

# RESTful Interfaces und Templates mit Flask

## Einleitung

Im Folgenden soll eine kleine Flask-Anwendung gebaut werden die ein RESTful Interface implementiert. Das Thema dreht sich um den Stundenplan der FH Hof, den wir mit einem Python-Skript von der Website geparkt haben, und als JSON im Ordner `data/` bereitstellen.

Wir haben bereits ein Modul zum Laden dieser Daten bereitgestellt (in `loader/load.py`). Ihr müsst lediglich `app.py` für diese Übung anpassen, ein Skelett für eine Flask-Anwendung liegt dort bereits vor.

Ausführen könnt ihr eure Anwendung indem ihr in das Directory `practice` navigiert, und dort `app.py` ausführt. Danach könnt ihr einem Webbrowser (zB Chromium) eurer Wahl,

Testen könnt ihr die Aufgaben **a)** und **b)** indem ihr `test.py` ausführt:

```
$ python2 test.py
** Querying: http://localhost:5000/api/count/all
** Querying: http://localhost:5000/api/count/Inf
** Querying: http://localhost:5000/api/list
** Querying: http://localhost:5000/api
```

```
-----
Ran 4 tests in 0.016s
```

```
OK
```

## Aufgaben

**a)**

Erweitere `app.py` so dass eine Abfrage von `localhost:5000/api/Inf/5` unseren aktuellen Stundenplan liefert. Der Studiengang `Inf` und das Semester soll dabei entsprechend durch andere Studiengänge wie `BW` ersetzt werden.

*Beispiel Antwort:*

```
{
  "Dienstag": [
    {
      "desc": "\"Erstellung von Applikationen f\u00fcr ein Windows Phone\" 11.03.14.03.2013",
      "name": "Praxisblock I",
      "prof": "Prof. Dr. Martin Thost",
      "room": "FB006",
      "time": "09:00-17:30",
      "type": "HF:1"
    }
  ],
  ...
}
```

Für das Abfragen eines Python Dictionaries mit dem Stundenplan könnt ihr die Funktion `load(studiengang, semester)` nutzen. Für das Umwandeln in valides JSON könnt ihr die eingebaute Funktion `json.dumps(python_object)` nutzen.

Wird ein Stundenplan von einem nichtexistierenden Kurs geholt, so soll ein leeres Dictionary zurückgegeben werden. (`{}`). Für nichtexistierende Kurse löst die `load`-Funktion eine `NoSuchCourse` Exception aus.

**b)**

Die API soll um ein paar Funktionen erweitert werden:

- `list_courses` soll eine sortierte Liste aller Studiengänge wiedergeben.

*Beispiel:*

```
$ curl localhost:5000/api/list_courses  
["BBB", "BW", "GP", "IM", "Inf", ..., "Wing MT", "Wing WT"]
```

Zum Abfragen der unsortierten Python Liste könnt ihr die Funktion `list_courses` verwenden.

- `count` soll die Anzahl der aktuell studierenden Semester für einen Studiengang ausgeben. Für Inf wäre das beispielsweise 3 (Inf1, Inf3, Inf5).

*Beispiel:*

```
$ curl localhost:5000/api/count/Inf  
3  
$ curl localhost:5000/api/count/all  
84
```

Zum Abfragen der Anzahl könnt ihr die bereitgestellte Funktion `count(studiengang)` nutzen.

c)

Erweitere die Flask-Anwendung um eine weitere URL `localhost:5000/view/<studiengang>/<semester>`.

Diese soll bei Aufruf von beispielsweise `view/Inf/5` unseren Stundenplan rendern.

Nutze dazu die Methode `render_template()` und schreibe ein Jinja2 Template dass fähig ist den Stundenplan als HTML zu rendern.

Inn `templates/hello.html` ist eine leichte Hilfestellung bzgl. HTML Tables.

*Beispiel:*

Dienstag					
room	note	time	prof	type	name
FB010	(Inf+MI5)	08:00-09:30	Prof. Dr. Peter Stöhr	FWM:1	ObjectiveC und iOSProgrammierung
FB102	(Inf+MI+Wi5)	11:30-13:00	Prof. Dr. Jürgen Heym	FWM:1	RoutingSwitchingTroubleshooting 1
FB114	(Inf+MI+Wi5)	11:30-13:00	Prof. Dr. Horst Heineck	FWM:1	Oracle 11g Release 2 Administration Workshop I
FA017	(Inf5 + Wi5)	14:00-15:30	Prof. Dr. Philipp Schaible	FWM:1	Wiederverwendungsbasierte Entwicklung von Systemen
FA013	Ulrich Lang	AWM:1	Meteorologie nicht nur für Privatpiloten	17:30-19:00	
FB023	Grundlagen der Medizin und der Medizintechnik für Informatiker und Ingenieure	19:15-20:45	Prof. Dr. med. Johannes Bodky	AWM:1	Teil Grundlagen der Medizin und Medizintechnik Beginn KW 41