Advanced Programming with Python
Session 2

Pepe García jgarciah@faculty.ie.edu

Plan for today

- Flask routing

Plan for today

- Flask routing
- HTTP clients with Requests

- Flask routing
- HTTP clients with Requests
- returning different status codes

Plan for today

- Flask routing
- HTTP clients with Requests
- returning different status codes
- using request bodies

HTTP routes

We can also capture part of the path as a variable:

```python
@app.route("/hello/<name>")
def hello(name):
    return "hello " + name
```

HTTP methods

One can specify which methods the function handles in the **methods** parameter

```python
@app.route("/hello", methods=["GET"])
def hello():
    return "hi!"


@app.route("/goodbye", methods=["POST"])
def goodbye():
    return "bye!"
```

Returning JSON

Flask has a **jsonify** function that we can use to convert the data we want to JSON:

```python
from flask import Flask, jsonify

app = Flask("hello server")

@app.route("/hello")
def hello():
    return jsonify({"message": "hello", "name": "Pepe"})
```

HTTP

Exercise

Accept the assignment and clone it **https://bit.ly/3saKAMq**
Let's do the `tweeter_json.py` exercise from the homework repository.

HTTP clients

So far, we've been focusing only on one side of the client-server side, the server.

However, we can create HTTP clients in Python too!

HTTP clients. requests library

We can use requests to get an HTTP response as follows:

```python
import requests

response = requests.get("url")

data = response.json()

print(data)
```

Status codes

We all know the infamous **404 Not Found** HTTP status code. Apart of it, there are a lot more that are used when developing HTTP servers. Some of the most used are:

Status codes

200 OK

Used whenever everything went correctly.

Status Codes

201 Created

Used to give the user feedback so they know the resource has been created.

Status Codes

400 Bad request

A general error in the received request. It's used commonly too mark a request as invalid because of some validation problem.

Status Codes

404 Not found

Whenever the resource requested by the user is not found

Interlude... Tuples

Interlude… Tuples

We all remember the list data structure, a structure that can hold zero or more elements of different types.

Well, there's another list-like data structure, called `tuple`. The big difference between the two of them is that lists can grow or shrink in size, with the `.pop` or `.append` methods, while tuples cannot change their size.

Interlude... Tuples

```python
my_list = [1,2,3]  # We create lists with square brackets
my_tuple = (1,2,3) # We create tuples with parentheses

my_list.append(4) # adds an element at the end of my_list
my_tuple.append(4) # ERROR! tuple object has no attribute append
```

Interlude… Tuples

Something else to remark about tuples is that, if Python sees comma separated values without any surounding (parentheses, curly brackets, or square brackets), will understand them as a tuple.

```python
tuple_with_parentheses = (1,2,3)
tuple_without_parentheses = 1,2,3


print(type(tuple_with_parentheses))
# <class 'tuple'>
print(type(tuple_without_parentheses))
# <class 'tuple'>
```

Status Codes

Flask allows returning a **tuple** in any route, in which the first parameter is the **response body**, and the second the **status code**:

```python
@app.route("/users/<user_id>")
def get_user(user_id):
    if user_not_found():
        return jsonify({"error": "not found"}), 404
```

Status Codes

See `exercises/translations.py`

Practice

Let's implement a simple flask server that finds the correct translation for `hello` in a dictionary of translations.
We want our server to respond to requests to **/translation/<language>**.
The dictionary can look like this:

```
translations = {
  "en": "hello",
  "es": "hola",
  "it": "ciao",
}
```

If the received language doesn't exist, we want to return a 404 response.