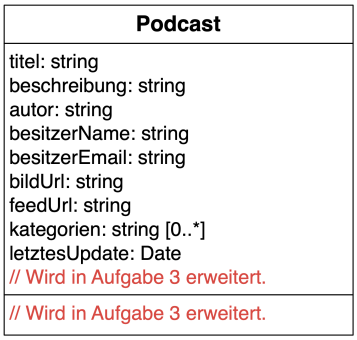
Hinweis zum entstehenden Code:

- Zur Verwaltung und zum kollaborativen Bearbeiten Ihres Quellcodes empfehlen wir Ihnen die Nutzung von Git (z.B. in Verbindung mit dem GitLab-Server des FB4, Login per FH-Account). Eine kurze Einführung in Git finden Sie hier.
- Falls Sie Git nicht verwenden, so legen Sie Ihren Quellcode pro Praktikumsaufgabe in separaten Verzeichnissen (z.B. „praktikum2“, „praktikum3“) ab, damit die einzelnen Entwicklungsschritte bei den Abnahmen ersichtlich sind.

Aufgabe 1: Objekt "Podcast"

Erweitern Sie das in Praktikum 8 erstellte externe JavaScript folgendermaßen:

1. Definieren Sie in dem Skript ein Objekt `Podcast`, welches alle Eigenschaften eines Podcasts enthält, wie in folgendem (unvollständigen) Klassendiagramm dargestellt:



Die Eigenschaften `feedUrl` und `bildUrl` enthalten die URLs zum RSS-Feed des Podcasts (über welchen der Podcast abonniert werden kann) sowie zum Bild (Logo) des Podcasts.

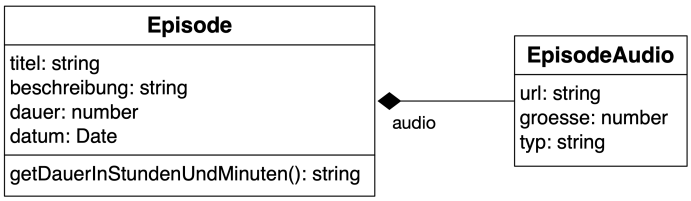
2. Definieren Sie das Objekt in einer Form, so dass mehrere Instanzen erstellt werden können.

3. Verwenden Sie für die Eigenschaft `kategorien` ein Array, welches die Kategorien des Podcasts (z.B. "Technology", "Web-Engineering") als Zeichenketten enthält.

4. Die Eigenschaft `letztesUpdate` soll jeweils Datum *und* Uhrzeit enthalten. Verwenden Sie zu diesem Zweck das `Date` -Objekt von JavaScript.

Aufgabe 2: Objekt "Episode"

1. Definieren Sie ein weiteres Objekt `Episode` , welches alle Eigenschaften einer Episode enthält, wie in folgendem (unvollständigen) Klassendiagramm dargestellt:



Die Eigenschaft `datum` soll analog zum `Podcast` -Objekt ein `Date` sein, welches Datum und Uhrzeit enthält.

2. Die Eigenschaft `dauer` gibt die Dauer der Episode in Millisekunden an. Realisieren Sie zusätzlich die Methode `getDauerInStundenUndMinuten` , welche die Dauer umrechnet und in einem String im Format `1h 20min` zurückgibt.

   Hinweis: Mit Hilfe der Standardfunktion `Math.floor(x)` können Sie Zahlen abrunden.

3. Die Informationen zur Audio-Datei einer Episode kapseln wir in einem separaten Objekt `EpisodeAudio` . Definieren Sie auch dieses Objekt, wie in obigem Klassendiagramm

dargestellt (Beziehung beachten). `EpisodeAudio` besitzt folgende Eigenschaften:

   - `url` : URL zur Audio-Datei
   - `groesse` : Größe der Audio-Datei in Bytes
   - `typ` : Mime-Typ der Audio-Datei (z.B. "audio/mpeg")

4. Definieren Sie die Objekte in einer Form, so dass mehrere Instanzen erstellt werden können.

Aufgabe 3: Episoden zu Podcasts hinzufügen

1. Erweitern Sie das Objekt `Podcast` um eine zusätzliche Eigenschaft `episoden` für die Episoden zu einem Podcast. Verwenden Sie für die Eigenschaft `episoden` ein Array, welches `Episode` -Objekte verwaltet. Beim Erstellen eines neuen `Podcast` -Objektes soll `episoden` zunächst als leeres Array initialisiert werden.

2. Realisieren Sie für das Objekt `Podcast` eine Methode `addEpisode(episode)` , die eine neue Episode zu einem Podcast hinzufügt.

3. Sorgen Sie dafür, dass das Array `episoden` stets sortiert ist, und zwar *absteigend nach der Eigenschaft* `datum` . Führen Sie die Sortierung direkt in der Methode `addEpisode` nach dem Hinzufügen der neuen Episode durch.

Hinweise:

- Verwenden Sie für 2. (`addEpisode`) und 3. (Sortieren) entsprechende Standardmethoden, die Ihnen das Array-Objekt von JavaScript anbietet.
- Für die Sortierung von `Date` -Objekten können Sie diese einfach in einem arithmetischen Ausdruck subtrahieren, z.B.:

```
// datum1 und datum2 sind Date-Objekte
datum2-datum1
```

Aufgabe 4: Objekte erzeugen und ausgeben

1. Erzeugen Sie mindestens zwei `Podcast` -Objekte als Beispieldatensätze. Sammeln Sie die erstellten `Podcast` -Objekte in einem Array. Fügen Sie zu jedem `Podcast` -Objekt mindestens zwei `Episode` -Objekte mit unterschiedlichen Daten hinzu. Verwenden Sie dazu die Methode `addEpisode` .

2. Nutzen Sie eine Schleife, um die `Podcast` -Objekte sowie die jeweils zugehörigen Episoden auf der Konsole auszugeben (`console.log([...])`). Die Ausgabe soll folgendem Format folgen (mit `$` beginnende Anteile sind Platzhalter, die durch

konkrete Daten zu ersetzen sind):

```
$podcast.titel:
    $episode1.titel ($episode1.dauerInStundenUndMinuten)
    $episode2.titel ($episode2.dauerInStundenUndMinuten)
    [...]
```

Beispielausgabe:

```
Working Draft:
    Revision 473: Vue 3, taugts? (1h 18min)
    Revision 472: GraphQL in 2021 (1h 6min)
Die Wochendämmerung:
    Bundesnotbremse, Intensivstationen, Schul-Bildung, Mietendeckel u
    Neue Physik, Biden erwägt TRIPS-Waiver, keine Transparenz und neu
```

Kontrollieren Sie anhand der Ausgabe, ob die Episoden korrekt sortiert sind (siehe Aufgabe 3 Punkt 3).

Allgemeine Hinweise:

- Validieren Sie Ihren HTML-Code mit dem W3C Markup Validator. Der entstehende HTML-Code soll beim Check keine Fehler und Warnungen produzieren.
- Ausnahme: Warnungen bzgl. `date` - und `time` -Eingabefeldern können Sie ignorieren.
- Validieren Sie Ihren CSS-Code mit dem W3C CSS Validation Service. Der entstehende CSS-Code soll beim Check keine Fehler und Warnungen produzieren.