Carlos J. Costa

Flask
web development,
one drop at a time

Flask

- Flask is a microframework for Python
- based on Werkzeug, Jinja 2 and good intentions.
- BSD licensed

Flask Application (ex01)

```python
from flask import Flask
app = Flask(__name__)
@app.route('/')
def hello_world():
    return 'Hello, World!'
if __name__ == "__main__":
    app.run()
```

Flask Application

```
#import Flask class
from flask import Flask
#create object app
app = Flask(__name__)
# use the route() decorator to tell Flask what URL should
# trigger our function
@app.route('/')
#The function is given a name which is also used to
# generate URL
def hello_world():
    return 'Hello, World!'
# The global variable, __name__, in the module that is the entry point to your program, is '__main__'.
# Otherwise, it's the name you import the module by. So, code under the if block will only run if the
# module is the entry point to your program. It allows the code in the module to be importable by other
# modules, without executing the code block  beneath on import.
if __name__ == "__main__":
    app.run()
```
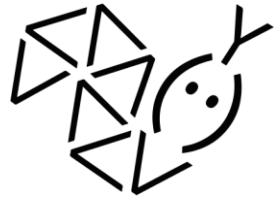
Flask Application

- Save in a folder flask_app.py

- Call:

    python flask_app.py

- In the browser:

    localhost:5000

Alternatives

Rounting

- Use the route() decorator to bind a function to a URL.

```
@app.route('/')
def index():
    return 'Index Page'


@app.route('/hello')
def hello():
    return 'Hello, World'
```

Rounting (ex02)

```python
from flask import Flask
app = Flask(__name__)
@app.route('/')
def hello_world():
    return 'Hello, World (in index paage)'
@app.route('/hello')
def hello():
    return 'Hello, World (in hello page)'
if __name__ == "__main__":
    app.run()
```

Static Files

- Dynamic web applications also need static files.

- E.g.CSS and JavaScript

- Create a folder called in the package or next to the module

    /static

    url_for('static', filename='style.css')

Rendering Templates

- Flask configures the Jinja2 template engine automatically

- To render a template use the render_template() method

- Example:

  /flask_app.py

  /templates

  /hello.html

Rendering Templates
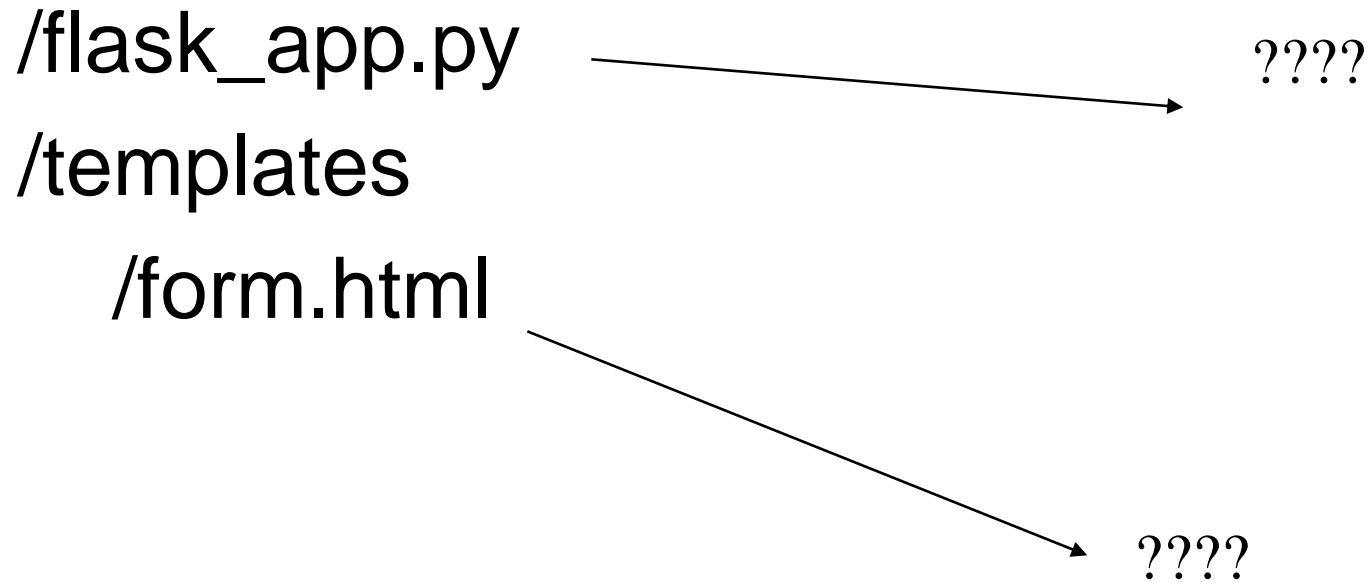
/flask_app.py

/templates

   /hello.html

```python
from flask import render_template

@app.route('/hello/')
@app.route('/hello/<name>')
def hello(name=None):
    return render_template('hello.html', name=name)
```

```html
<!doctype html>
<title>Hello from Flask</title>
{% if name %}
  <h1>Hello {{ name }}!</h1>
{% else %}
  <h1>Hello, World!</h1>
{% endif %}
```

Rendering Templates (ex03)

/flask_app.py   ————————————————→   ????

/templates

    /form.html ————————————————→   ????

Rendering Templates (ex03)

flask_app.py
File

```python
from flask import Flask, render_template
app = Flask(__name__)
@app.route('/')
def render_static():
    return render_template('form.html')
@app.route('/hello')
def hello():
    return 'Hello, World (in hello page)'
if __name__ == "__main__":
    app.run()
```
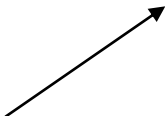
Rendering Templates (ex03)

?

```
<html>
  <body>
    <form action = "/save/" method = "POST">
      <p>author <input type = "text" name = "Author" /></p>
      <p>phrase <input type = "text" name = "Phrase" /></p>
      <p><input type = "submit" value = "submit" /></p>
    </form>
  </body>
</html>
```

HTTP Methods

- Web applications use different HTTP methods when accessing URLs.

- By default, a route only answers to GET requests.

- use the methods argument of the route() decorator to handle different HTTP methods.

HTTP Methods

```python
from flask import request


@app.route('/login', methods=['GET', 'POST'])
def login():
    if request.method == 'POST':
        return do_the_login()
    else:
        return show_the_login_form()
```

Rendering Templates (ex04)

```python
from flask import Flask, render_template, request
app = Flask(__name__)
@app.route('/')
def render_static():
    return render_template('form.html')
@app.route('/save/', methods=['GET', 'POST'])
def hello():
    return 'Hello!!!'
if __name__ == "__main__":
    app.run()
```

New
flask_app.py
File

Save Data (ex05)

```python
from flask import Flask, render_template, request
app = Flask(__name__)
@app.route('/')
def render_static():
        return render_template('form.html')
@app.route('/save/', methods=['GET', 'POST'])
def index():
        data = request.form['Author']+" - "+request.form['Phrase']
        fo= open("test1.txt", "a+")
        fo.write(data+"\n")
        fo.close()
        return "thank you"
if __name__ == '__main__':
        app.run(debug = True)
```

flask_app.py
File

Tiny App (ex06)

/flask_app.py

/templates

   /form.html

   /index.html

Tiny App (ex06)

```python
from flask import Flask, request, render_template
app = Flask(__name__)
@app.route('/save/', methods=['GET', 'POST'])
def write():
        data = request.form['Author']+" - "+request.form['Phrase']
        fo= open("test1.txt", "a+")
        fo.write(data+"\n")
        fo.close()
        return render_template('index.html')
@app.route('/read/')
def read():
        fo= open("test1.txt", "r")
        data1=fo.read()
        fo.close()
        return data1
@app.route('/')
def render_static():
    return render_template('form.html')
if __name__ == '__main__':
    app.run(debug = True)
```

flask_app.py
File

Tiny App (ex06)

<div style="border: 2px solid green; display: inline-block; padding: 10px;">
form.html
File
</div>

```
<html>
  <body>
    <form action = "/save/" method = "POST">
      <p>author <input type = "text" name = "Author" /></p>
      <p>phrase <input type = "text" name = "Phrase" /></p>
      <p><input type = "submit" value = "submit" /></p>
    </form>
  </body>
</html>
```

Tiny App (ex06)

```
<html>
  <body>
    menu
    <p><a href="/read/">list data</a></p>
    <a href="../">form</a>
  </body>
</html>
```

- https://github.com/masterfloss/python-web