Praktikum zu Web-Technologien/Web-Engineering, Wintersemester 2023/2024

Praktikum 8 (Objekte in JavaScript)

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

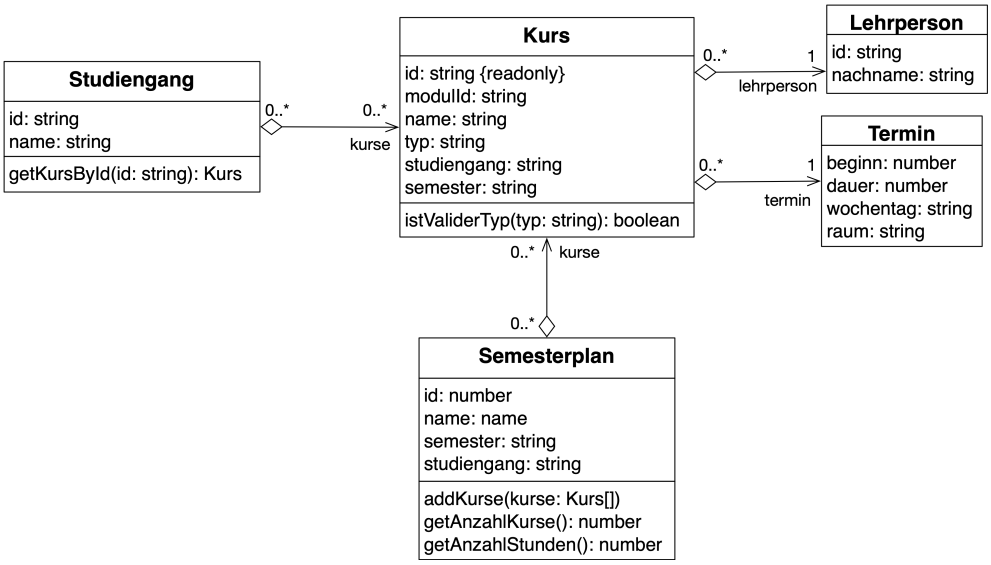
Hinweis zu Bonuspunkten: Zur Vergabe der Bonuspunkte werden wir den von Ihnen produzierten Quellcode begutachten, sobald Sie bestimmte *Meilensteine* erreicht haben. Sobald ein Praktikum einen solchen Meilenstein markiert, finden Sie einen entsprechenden Hinweis auf dem jeweiligen Praktikumsblatt. Dieser 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 18.01.24 (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: Fachobjekte

Wir definieren nun die Fachobjekte für die Daten unserer Anwendung. Erweitern Sie die JavaScript-Datei aus Praktikum 7 um folgende Fachobjekte:



Berücksichtigen Sie folgende Anforderungen:

1. Definieren Sie die Fachobjekte `Kurs`, `Lehrperson`, `Termin`, `Studiengang` und `Semesterplan` wie in obigem UML-Klassendiagramm dargestellt (Attribute, Methoden, Relationen und Richtung der Relationen beachten!). Implementieren Sie alle Objekte in einer Form, so dass jeweils mehrere Instanzen erstellt werden können.

2. Besonderheiten des Objektes `Kurs`:

    1. Der Wert der Eigenschaft `id` soll nicht als Konstruktorparameter übergeben, sondern automatisch erzeugt werden (daher die Markierung `{readonly}` im Diagramm). Konkatenieren Sie dazu die Werte der Eigenschaften `modulId`, `termin.wochentag`, `termin.beginn` und `termin.raum`.

    2. Die Funktion `istValiderTyp` soll überprüfen, ob der übergebene String `typ` einen gültigen Kurstyp darstellt. Ist dies der Fall, soll die Funktion `true` zurückgeben, ansonsten `false`. Folgende Typen sollen erlaubt sein:

Kurstyp	**Wert für `typ`**
Vorlesung	`V`
Übung	`Ü`
Praktikum	`P`
Übung/Praktikum	`ÜPP`
Seminaristische Vorlesung	`SV`
Tutorium	`T`
Seminar	`S`

Verwalten Sie die möglichen Werte als *Konstanten* in Ihrem Code.

3. Beim *Initialisieren* eines `Kurs`-Objektes soll überprüft werden, ob der gegebene Wert für `typ` gültig ist (s.o. Funktion `istValiderTyp`). Wurde kein valider Typ übergeben, so soll das `Kurs`-Objekt nicht erstellt und ein Fehler geworfen werden. Dabei hilft Ihnen folgender Codeausschnitt:

```
// „eineNachricht“ ist eine Fehlermeldung als Zeichenkette
throw new Error(eineNachricht);
```

3. Besonderheiten des Objektes `Studiengang`:

    1. Die Eigenschaft `id` beinhaltet ein Kürzel zur Identifikation des Studienganges (z.B. `WIPB` für "Wirtschaftsinf. BPO 2018", `STDBSW` für "Software- und Systemtechnik VR Softwaretechnik BPO 2015").

    2. Verwenden Sie für die Eigenschaft `kurse` ein Array, welches `Kurs`-Objekte verwaltet.

    3. Die Funktion `getKursById` liefert das `Kurs`-Objekt zur gegebenen ID (Eigenschaft `id`) oder `undefined`, falls kein solcher Kurs zum Studiengang gefunden wird.

4. Besonderheiten des Objektes `Semesterplan`:

    1. Der Wert der Eigenschaft `id` soll eindeutig sein und automatisch vergeben werden. Realisieren Sie diesen Wert als fortlaufende Nummer.

    2. Verwenden Sie für die Eigenschaft `kurse` ein Array, welches `Kurs`-Objekte verwaltet. Beim Erstellen eines neuen `Semesterplan`-Objektes soll `kurse` zunächst als leeres Array initialisiert werden.

    3. Die Funktion `addKurse` soll alle `Kurs`-Objekte im übergebenen Array zum Semesterplan hinzufügen.

    4. Die Funktion `getAnzahlKurse` gibt zurück, wie viele Kurse dem Semesterplan zugeordnet sind.

    5. Die Funktion `getAnzahlStunden` gibt die Gesamtdauer aller im Semesterplan enthaltenen Kurse zurück.

Aufgabe 2: Objekte erzeugen, sortieren und ausgeben

1. Erzeugen Sie mindestens drei vollständige `Kurs`-Objekte als Beispieldatensätze. Erzeugen Sie zudem je mindestens ein vollständiges `Studiengang`-Objekt sowie ein vollständiges `Semesterplan`-Objekt. Ordnen Sie dem `Studiengang`- sowie dem `Semesterplan`-Objekt jeweils die drei `Kurs`-Objekte zu.

2. Sorgen Sie dafür, dass die `Kurs`-Objekte sortiert sind, und zwar *aufsteigend nach der Eigenschaft* `modulId`. Verwenden Sie für die Implementierung entsprechende Standardmethoden, die Ihnen das Array-Objekt von JavaScript anbietet (keine eigene Sortierung implementieren!).

3. Nutzen Sie Schleifen und Template-Literale, um *alle* erstellten Beispieldatensätze auf der Konsole auszugeben (`console.log([...])`). Die Ausgabe soll folgendem Format folgen (mit `$` beginnende Anteile sind Platzhalter, die durch konkrete Daten zu ersetzen sind):

   ```
   $studiengang.name ($studiengang.id)
       $kurs1.modulId: $kurs1.name
       $kurs2.modulId: $kurs2.name
       [...]
   $semesterplan.name ($semesterplan.semester)
       $kurs1.modulId: $kurs1.name
       $kurs2.modulId: $kurs2.name
       [...]
   ```

   Beispielausgabe:

   ```
   Wirtschaftsinf. BPO 2018 (WIPB):
       46812: Datenbanken 2
       46834: Künstliche Intelligenz
       46990: BWL-Anwendungen
   Mein Plan (SS 22):
       46812: Datenbanken 2
       46834: Künstliche Intelligenz
       46990: BWL-Anwendungen
   ```

   Kontrollieren Sie anhand der Ausgabe, ob die Kurse korrekt sortiert sind (siehe Punkt 2).

Allgemeine Hinweise:

- Validieren Sie Ihren HTML-Code mit dem W3C Markup Validator. Der entstehende HTML-Code soll beim Check keine Fehler und Warnungen produzieren.
- Validieren Sie Ihren CSS-Code mit dem W3C CSS Validation Service. Der entstehende CSS-Code soll beim Check keine Fehler und Warnungen produzieren.