



serveur web

Samuel SORIN
contact@samuelsorin.fr

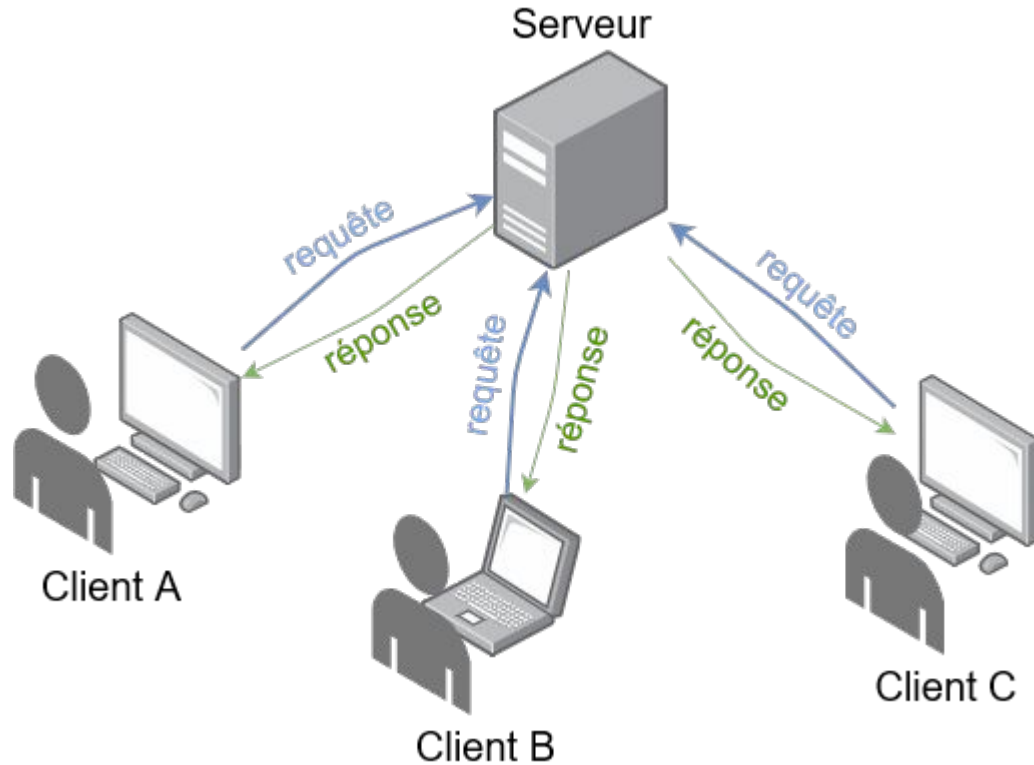
Serveur web

Serveur web - un peu de théorie : client / serveur

Les langages de programmation web sont séparés en 2 catégories :

- les technologies côté client : elles sont interprétées/exécutées par la machine cliente (html, css, js ...)
- les technologies côté serveur : elles sont interprétées/exécutées par le serveur (machine distante)
(Python, Java, PHP, ...)

Serveur web - un peu de théorie : client / serveur



Serveur web - un peu de théorie : client / serveur

Coté client

- le contenu
- la mise en forme
- les animations
- les interactions
- on récupère des infos sur l'utilisateur (mouvement du pointeur, géoposition, etc.)

Serveur web - un peu de théorie : client / serveur

Coté client

Avantage :

- fonctionne sur tous les navigateur. Donc disponible partout, par tout le monde
- pas de configuration
- pas besoin de serveur

Inconvénient :

- aucune persistance des données
- ne surtout pas y mettre de la donnée sensible (c'est complètement ouvert)
- c'est du code qui est interprété par tous les clients connectés au site
- par sécurité, les navigateurs bloquent pas mal de choses (CORS, etc.)

Serveur web - un peu de théorie : client / serveur

Coté serveur

- permet de traiter les données (trie, calcul, encodage ...)
- Définir des modèles de données
- Stocker les données (bases de données, json, xml...) pour les rendre persistantes
- on peut contrôler comment les fichiers et les contenus sont envoyés au client
- logique métier des applications
- etc, etc

Serveur web - un peu de théorie : client / serveur

Coté serveur

avantages :

- le code interprété par le serveur n'est pas visible par le client
- vous pouvez exécuter de grosses opérations de manière transparente

Inconvénients :

- vous ne pouvez pas savoir ce que l'utilisateur fait (vous voyez juste les requêtes)
- Installation et mise en place d'un serveur et de l'interpréteur correspondant au langage utilisé

Serveur web - Créer un serveur web

```
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()
```

port sur lequel le serveur sera disponible
adresse sur laquelle le serveur sera disponible
exemple : www.tonsite.com:8080

instancie le serveur
ajoute la gestion du GET et HEAD au serveur
dossier de travail, où sont placés tous les scripts.py
(il s'agit de l'adresse relative)
message pour dire que le serveur est actif

démarrage du serveur
le serveur "tourne" tant qu'on ne l'arrête pas

Serveur web - Créer une page web

```
#!/usr/bin/env python3
# coding: utf-8
import cgi

print("Content-type: text/html; charset=utf-8\n")

html = """
<!DOCTYPE html>
<head>
  <title>Mon programme</title>
</head>
<body>
  <h1>Hello World!</h1>
</body>
</html>
"""

print(html)
```

on précise l'interpréteur pour le serveur
codage du script - pour pouvoir utiliser les accents
interface cgi qui permet à Python et au client de "dialoguer"

on déclare le type de contenu que l'on va généré.
Cette information sera directement utilisé par le navigateur du client

on génère le code HTML à envoyer au navigateur du client
Cette information sera directement utilisé par le navigateur du client

on "envoie" le code généré au navigateur client

Serveur web - utiliser un formulaire

```
[...]  
  
form = cgi.FieldStorage()           # on récupère le formulaire  
name = form.getvalue("name")       # on récupère la valeur "name" présente dans le formulaire  
  
html = f"""  
[...]  
    <h1>Hello { name } !</h1>  
  
    <form action="/form.py" method="post">  
        <input type="text" name="name" value="" placeholder="Votre nom"/>  
        <input type="submit" value="Envoyer information au serveur">  
    </form>  
[...]  
"""  
  
[...]
```

Serveur web - utiles

Afficher les erreurs Python sur votre page web (debug) :

```
import cgi
cgi.enable()
```

Afficher les variables d'environnement :

Affiche toutes les informations concernant votre serveur web / page en cours

```
cgi.test()
```

Serveur web - exercice 1

a - Créer un serveur web et afficher une page html

b - Créer un nouveau serveur web, acceptant les scripts python pour afficher “Hello World” dans une page web.

c - Ajouter des paramètres GET pour modifier le message “Hello World” par “Hello *[prenom] [nom]*”, où *[prenom] [nom]* sont des paramètres

d - Utiliser un formulaire html pour envoyer les paramètres GET

e - Transformer les paramètres GET en paramètres POST

Serveur web - exercice 2 *(calculatrice)*

Créer une calculatrice qui permette de multiplier / additionner / soustraire / diviser 2 nombres.

La page web doit être composé :

- d'un formulaire comportant 3 champs : nombre1, opérateur, nombre2
- d'un texte pour afficher le résultat

Serveur web - exercice 3 *(calculatrice - avec list)*

Créer une calculatrice qui permette de multiplier / additionner / soustraire / diviser 2 nombres.

- le formulaire ne doit comporter qu'un seul champ
- utilisation des list obligatoire

Serveur web - exercice 4 *(Persistence des données)*

Créer une page web qui demande et affiche le nom et l'âge de l'utilisateur.

Le parcours doit se faire en 3 étapes :

- Formulaire de demande de nom
- Formulaire de demande d'âge
- Message affichant âge et nom

Serveur web - exercice 5 *(Liste de course)*

Créer une page web pour ajouter des éléments dans une liste.

Conseil : créez votre programme en plusieurs étapes. Par exemple,

- commencer par afficher une liste de course toute faite
- ajouter 1 élément à cette liste de course
- ...