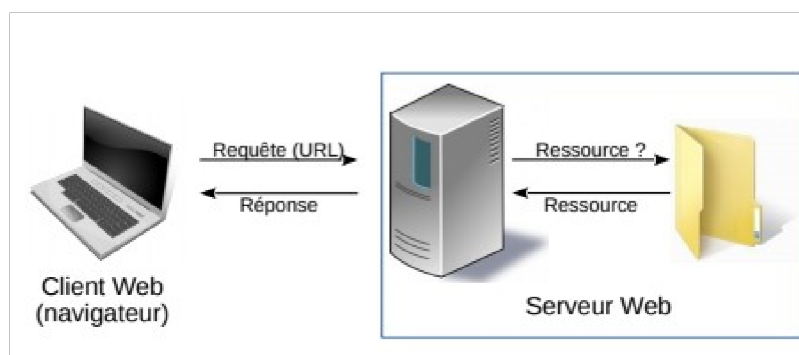


## ACT Créer un serveur web en python

Un serveur web est une application spécifique qui tourne sur un ordinateur. Il existe des serveurs écrit en langage C, en Java, en Python,...

Le rôle du serveur Web est de servir des clients. Ces derniers envoient au serveur une requête précisant leur demande. Le serveur répond à cette demande si celle-ci est valide.

Pour communiquer le serveur et le client utilise un langage commun, il s'agit du protocole **HTTP**.



Les échanges entre le client et le serveur se font sous forme de **paquets** qui transportent sur le réseau Internet les requêtes du client et les réponses du serveur.

Les sources sont disponibles sur <https://replit.com/@lyceestcharles/httpserver#main.py>

Ce langage est assez compréhensible pour nous, voici par exemple la requête d'un client demandant la page **index.html** sur le serveur **ollier.yo.fr** :

```
GET /index.html HTTP/1.0
Host: ollier.yo.fr
```

Cette requête correspondrait à la saisie par un utilisateur de l'**URL** : <https://ollier.yo.fr/>

Ici **ollier.yo.fr** correspond au nom de la machine, on dit aussi **nom de domaine** de la machine. Pour trouver leur chemin sur le réseau Internet ces paquets ont besoin de connaître l'**adresse IP** du serveur. Pour trouver cette adresse IP à partir du nom de domaine le client interroge un **serveur DNS**. Ici, notre serveur **ollier.yo.fr** aurait par exemple pour adresse IP **146.88.237.17**

**Note** : Pour connaître l'adresse IP d'un serveur, on peut passer par un service de DNS lookup (ou annuaire inversé), mais plus simplement, en terminal :

```
$ host nom_de_mon_serveur
```

## ETAPE 1 : Un premier serveur

Une façon simple et légère de créer un serveur web avec python.

- 1) Créer une page web html simple et l'enregistrer sous **index.html** comme par exemple :

```
<!DOCTYPE html>
<html>
  <head>
    <title>python friendly</title>
  </head>
  <body>
    <h1>C'est mon serveur !</h1>
    <p></p>
    <p>Super, mon serveur HTTP fonctionne...</p>
  </body>
</html>
```

- 2) Nous allons maintenant créer un fichier python et l'enregistrer sous **serveur.py**. (Attention, tous les fichiers créés doivent se trouver dans le même répertoire)

```
#!/usr/bin/python3
```

```
import http.server
import socketserver
```

```
PORT = 8080
```

```
handler = http.server.SimpleHTTPRequestHandler
```

```
httpd = socketserver.TCPServer(("", PORT), handler)
print("Serveur opérationnel sur le port : ", PORT)
```

```
httpd.serve_forever()
```

→ indique le programme à utiliser pour l'exécution directe

→ bibliothèques à importer pour gestion d'un serveur web http

→ définition du port utilisé

→ définition du serveur de requêtes

→ création de l'instance du serveur

→ démarrage du serveur

**Note** : Pour démarrer directement le serveur, il faudra modifier les permissions en rendant ce programme « exécutable ». Pour cela, dans un terminal, on saisira :

```
$ chmod +x serveur.py
```

- 3) Taper l'URL suivante dans le navigateur **localhost :8080/**  
Essayer ensuite avec **127.0.0.1:8080/**
- 4) Modifier la page web et afficher de nouveau la page.
- 5) Accéder au serveur web d'une autre machine en remplaçant *localhost* par l'adresse IP de la machine.  
On pourra utiliser une des commandes ci-dessous pour déterminer les adresses des machines sur le réseau :

```
$ arp -a
```

ou

```
$ ifconfig
```

## ETAPE 2 : Un serveur avec formulaire

Dans cette étape nous allons voir comment rendre le serveur interactif avec le module **cgi** et aussi que l'on peut intégrer le code html directement dans le fichier python.

- 1) Créer dans le même répertoire les fichiers **serveur.py** et **index.py** suivants :

<b>serveur.py</b>	<b>index.py</b>
<pre>#!/usr/bin/env python3  import http.server  PORT = 8888 server_address = (":", PORT)  server = http.server.HTTPServer handler = http.server.CGIHTTPRequestHandler handler.cgi_directories = ["/"] print("Serveur actif sur le port :", PORT)  httpd = server(server_address, handler) httpd.serve_forever()</pre>	<pre>#!/usr/bin/env python3 # coding: utf-8 import cgi  form = cgi.FieldStorage() print("Content-type: text/html; charset=utf-8\n") print(form.getvalue("name"))  html = """&lt;!DOCTYPE html&gt; &lt;head&gt;   &lt;title&gt;formulaire&lt;/title&gt; &lt;/head&gt; &lt;body&gt;   &lt;form action="/index.py" method="post"&gt;     &lt;input type="text" name="name" value="Votre nom" /&gt;     &lt;input type="submit" name="send" value="Envoyer information au serveur"&gt;   &lt;/form&gt; &lt;/body&gt; &lt;/html&gt; """ print(html)</pre>

- 2) Modifier les permissions pour **index.py** : taper dans un terminal

```
$ chmod +x index.py
```

(ce n'est pas obligatoire pour **serveur.py**)

- 3) Taper l'URL suivante dans le navigateur : **localhost :8888/index.py**

- 4) Saisir des informations dans le formulaire et observer les informations transmises par le serveur en modifiant l'attribut **method** :

```
method="post" ou method="get"
```

Expliquer ce que signifie « GET » et « POST ».

- 5) Modifier le code précédent pour afficher un style différent, des images, d'autres messages...

### ETAPE 3 : Une calculatrice

- 1) Modifier le fichier **index.py** pour permettre de faire une calculatrice « *en ligne* », pour avoir un rendu de ce type :

**index.py**

## Calculatrice

**1er nombre :**

**opération**

( A = Addition ; S = Soustraction ; M = Multiplication ; D = Division

**2nd nombre :**

Le bouton *Calculer* du fichier **index.py** appelle le script **calcul.py** qui permet le calcul :

## Calculatrice

**le résultat est :**

**0.3333333333333333**

Créer ce fichier **calcul.py**.

- 2) Utiliser la calculatrice d'autres élèves sur le réseau.
- 3) Améliorer cette calculatrice pour obtenir ce genre de résultat, avec une seule saisie :

## Calculatrice

Résultat :  $6/7 = 0.8571428571428571$