GKTCS INNOVATIONS
Revolution in Learning and Development

Flask

By
Surendra Panpaliya

GKTCS INNOVATIONS
Revolution in Learning and Development

- Mr. Surendra R Panpaliya is Founder, CEO and an International Coporate Trainer at GKTCS Innovations, Pune.
- He has total 19 + years of experience in Corporate Training, Software Development and Academia.
- He has provided training to 100 + MNC and Institutes across globe. Some of the popular organisations are: HCL, Singapore, ISRO, Ahmadabad, E-Oman Government, Muscat, Oman, Saudi Electrical Company, Riyadh, Saudi, Accenture,
- Taught 10000 + Academic & Industrial Trainees.
- He has completed his Master Degree in Computer Application with first class in the year 2000 .

GKTCS INNOVATIONS
Revolution in Learning and Development

What is Web Framework?

Web Application Framework or simply Web Framework represents a collection of libraries and modules that enables a web application developer to write applications without having to bother about low-level details such as protocols, thread management etc.

GKTCS INNOVATIONS
Revolution in Learning and Development

What is Flask?

Flask is a web application framework written in Python.

It is developed by Armin Ronacher, who leads an international group of Python enthusiasts named Pocco.

Flask is based on the Werkzeug WSGI toolkit and Jinja2 template engine. Both are Pocco projects.

GKTCS INNOVATIONS
Revolution in Learning and Development

What is WSGI?

WSGI Web Server Gateway Interface (WSGI) has been adopted as a standard for Python web application development.

WSGI is a specification for a universal interface between the web server and the web applications.

GKTCS INNOVATIONS
Revolution in Learning and Development

What is Werkzeug?

It is a WSGI toolkit, which implements requests, response objects, and other utility functions.

This enables building a web framework on top of it.

The Flask framework uses Werkzeug as one of its bases.

GKTCS INNOVATIONS
Revolution in Learning and Development

What is Jinga2?

Jinga2 is a popular templating engine for Python.

A web templating system combines a template with a certain data source to render dynamic web pages.

GKTCS INNOVATIONS
Revolution in Learning and Development

What is Jinga2?

Flask is often referred to as a micro framework.

It aims to keep the core of an application simple yet extensible.

Flask does not have built-in abstraction layer for database handling, nor does it have form a validation support.

Instead, Flask supports the extensions to add such functionality to the application.

GKTCS INNOVATIONS
Revolution in Learning and Development

FLASK -Virtual ENVIRONMENT

Installation - Stable Use pip to install Flask in a virtualenv. pip install flask

Step by step instructions for creating a virtualenv for your project:

1. mkdir flaskex && cd flaskex
2. python3 -m venv flaskproj1
or `virtualenv env` for Python 2
3. source flaskproj1/bin/activate
4. pip install flask

Or

```
$ pip install Flask==0.10.1
```

FLASK -Virtual ENVIRONMENT

```
$ git init
```

```
$ pyvenv-3.5 env
$ source env/bin/activate
```

```
$ touch app.py .gitignore README.md requirements.txt
```

This will give you the following structure:

```
├── .gitignore
├── app.py
├── README.md
└── requirements.txt
```

```
$ pip freeze > requirements.txt
```

GKTCS INNOVATIONS
Revolution in Learning and Development

FLASK -Virtual ENVIRONMENT

- **Virtualenv** - https://realpython.com/python-virtual-environments-a-primer/
- **Flask** - http://flask.pocoo.org/
- **git/Github** - https://realpython.com/python-git-github-intro/
- **Heroku (basics)** - https://devcenter.heroku.com/articles/getting-started-with-python

FLASK -Virtual ENVIRONMENT

```
Surendras-MacBook-Pro:flaskproj1 surendra$ pwd
/Users/surendra/flaskex/flaskproj1
Surendras-MacBook-Pro:flaskproj1 surendra$ source ./bin/activate
(flaskproj1) Surendras-MacBook-Pro:flaskproj1 surendra$ ls
bin      include     lib      pyvenv.cfg
(flaskproj1) Surendras-MacBook-Pro:flaskproj1 surendra$ python
Python 3.7.0 (default, Jun 28 2018, 07:39:16)
[Clang 4.0.1 (tags/RELEASE_401/final)] :: Anaconda, Inc. on darwin
Type "help", "copyright", "credits" or "license" for more
information.
>>> import flask
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>

ModuleNotFoundError: No module named 'flask'
>>>
```

FLASK - Installation

```
(flaskproj1) Surendras-MacBook-Pro:flaskproj1 surendra$ pip install flask
Collecting flask
  Downloading https://files.pythonhosted.org/packages/9b/
93/628509b8d5dc749656a9641f4caf13540e2cdec85276964ff8f43bbb1d3b/Flask-1.1.1-py2.py3-none-any.whl (94kB)
    100% |████████████████████████████████| 102kB 330kB/s
Collecting Jinja2>=2.10.1 (from flask)
  Downloading https://files.pythonhosted.org/packages/1d/e7/
fd8b501e7a6dfe492a433deb7b9d833d39ca74916fa8bc63dd1a4947a671/Jinja2-2.10.1-py2.py3-none-any.whl (124kB)
    100% |████████████████████████████████| 133kB 348kB/s
Collecting click>=5.1 (from flask)
  Downloading https://files.pythonhosted.org/packages/fa/
37/45185cb5abbc30d7257104c434fe0b07e5a195a6847506c074527aa599ec/Click-7.0-py2.py3-none-any.whl (81kB)
    100% |████████████████████████████████| 81kB 6.1MB/s
Collecting Werkzeug>=0.15 (from flask)
  Downloading https://files.pythonhosted.org/packages/d1/ab/
d3bed6b92042622d24decc7aadc8877badf18aeca1571045840ad4956d3f/Werkzeug-0.15.5-py2.py3-none-any.whl (328kB)
    100% |████████████████████████████████| 337kB 648kB/s
Collecting itsdangerous>=0.24 (from flask)
  Downloading https://files.pythonhosted.org/packages/76/ae/
44b03b253d6fade317f32c24d100b3b35c2239807046a4c953c7b89fa49e/itsdangerous-1.1.0-py2.py3-none-any.whl
Collecting MarkupSafe>=0.23 (from Jinja2>=2.10.1->flask)
  Downloading https://files.pythonhosted.org/packages/ce/c6/
f000f1af136ef74e4a95e33785921c73595c5390403f102e9b231b065b7a/MarkupSafe-1.1.1-cp37-cp37m-
macosx_10_6_intel.whl
Installing collected packages: MarkupSafe, Jinja2, click, Werkzeug, itsdangerous, flask
Successfully installed Jinja2-2.10.1 MarkupSafe-1.1.1 Werkzeug-0.15.5 click-7.0 flask-1.1.1
itsdangerous-1.1.0
(flaskproj1) Surendras-MacBook-Pro:flaskproj1 surendra$
```

FLASK - Example

```
#Hello World

#Create hello.py:

from flask import Flask

app = Flask(__name__)

@app.route('/')

def hello():

    return 'Hello, World!'
```

GKTCS INNOVATIONS
Revolution in Learning and Development

FLASK - Example

Initialization

All Flask applications must create an application instance.

The web server passes all requests it receives from clients to this object for handling, using a protocol called Web Server Gateway Interface (WSGI).

The application instance is an object of class Flask, usually created as follows:

```
from flask import Flask

app = Flask(__name__)
```

GKTCS INNOVATIONS
Revolution in Learning and Development

FLASK - Example

Initialization

The only required argument to the Flask class constructor is the name of the main module or package of the application. For most applications,

Python's __name__ variable is the correct value.

GKTCS INNOVATIONS
Revolution in Learning and Development

FLASK - Example

Routes and View Functions

Clients such as web browsers send requests to the web server, which in turn sends them to the Flask application instance.

The application instance needs to know what code needs to run for each URL requested, so it keeps a mapping of URLs to Python functions.

The association between a URL and the function that handles it is called a **route.**

GKTCS INNOVATIONS
Revolution in Learning and Development

FLASK - Example

The most convenient way to define a route in a Flask application is through the app.route decorator exposed by the application instance, which registers the decorated function as a route.

The following example shows how a route is declared using this decorator:

```
@app.route('/')
def index():
    return ‘Hello World!’
```

GKTCS INNOVATIONS
Revolution in Learning and Development

FLASK - Example

```
@app.route('/')
def index():
    return 'Hello World!'
```

Decorators are a standard feature of the Python language; they can modify the behavior of a function in different ways.

A common pattern is to use decorators to register functions as handlers for an event.

The return value of this function, called the response, is what the client receives. If the client is a web browser, the response is the document that is displayed to the user

GKTCS INNOVATIONS
Revolution in Learning and Development

FLASK - Example

Functions like index() are called view functions.

A response returned by a view function can be a simple string with HTML content, but it can also take more complex forms.

GKTCS INNOVATIONS
Revolution in Learning and Development

FLASK - Example

```
@app.route('/user/<name>')

def user(name):
 return '<h1>Hello, %s!</h1>' % name
```

The portion enclosed in angle brackets is the dynamic part, so any URLs that match the static portions will be mapped to this route.

When the view function is invoked, Flask sends the dynamic component as an argument.

GKTCS INNOVATIONS
Revolution in Learning and Development

FLASK - Example

The dynamic components in routes are strings by default but can also be defined with a type.

For example, route /user/
would match only URLs that have an integer in the id dynamic segment.

Flask supports types int, float, and path for routes.

The path type also represents a string but does not consider slashes as separators and instead considers them part of the dynamic component.

GKTCS INNOVATIONS
Revolution in Learning and Development

FLASK - Example

Server Startup The application instance has a run method that launches Flask's integrated development web server:

```
if __name__ == '__main__':
    app.run(debug=True)
```

The __name__ == '__main__'

Python idiom is used here to ensure that the development web server is started only when the script is executed directly.
When the script is imported by another script, it is assumed that the parent script will launch a different server, so the app.run() call is

GKTCS INNOVATIONS
Revolution in Learning and Development

FLASK - Example

Once the server starts up, it goes into a loop that waits for requests and services them.
This loop continues until the application is stopped, for example by hitting Ctrl-C.
There are several option arguments that can be given to app.run() to configure the mode of operation of the web server.

During development, it is convenient to enable debug mode, which among other things activates the debugger and the reloader.

This is done by passing the argument **debug set to True.**

GKTCS INNOVATIONS
Revolution in Learning and Development

FLASK - Example

Then run it with:

```
export FLASK_APP=hello.py

flask run

 * Running on http://localhost:5000/
```

Adding the code below will allow running it directly with python hello.py.

```
if __name__ == '__main__':

 app.run()
```

GKTCS INNOVATIONS
Revolution in Learning and Development

FLASK - Example

```
1  #Hello World
2  #Create hello.py:
3
4  from flask import Flask
5  app = Flask(__name__)
6  @app.route('/')
7  def hello():
8      return 'Hello, World!'
9
10 if __name__ == '__main__':
11     app.run()
```

```
 * Serving Flask app "__main__" (lazy loading)
 * Environment: production
   WARNING: Do not use the development server in a production environment.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [21/Aug/2019 06:01:11] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [21/Aug/2019 06:01:12] "GET /favicon.ico HTTP/1.1" 404 -
```

FLASK - Example

```
Surendras-MacBook-Pro:~ surendra$ python
Flask_Hello.py
 * Serving Flask app "Flask_Hello" (lazy loading)
 * Environment: production
   WARNING: Do not use the development server in a
production environment.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on http://127.0.0.1:5000/ (Press CTRL+C
to quit)
127.0.0.1 - - [21/Aug/2019 06:37:39] "GET / HTTP/
1.1" 200 -
127.0.0.1 - - [21/Aug/2019 06:37:40] "GET /
favicon.ico HTTP/1.1" 404 -
```

FLASK - Example

Home | MySQL_Python_Example | 127.0.0.1:5000

Hello, World!

FLASK - Example

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


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


@app.route('/<name>')
def hello_name(name):
    return "Hello {}!".format(name)

if __name__ == '__main__':
    app.run()
```

FLASK - Example

Run your app locally to make sure everything works - `python app.py`

Test it out by adding a name after the URL - i.e.,

http://localhost:5000/surendra

GKTCS INNOVATIONS
Revolution in Learning and Development

FLASK - Example

Installation - Development If you want to develop and contribute to the Flask project, clone the repository and install the code in development mode.

git clone ssh://github.com/pallets/flask

cd flask

python3 -m venv env

source env/bin/activate

pip install -e

GKTCS INNOVATIONS
Revolution in Learning and Development

sphinx

Used to build the documentation.

pip install sphinx

cd docs

make html

firefox _build/html/index.html

py.test

Used to run the test suite.

pip install pytest

GKTCS INNOVATIONS
Revolution in Learning and Development

GKTCS INNOVATIONS

Revolution in Learning and Development

E-Learning

Corporate Training

Software Development

Consultancy

www.gktcs.com

support@gktcs.com

+91 9975072320

+91 7620379390