URL Routes and Views

In this lesson, we will get to know what views are, and how they are associated with the URL routes. We will also learn how to create static routes.

WE'LL COVER THE FOLLOWING

Introduction

The homepage of a website can usually be found at the URL `hostname` followed by `/`, `/home`, `/index` or something self-explanatory.

These kinds of URLs enable the users to remember the URL and access it easily. It would be unexpected if the homepage were at a random URL such as `/39283@&3911` or `/more_eggs`.

Flask allows us to use the `route()` decorator to bind a *meaningful* URL to each **view function** we create.

What is a view function?

In the discussion of the **MTV** architecture, we studied what a *view* is. In Flask, we create a function that acts as the *view*. Recall from the previous lesson; we created a function called `hello`. This function acted as a *view*. Then, we bonded it with a `route` bonded it with a `route`.

```
@app.route("/")
def hello():
    return "Hello World!";
```

> **Note:** In developer circles, this function and the `route` combined is commonly referred to as the **view** or just simply the **route**. Also, in this course, we will be using these terms interchangeably.

The `route()` decorator

The route decorator takes the following parameters:

- `rule`: The **rule** represents the URL rule which is passed as a string to the decorator.
- `endpoint` *(not needed)*: The **endpoint** is the name of the **view function** which is bound to the URL route. Flask assumes this parameter itself, and the developer does not need to specify it.
- `options` *(optional)*: The **options** are an optional parameter. We will discuss it in more detail later.

> **FYI:** If you are unfamiliar with the concept of decorators, please refer to the Python docs at PEP318.

Static routing

In static routing, we specify a constant URL string as a rule to the `route()` decorator. For example in the mini-application given below, we have specified two static routes having URLs `/` and `/educative` respectively.

> **Note:** The `rule` is a case-sensitive string. Therefore, `/educative` and `/Educative` are totally different **URLs**.

An example using static URL routes

```
from flask import Flask, render_template
app = Flask(__name__)

@app.route("/")
def home():
    return "Welcome to the HomePage!"

@app.route("/educative")
def learn():
    return "Happy Learning at Educative!"


if __name__ == "__main__":
    app.run(debug=True, host="0.0.0.0", port=3000)
```

Try this: select the URL given below the **Run** button to view the webpage in a separate tab. Append `/educative` to the URL and observe the output!

Explanation

`home()` view

This view function corresponds to the route `"/"`. When you open the URL or check the **"Output"** tab, this view will be called. The `"/"` is always the default route of any web application.

For example, when we go to educative.io the `host` = "educative.io" and `route` = "/".

`learn()` view

This view function corresponds to the route `"/educative"`. When you open the URL and append `"/educative"`, this view will be called.

For example, when we go to educative.io/explore, the `host` = "educative.io" and `route` = "/explore".

In the next lesson, we will learn how to create dynamic routing! Stay tuned.