flask

December 24, 2014

1 5 Applications Web et python

Classiquement sur internet on utilise un serveur ***lamp*** :
linux (os), **a**pache (serveur), **m**ysql (base de données), **p**hp (language de programmation pour avoir des pages dynamiques).

Il existe de nombreux outils pour le web écrit en python: *serveur Web* (Zope, gunicorn) ; *script cgi*; *frameworks Web* (Flask, Django, cherrypy etc . . . ,

nous utiliserons le framework ***flask***. Pour se documenter je vous conseille deux sites.
- http://flask.pocoo.org/docs - http://openclassrooms.com/courses/creez-vos-applications-web-avec-flask

1.1 5.1 Une page web avec flask

- Télécharger le fichier https://github.com/debimax/cours-debimax/raw/master/documents/isn-flask.tar.gz.
- Décompressez le (en console : tar zxvf isn-flask.tar.gz).

1.1.1 5.1.1 Flask et les templates

a) Le principe Avec *Flask* on pourrait mettre le code dans un seul *fichier.py* mais on utilise de préférence des *templates* (avec *jinja2*).

L'arborescence d'un projet Flask sera :

```
projet/script.py
projet/static/
projet/templates
projet/templates/les_templates.html
```

On met tous les templates dans le dossier ***templates/***. Le dossier ***static/*** contiendra lui toutes les images, fichiers css, js . . .

Ouvrez avec *geany* le fichier ***exemple.py*** :

```
1 #!/usr/bin/python3
2 # -*- coding: utf-8 -*-
3 from flask import Flask, render_template, url_for
4 app = Flask(__name__)   # Initialise l'application Flask
5
6 @app.route('/')
7 def accueil():
8     lignes=['ligne {}'.format(i) for i in range(1,10)]
9     return render_template("accueil.html", titre="Bienvenue !",lignes=lignes)
10
11 if __name__ == __main__ :
12     app.run(debug=True)
```

Dans le code ci-dessous, quel est le contenu de la variable ***lignes***?

```
liges=['ligne{}'.format(i) for i in range(1,10)]
```

.....

Ouvrez avec *geany* maintenant le fichier ***templates/accueil.html***.

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <meta charset="utf-8" />
5 <title>{{ titre }}</title>
6 </head>
7 <body>
8 <h1>{{ titre }}</h1>
9 <ul>
10 {% for ligne in lignes %}
11 <li>{{ ligne }}</li>
12 {% endfor %}
13 </ul>
14 </body>
15 </html>
```

Ouvrez une console dans ***/isn-flask/*** puis tapez *** python3.4 exemple.py***
Ouvrez alors un navigateur internet à l'adresse `http://localhost:5000` (celui qui n'est pas écrit pablo).

b Ajouter la date Il faudra importer la librairie ***time*** puis utiliser le décorateur ***@app.context_processor*** pour passer l'heure à tous les templates.

```
3 from flask import Flask, render_template, url_for
4 import time
5 app = Flask(__name__) # Initialise l'application Flask
6
7 @app.context_processor
8 def passer_date_heure():
9     date=time.strftime('%d/%m/%Y',time.localtime())
10     heure=time.strftime('%H',time.localtime())
11     return dict(date=date,heure=heure)
12
13 @app.route('/')
```

Il faut maintenant modifier le template ***accueil.html***

```
13     </ul>
14    <footer>Nous somme le {{ date }} et il est {{ heure }} heures.</footer>
15     </body>
```

Question: Pourquoi est il peu judicieux d'utiliser le serveur python pour afficher l'heure?

.....

.....

c Les fichiers css Nous allons utiliser un fichiers *css* pour l'habillage (présentation) de notre page web.
Modifions le fichier ***templates/accueil.html***.

```
4 <title> titre </title>
6 <link href="{{ url_for('static',filename='mon_style.css')}}" rel="stylesheet" type="text/css" />
7 </head>
```

```
15    <footer>Nous somme le <em class="Rouge">{{ date }}</em> et il est <em class="Rouge">{{ heure }}</e
```

Créer ensuite le fichier ***static/mon_style.css*** avec ceci:

```
1 h1 {
2   color: #ff00ff;
3   text-align: center;
4 }
5
6 footer {
7   position: absolute;
8   bottom: 0;
9   width: 100%;
10  height: 20px;
11  background-color: #f5f5f5;
12  text-align: center;
13 }
14
15 em.Rouge {color: #ff0000;}
```

Je met le contenu de la balise
<h1>
au centre et en couleur.
Je place le 'footer' au centre en bas de la page avec un fond de couleur.
Je met en rouge la classe Rouge.

d Le javascript Nous allons maintenant utiliser un fichier ***jascript (.js)*** pour afficher l'heure. Le code javascript sera effectué par le navigateur. Ce n'est pas le serveur qui exécute le code donc pas besoin de rafraichir la page internet pour que le code soit exécuté.
Il n'est pas question d'expliquer le code javascript mais vous pouvez regarder à quoi cela ressemble dans le ficher ***static/mes_scripts.js***.

Comme pour le fichier *.css* Il faut indiquer le fichier *.js* aux templates.

- Modifions le fichier ***templates/accueil.html***.

```
 6 <link href=" url_for('static', filename='mon_style.css') " rel="stylesheet" type="text/css" />
 7 <script type=text/javascript src="{{url_for('static', filename='mes_scripts.js') }}"></script>
 8 </head>

25 </div>
26 <footer>Nous sommes le <em class=Rouge id="date"></em> <script type="text/javascript"> window.or
27</body>
```

- Modifions le fichier ***templates/accueil.html***.

```
 6 <script type=text/javascript src="url_for('static', filename='mes_scripts.js') "></script>
 7 <link rel="shortcut icon" href="{{ url_for('static', filename='favicon.ico') }}">
 8 </head>

11 <p>Bonjour <b> prenom
12 <footer>Nous sommes le <em class=Rouge id="date"></em> <script type="text/javascript">window.onl
13 </body>
```

d Les images Ajoutons un favicon. Vous savez c'est la petite images à gauche de l'adresse dans un navigateur. Le nom de l'image doit être ***favicon.ico***.

J'ai déjà mis deux images dans le dossier ***static/***.
Il suffit donc de rajouter dans ***templates/accueil.html***

```
 7 <link rel="shortcut icon" href="{{ url_for('static' , filename='favicon.ico')}}">
 8 </head>
 9 <body>
10 <h1>{{ titre }} <img src="{{ url_for('static',filename ='fleur.png') }}" alt="fleur" title="fleur" bo
11 <ul>
```

1.1.2 5.1.2 Les formulaires

Modifions la page ***templates/accueil.html*** pour mettre un *formulaire*

```
15  </ul>
16  <div id="content">
17     <form method="post" action="{{ url_for('hello') }}">
18     <label for="nom">Entrez votre nom:</label>
19     <input type="text" name="nom" /><br />
20     <label for="prenom">Entrez votre prénom:</label>
21     <input type="text" name="prenom" /><br />
22     <input type="submit" />
23     </form>
24 </div>
```

pour le fichier ***exemple.py*** on crée une nouvelle route.

```
13  @app.route('/')
14  def accueil():
15      lignes=['ligne {}'.format(i) for i in range(1,10)]
16      return render_template("accueil.html", titre="Bienvenue !",lignes=lignes, date=date, heure=heure
17
18 @app.route('/hello/', methods=['POST'])
19 def hello():
20     nom=request.form[ nom ]
21     prenom=request.form[ prenom ]
22     return render_template( page2.html ,titre="Page 2", nom=nom, prenom=prenom,date=date,heure=heure
```

1.1.3 5.1.3 Une partie commune à toutes les pages

Nous avons plusieurs templates qui utilisent le même bas de page.
Il est donc possible d'utiliser un seul fichier template et de l'inclure dans les autres. Créer un fichier *templates/footer.html*

```
<footer>Nous sommes le <em class=Rouge id="date"></em> <script type="text/javascript">window.onload = da
```

Puis modifier les templates ***templates/accueil.html***

```
25 </div>
26 {% include 'footer.html' %}
27 </body>
```

et ***templates/page2.html***.

```
12 <p>Bonjour <b>{{ prenom }} {{ nom }}.</b></p>
13 {% include 'footer.html' %}
14 </body>
```

1.1.4 5.1.4 Mettre le site sur internet

Habituellement on s'héberge soit même mais il est possible de mettre son site chez un hébergeur. Il en existe plusieurs qui autorise les scripts python.

Ouvrez le navigateur pablo à l'adresse https://isn-flask.herokuapp.com/
Oui vous avez reconnu notre tp. J'ai simplement affiché la *date* et l'*heure* avec du code du javascript.

1.1.5 5.1.5 Prolongement

- Utilisation d'une base de donnée
- Créer une image aux couleurs aléatoire (module pil) et l'afficher.

1.2 5.2 Exercices

1. - Créer un dossier ***static/file/*** et mettre dans ce dossier quelques fichiers textes (.txt) et images (.png ou .jpg).

   - Créer dans le fichier ***exemple.py*** une fonction ***listdir()*** qui retourne la liste les noms des fichiers contenus dans le dossier ***static/file/*** *(On utilisera la librairie* ***os*** *pour lister).*
   - modifier le ***templates/accueil.html*** pour faire afficher le nom des fichiers.

2. Modifier alors la fonction ***listdir()*** pour n'afficher que le nom des images.
3. Afficher cette fois les images dans la page internet.