# Lab 02. Get Data

In this lab, we will implement a web service to get data from server.

Objectives:

- Understand Paths and Path Parameters
- Understand Query Strings
- Learn how to use Documentation

Tags:

- path, path parameters, query strings, documentation

## A. Create a New Project

1. Create a folder `lab02`, which will be the project folder for Lab 02.
2. Create a new python file "main.py" inside the `lab02` folder.
3. Add following code to the file.

```
from fastapi import FastAPI

app = FastAPI()

@app.get("/")
def get_root():
    return {'message': 'Lab 02'}
```

4. In the terminal, goes into the `lab02` folder, and run following command.

```
uvicorn main:app --reload
```

5. Above command serves your app in local machine at `http://127.0.0.1:8000`. Go to that URL in the browser.

## B. Path with Parameters

We would like to create a set of APIs which allow users to work with cities and their Abbreviations.

1. Add following code to `main.py` to add a route for path `/weekdays` with `GET` operation.
    - It uses a dictionary to store data in memory. (This is not recommended in Production.)

```
weekdays = {0:'Sun', 1:'Mon', 2:'Tue'}

@app.get('/weekdays')
def get_weekdays():
    return {'data': weekdays}
```

2. Check out the route `/weekdays` in web browser.

3. Add following code to `main.py` to add a route for path `/weekdays/{weekday_id}` with `GET` operation, where `weekday_id` is the path parameter.
    - The value of the path parameter `weekday_id` will be passed to the function as an argument.
    - By default, the argument is a string. We make use of `int()` function to convert it to integer.
    - It echoes the `weekday_id` as returned value.

```
@app.get('/weekday/{weekday_id}')
def get_weekday(weekday_id):
    weekday_id = int(weekday_id)
    return {'weekday_id': weekday_id}
```

**Exercise**

4. Modify `get_weekday()` function to return weekday name corresponding to the `weekday_id`.

```
@app.get('/weekday/{weekday_id}')
def get_weekday(weekday_id):
    weekday_id = int(weekday_id)
    return {'weekday_id': weekdays[weekday_id]}
```

## Type Conversion & Validation

5. We can use Python standard type annotations to specify the data type of argument.
    - With type annotation, FastAPI will perform data conversion automatically.
    - Test it on web browser http://localhost:8000/weekday/1

```
@app.get('/weekday/{weekday_id}')
def get_weekday(weekday_id: int):
    return {'weekday': weekdays[weekday_id]}
```

6. FastAPI also provides data validation based on type annotation.
    - Test it on web browser http://localhost:8000/weekday/abc
    - If we pass in non-numeric data in path parameter, it will return a HTTP error. It returns following error message.

```
{"detail":[{"loc":["path","weekday_id"],"msg":"value is not a valid integer","type":"type_error.integer"}]}
```

**Exercise**

7. Create another route `/find_weekday/{year}/{month}/{day}`, which returns weekday of the date specified by `year`, `month` and `day`.
    - You can create a datetime object using `datetime(year=year, month=month, day=day)`.
    - You can get the weekday number of a datetime object using its `weekday()` method. Monday has an integer value of 0, and Sunday has an integer value of 6.
    - Test your code by visiting web browser http://127.0.0.1:8000/find_weekday/2019/2/4

```
from datetime import datetime

@app.get('/find_weekday/{year}/{month}/{day}')
def find_weekday(year: int, month: int, day: int):
    dt = datetime(year=year, month=month, day=day)
    wd = dt.weekday()
    return {'weekday': weekdays[wd]}
```

## C. Status Code

HTTP status code gives different meaning of the request result. By default, FastAPI returns default parameter `status_code=200`.

To use a different status code, return a `JSONResponse` object with custom content and set the `status_code` accordingly:

1. Enhance the `GET /weekday/{weekday_id}` route to return `404` error when `weekday_id` is not found in the dictionary.

```
@app.get('/weekday/{weekday_id}')
def get_weekday(weekday_id: int):
  try:
    return {'weekday': weekdays[weekday_id]}
  except:
    content = {'message': f'Invalid weekday ID: {weekday_id}'}
    return JSONResponse(status_code=status.HTTP_404_NOT_FOUND,
content=content)
```

## D. Query Strings

Query strings are key-value pairs added to the back of base URL. It is used as a way to send optional data to server.

In FastAPI, function parameters that are not part of the path parameters, they are automatically interpreted as "query" parameters.

1. Enhance the `GET /weekdays` route to accept a query string `case`, which can be either `upper` or `lower`. If `case` query string is present, API will convert result to `upper` or `lower` case accordingly.

```
from typing import Optional
from enum import Enum
class CaseEnum(Enum):
  upper = 'upper'
  lower = 'lower'

@app.get('/weekdays')
def get_weekdays(case: Optional[CaseEnum] = None):
  if case == CaseEnum.upper:
    return { k:v.upper() for k,v in weekdays.items()}
  elif case == CaseEnum.lower:
    return { k:v.lower() for k,v in weekdays.items()}
```

```
    else:
      return {'weekdays': weekdays}
```

## E. Documentation

FastAPI automatically generates interactive API documentation for your project.

- Visit path `/docs` to view the documentation. http://127.0.0.1:8000/docs
- You can test API directly in the documentation.