First Steps with FastAPI

Reindert-Jan Ekker

@rjekker   www.codesensei.nl

Overview

Create a FastAPI app
– Explore
– Run

Check result in browser
– REST Api
– Auto-generated documentation

Common Problems

FastAPI code execution
– Async

Demo

- Start a PyCharm project

- Install FastAPI

- Create a FastAPI app

- Run

- Inspect the result

Control flow is dictated by incoming HTTP Requests

An operation is only called if its URL is requested.

```
python -m pip install "fastapi[all]"
```

Install FastAPI

Extra dependencies are installed using [all]
This includes the uvicorn server
Make sure to install in an active virtual environment

```python
from fastapi import FastAPI

app = FastAPI()
```

Creating a FastAPI Application Object

The app object represents our REST service

```python
@app.get("/date")

def date():

    """Return the current date/time."""

    return {'date': datetime.now()}
```

Add an Operation

URLs should start with a slash
There is no connection between URL and function name
When a GET request comes in for /date, the function will be called

```python
from fastapi import FastAPI

app = FastAPI()
```

Creating a FastAPI Application Object

The app object represents our REST service

```python
@app.get("/date")

def date():

    """Return the current date/time."""

    return {'date': datetime.now()}
```

Add an Operation

URLs should start with a slash
There is no connection between URL and function name
When a GET request comes in for /date, the function will be called

```
uvicorn my_project:app --reload
```

Running FastAPI Project

uvicorn is the HTTP server
my_project is the name of the python file containing the application object
app is the name of the application object inside that file

Resources

FastAPI documentation: https://fastapi.tiangolo.com/

Demo code: github.com/codesensei-courses/fastapi-fundamentals

Summary

Install FastAPI

Create and run a FastAPI app

Autogenerated documentation

Common Problems

FastAPI code execution

Up Next:
Serving Data with FastAPI