LU03b - RESTful Flask

Flask ist ein leichtgewichtiges Web-Framework für Python, das sich besonders gut für die Entwicklung von Webanwendungen und RESTful APIs eignet. Flask-RESTful ist eine Erweiterung des Flask-Frameworks für die einfache Implementierung von RESTful Webservices in Python.

Unterlagen zu Flask im Modul 323

Flask-RESTful erleichtert die Erstellung von REST-APIs durch die Bereitstellung von Tools und Strukturen, um Ressourcen, Routen und andere Funktionen eines RESTful Webservices zu organisieren. Flask-RESTful bietet eine einfache und effektive Möglichkeit, RESTful Webservices in Python zu entwickeln. Es ermöglicht die Strukturierung von Code und bietet Funktionen, um häufige Aufgaben bei der Erstellung von RESTful APIs zu erleichtern.

Grundlegende Konzepte und Funktionen

Installation

Flask-RESTful kann mit dem Python-Paketmanager pip installiert werden:

```
pip3 install flask-restful
```

Struktur des Projekts

Eine Flask-Applikation benötigt ein zentrales Python-Skript, welches die Defintion der Services enthält. Dieses Skript wird häufig `app.py` genannt.

Weitere Vorgaben zur Struktur des macht Flask nicht. Es empfiehlt sich jedoch ein etabliertes Architekturmuster für seine Applikation zu verwenden. Für meine Projekte verwende ich die MVC-Architektur und das DAO-Pattern.

Beispiel

Erstellen eines einfachen RESTful Webservices

Ein einfacher Flask-RESTful Webservice kann durch die Erstellung einer Flask-App und der Integration von Flask-RESTful erreicht werden.

app.py

```
from flask import Flask
from flask_restful import Resource, Api

def create_app():
    app = Flask(__name__)
    CORS(app)
    api = Api(app)

    api.add_resource(HelloWorld, '/hello')
    return app

if __name__ == '__main__':
    app = create_app()
    app.run(debug=True)
```

In der Funktion `create_app` wird die Flask-Applikation erstellt. Dieses Beispiel definiert eine Ressource (`HelloWorld`) und den Pfad (`/hello`) zu dieser Ressource.

Definition von Ressourcen

Ressourcen werden durch Klassen repräsentiert, die von der Resource-Klasse von Flask-RESTful erben. Jede Ressource implementiert HTTP-Methoden (GET, POST, PUT, DELETE) als Methoden der Klasse (z.B., get, post, put, delete).

helloworld.py

```
from flask import make_response
from flask_restful import Resource

class HelloWorld(Resource):
    def get(self):
        data = {'hello': 'world'}
        return make_response(
            data, 200
        )
```

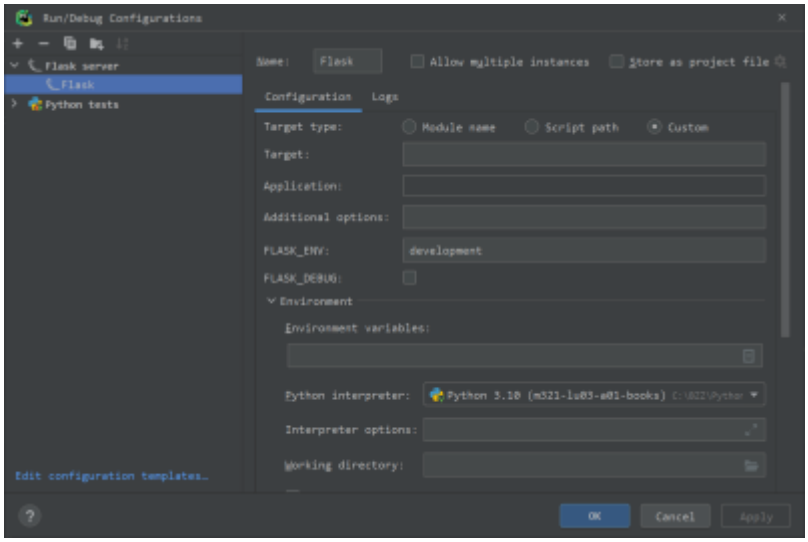
In diesem Beispiel würde eine GET-Anfrage an die URL `/hello` die get-Methode der HelloWorld-Ressource aufrufen. Diese Methode liefert ein JSON-Objekt und den HTTP-Status 200 als Antwort.

Flask-RESTful mit PyCharm

Um die Applikation auszuführen, musst du eine Runtime Configuration hinzufügen:

1. Menu „Run“ → „Edit Configurations …“
2. Füge eine neue Konfiguration hinzu mit dem „+“-Symbol.

3. Wähle „Flask Server“



Die Einstellungen sollten automatisch korrekt sein. Du kannst das Fenster mit [OK] schliessen.

M321-LU03