GANATAN

Flask avec angular

Guide Complet

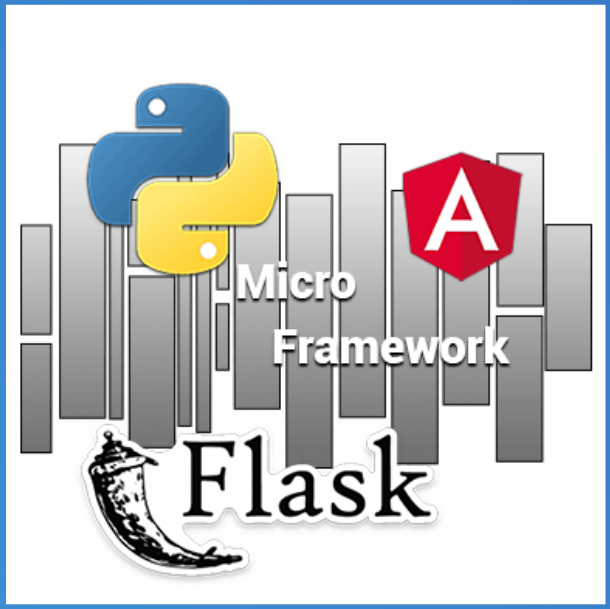
Mise à jour du 3 janvier 2025

Flask est un **micro-framework** créé en 2003.

Avec **Django** il est l'un des deux **Frameworks Python** les plus utilisés.

Dans ce guide complet je vais vous montrer comment créer une **API** avec Flask.

Nous verrons de quelle façon nous pourrons l'utiliser avec le **Framework Angular**.



www.ganatan.com

Ce que nous allons faire ?

- **Qu'est-ce que Flask?**
  C'est un micro-framework Python, qu'est-ce que cela veut dire ?
- **Installation de Flask et**
  Comment installer et utiliser Flask ?
- **Première Application**
  Comment créer une application avec Flask en quelques minutes ?
- **Création des API**
  Comment créer une API et l'utiliser avec Angular ?
- **Code source**
  Le code complet du projet **angular-app-multi-api** sur github.

Qu'est ce que Flask ?

Si vous lisez ces lignes c'est que vous essayez d'être développeur.

C'est pas toujours facile je vous l'accorde, alors nous aurons besoin d'aide et des amis nous seraient bien utiles.

Notre meilleur ami s'appellera **Framework**.

Un Framework est un ensemble d'outils, de librairies qui nous facilitera la tâche.

Il existe deux types de Frameworks

- **Les full-stack frameworks**
- **Les micro-frameworks**

Détaillons un peu ces deux cas.

- **full-stack frameworks**

  En tant que full-stack, ils tentent de nous fournir presque tout ce qui est nécessaire pour créer une application.

  Cela va du service Web à la gestion de base de données jusqu'à la génération de pages HTML.

  Ils contiendront des éléments essentiels comme

  - Un moteur de template

  - L'internationalisation

  - Un ORM (object-relational mapping) ce qui fera le lien avec la base de données.

- **micro-frameworks**

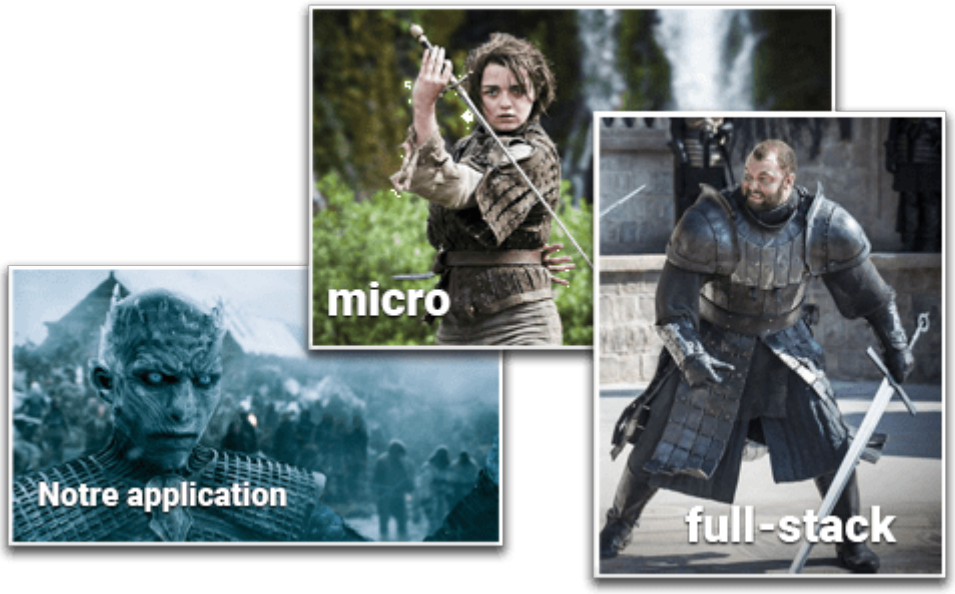
  Ils sont utilisés pour créer de petites applications.

  On pourrait remplacer le mot micro par miniature.

  Dans le cas d'un micro-framework on aura le minimum de fonctionnalités (la base en fait).

Plutôt facile à visualiser me diriez-vous non ?

Arya Stark vs La montagne



Mais pas toujours facile de choisir ?

Si vous vouliez vous occuper du roi de la nuit et de ses marcheurs blancs quel ami choisiriez-vous ?

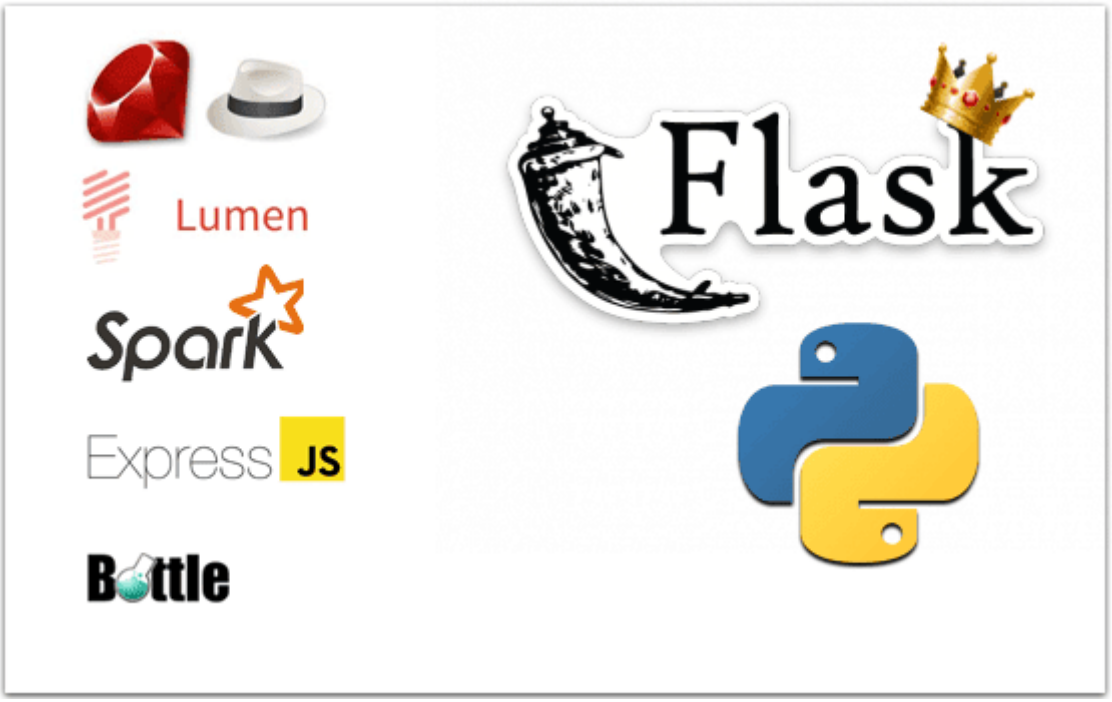
Flask est un micro-framework open-source en Python

Pour simplifier on pourrait dire que Flask est un Framework miniature.

Ci-dessous quelques micro-frameworks parmi les plus connus

- **Sinatra** pour **Ruby**
- **Lumen** pour **PHP**
- **Spark** pour **Java**
- **Express.js** pour **Node.js**
- **Bottle** pour **Python**
- **Flask** pour **Python**

Flask un grand parmi les petits



Les origines

Flask a été créé initialement par le développeur Autrichien **Armin Ronacher** le **1er avril 2010.**

Commencé comme un poisson d'avril Flask est devenu le **Framework Python le plus populaire sur Github**.

Le créateur de Flask

Le projet fût d'abord appelé **Denied**.

Flask est basé sur deux modules

- **werkzeug**

  Une bibliothèque Python de type **WSGI.**

  Web **S**erver **G**ateway **I**nterface étant une spécification qui définit une interface entre des serveurs et des applications web.

  https://werkzeug.palletsprojects.com/en/1.0.x/

- **jinja2**

  Un moteur de template (ou moteur de modèle).

  https://jinja.palletsprojects.com/en/2.11.x/

Deux modules sur lesquels avait travaillé Armin Ronacher.

Depuis Flask est distribué sous **licence BSD** (*Berkeley Software Distribution*).
Il s'agit d'une licence libre et non restrictive.

Voyons voir quelle est sa renommée !

Simple faisons un petit tour sur **Github**

- Star **61300**, Fork **15300**, Commits **4860**

https://github.com/pallets/flask

Le site officiel

- Flask

https://flask.palletsprojects.com/en/1.1.x/

Les avantages de Flask

Bien qu'il soit destiné à créer de petites applications flask offre des avantages indéniables.

- Il dispose d'un **serveur web intégré**.
  Donc nul besoin d'installer Apache ou nginx, Flask suffira amplement.
- Il est **facile** à prendre en main.
- Il est **simple** et **léger**, et donc **rapide** à implémenter.
- Il est **compatible WSGI** grâce à l'utilisation de Werkzeug.
  Toutes les applications ainsi développées seront compatibles entre elles.
- Il permet de travailler sur les **objets connectés**.
- Il permet de créer des **API REST** (**R**epresentational **s**tate **t**ransfer)

Ronacher l'a concu pour pouvoir construire rapidement une application web en utilisant **un seul fichier Python.**

C'est évidemment ce point qui nous intéressera plus particulièrement.

En effet **Flask** fera office de **Backend** là où **Angular** jouera le rôle de **Frontend.**

Alors c'est parti pour nos expérimentations.
A nous de vérifier si c'est simple à faire.

Installation de Flask

Pour connaitre les détails qui m'ont permis de décrire l'installation,

c'est ici sur le site officiel

https://flask.palletsprojects.com/en/1.1.x/

Il nous faut bien sûr quelques prérequis

- Installer l'interpréteur **Python**.
- Installer le gestionnaire de dépendances **pip**.

Pour plus de détails revoyez le tutoriel dédié à Python

https://www.ganatan.com/tutorials/python-avec-angular

Flask est tout simplement un package (dépendance ou librairie) destinée à Python.

Utilisons donc pip pour l'installer.

Je vous propose aussi d'utiliser les commandes linux classiques (apt-get)

```
# Installation de deux manières

# Installation avec les paquets
apt-get install python-flask

# Installation via le gestionnaire de dépendances Pypi
# Installation d'une version spécifique
pip3 install flask==3.0.8

# Installation de la dernière version disponible
pip3 install flask

# Vérification de l'installation
flask --version
```

```
# Et voilà ce que l'on obtient
Python 3.8.3
Flask 1.1.2
Werzeug 1.0.1
```

Les différentes versions de Flask peuvent être gérées en utilisant **virtualenv**.

Cela nous permettra de gérer les changements de version de Python.

Les **environnements virtuels** vont permettre de travailler avec différentes version de flask

Ils ne sont possibles qu'avec Python 3.4

Première application Flask

Ce tutoriel fait partie d'une série de tutoriels permettant au Frontend Angular d'utiliser différents Backend.

Pour cette raison j'utilise à chaque fois des ports différents qui se succèdent.

Par exemple

- Backend avec **Python**

  port 5201
- Backend avec **Django**

  port 5202
- Backend avec **Flask**

  port 5203

C'est la raison pour laquelle nous utiliserons le port **5203** dans les exemples qui suivent.

Libre à vous de le changer à votre guise.

Flask est très intéressant pour sa simplicité.

Voyons voir.

Nous allons créer un serveur HTTP avec du routing en quelques lignes dans un fichier écrit en python.

http-server-simple.py

```
from  flask import Flask

app = Flask(__name__)


@app.route('/')
def home():
    message = "Flask HTTP Server \n"    return message


@app.route('/movies')
def movies():
    message = "Movies List \n"    return message


port = 5203if __name__ == '__main__':
    app.run(host='0.0.0.0', port=port)
```

```
# Exécution de l'application
py http-server-simple.py

# Tester la page home
http://localhost:5202

# Tester la page movies
http://localhost:5202/movies
```

HTTP Serveur avec pages HTML

Faisons un peu plus compliqué en rajoutant des pages HTML.

http-server-html.py

```
from  flask import Flask

app = Flask(__name__)


@app.route('/')
def home():
    return '''
<html>
    <head>
        <title>Flask Example Starter</title>
    </head>
    <body>
        <h1>Home Page</h1>
        <h2>Flask HTTP server</h2>
    </body>
</html>'''



@app.route('/movies')
def list():
    return '''
<html>
    <head>
        <title>Flask Example Starter</title>
    </head>
    <body>
        <h1>Movies List</h1>
        <h2>Gladiator</h2>
        <h2>Alien</h2>
    </body>
</html>'''


if __name__ == '__main__':
    app.run(host='0.0.0.0', port=5203)
```

```
# Exécution de l'application
py http-server-html.py

# Tester la page home
http://localhost:5202

# Tester la page movies
http://localhost:5202/movies
```

Pour aller plus loin

Pour faire le tour de Python

Je vous propose trois tutoriels pour créer des API

- https://www.ganatan.com/tutorials/python-avec-angular
- https://www.ganatan.com/tutorials/django-avec-angular
- https://www.ganatan.com/tutorials/flask-avec-angular