## Einleitung

Ihre Aufgabe ist die Entwicklung eines Teiles eines Backends für einen elektronischen Kalender.

## Technische Rahmenbedingungen

- Verwenden Sie zur Entwicklung des Programms TypeScript >= 3.x und Node.js >= 10.x.
- Verwenden Sie das *Express.js*-Framework zur Umsetzung der Web API.
- Sie können davon ausgehen, dass Strings mit der JavaScript-Methode `parse` in ein `Date`-Objekt umgewandelt werden können (ist erst seit ES5 empfehlenswert).
- [Optional: +1 Zusatzpunkt] Persistieren Sie die Termine mit Hilfe von **Loki.js**.

## Anforderungen

### Mindestanforderungen

- Ihr Programm muss die im Unterricht besprochenen Empfehlungen (*Best Practices*) für Node.js-Projekte einhalten (z.B. Konfigurationsdateien mit entsprechendem Inhalt, Git *ignore* file, etc.).
- Schreiben Sie leicht lesbaren, ordentlich formatierten Code mit sinnvollen Namen für Variablen, Funktionen und Typen.
- Gestalten und Entwickeln Sie eine Web API-Methode zum **Einfügen neuer Termine** (Englisch: *Appointments*) unter folgenden Rahmenbedingungen:
  - Ein Termin **muss** einen Start- und einen Endzeitpunkt haben (jeweils Datum und Zeit).
  - Ein Termin **kann** einen optionalen Ort haben.
  - Ein Termin **muss** eine Beschreibung haben.
  - Wird versucht, einen ungültigen Termin einzufügen (Pflichtfeld fehlt, Start- oder Endzeitpunkt sind ungültig, Endzeitpunkt ist vor Startzeitpunkt), geben Sie einen *Bad Request*-Fehler zurück.
  - Wenn der Termin erfolgreich angelegt werden konnte, geben Sie *Created* zurück.
- Gestalten und Entwickeln Sie eine Web API-Methode zum **Abfragen von Terminen** unter folgenden Rahmenbedingungen:
  - Die Methode muss **optionales Filtern** auf Jahr und/oder Monat ermöglichen (bezogen auf den Startzeitpunkt).

  - Bei Übergabe eines fehlerhaften Filters (Jahr oder Monat sind ungültig), geben Sie einen *Bad Request*-Fehler zurück.

- **Überlappende Termine sind nicht möglich**. Falls versucht wird, einen Termin einzufügen, der sich mit einem bestehenden Termin überschneidet, geben Sie einen *Conflict*-Fehler zurück.
- Vergeben Sie **für jeden Termin eine eindeutige Nummer** (*id*). Die Web API zum Anlegen von Terminen muss den kompletten Termin inkl. generierter ID zurückgeben.
- Gestalten und Entwickeln Sie eine Web API-Methode zum **Löschen eines Termins** unter folgenden Rahmenbedingungen:
  - Zum Identifizieren des Termins wird die ID verwendet.
  - Bei Übergabe einer fehlerhaften ID (keine Zahl), geben Sie einen *Bad Request*-Fehler zurück.

### Testrequests

```
# Valid new appointment
POST http://localhost:8080/api/calendar
Content-Type: application/json

{
    "start": "2019-01-02T09:00:00",
    "end": "2019-01-02T11:00:00",
    "loc": "Office",
    "desc": "Project Meeting"
}

###
# Valid new appointment
POST http://localhost:8080/api/calendar
Content-Type: application/json

{
    "start": "2019-02-05T14:00:00",
    "end": "2019-02-05T16:00:00",
    "loc": "School",
    "desc": "Choir"
}

###
# Valid new appointment
POST http://localhost:8080/api/calendar
```

```
Content-Type: application/json

{
    "start": "2018-03-27T08:00:00",
    "end": "2018-03-27T18:00:00",
    "loc": "School",
    "desc": "Basketball Training"
}

###
# Invalid appointment (missing mandatory members)
POST http://localhost:8080/api/calendar
Content-Type: application/json

{
    "end": "2019-01-05T16:00:00"
}

###
# Invalid appointment (invalid dates)
POST http://localhost:8080/api/calendar
Content-Type: application/json

{
    "start": "dummy",
    "end": "2019-02-31T16:00:00",
    "loc": "School",
    "desc": "Choir"
}

###
# Invalid new appointment (overlapping)
POST http://localhost:8080/api/calendar
Content-Type: application/json

{
    "start": "2019-01-02T10:00:00",
    "end": "2019-01-02T12:00:00",
    "loc": "Office",
    "desc": "Project Meeting"
}

###
```

```
# Invalid new appointment (end before start)
POST http://localhost:8080/api/calendar
Content-Type: application/json

{
    "start": "2019-01-02T12:00:00",
    "end": "2019-01-02T10:00:00",
    "loc": "Office",
    "desc": "Project Meeting"
}

###
# Valid, unfiltered list of appointments
GET http://localhost:8080/api/calendar

###
# List of appointments filtered by year
GET http://localhost:8080/api/calendar?year=2019

###
# List of appointments filtered by year and month
GET http://localhost:8080/api/calendar?year=2019&month=1

###
# Invalid filter
GET http://localhost:8080/api/calendar?year=2019&month=13

###
# Invalid filter
GET http://localhost:8080/api/calendar?year=dummy

###
# Valid delete request
DELETE http://localhost:8080/api/calendar/1

###
# Valid delete request for ID that does not exist
DELETE http://localhost:8080/api/calendar/99

###
# Invalid delete request
DELETE http://localhost:8080/api/calendar/dummy
```