# Interaction client-serveur - Plan de travail

1NSI – janvier 2023

Savoir-faire de la séquence

-

## 1

☐ Exercice 1 : Requêtes et réponses ..... ☆☆☆☆☆

## Exercice 1 — Requêtes et réponses

1. Ouvrir le navigateur, allez chercher les outils de développement (F 12) puis allez sur l'onglet `Réseau` (ou network).
   (a) Allez à l'adresse `raw.opytex.org`.. Faire apparaitre les colonnes `chemin` (ou path) et `url`.
   (b) En cliquant sur la requête, noter la méthode, le domaine (host), le fichier, l'adresse IP de la requête et l'état (ou status) de la réponse.
   (c) Faire la même chose pour l'adresse `https://raw.opytex.org/1NSI/05_Interaction_client-serveur/`.
   (d) La requête vers l'adresse `https://optez.org` engendre plusieurs requêtes noter les éléments (précisés à la question 1b) pour chacune d'entre elles.
2. Éléments théoriques - recherche documentaire
   (a) Expliquer le rôle que joue les méthodes vues dans les questions précédentes
   (b) Expliquer le rôle que joue les codes status vus dans les questions précédentes

## Exercice 2 — Faire son serveur web avec Bottle

Dans cette partie, vous allez programmer une application web en utilisant la librairie python `bottle`. Vous trouverez la documentation (en anglais) à l'adresse suivante

`https://bottlepy.org/docs/dev/tutorial.html`

1. **Mise en place de Bottle et vérifications**
   (a) Copier-coller le code suivant (tiré de la documentation) pour initialiser l'application.

```python
from bottle import get, run

@get('/hello')
def hello():
    return "Hello World!"

run(host='localhost', port=8080, debug=True)
```

   (b) Exécuter votre programme puis rendez-vous avec le navigateur à l'adresse `http://localhost:8080/hello`
   Étudiez la requête (méthode, domaine, fichier et status).
   (c) Rendez-vous avec le navigateur à l'adresse `http://localhost:8080/plop`. Étudiez la requête (méthode, domaine, fichier et status).
   (d) Relier ce que vous avez noté aux questions précédentes avec le code exemple de Bottle.
   (e) Comment peut-on ajouter d'autres chemins à notre application ?
2. **Requêtes GET**
   Pour toutes les questions suivantes, vous devrez tester votre travail en visitant la page associée et vous noterez l'URL.
   (a) Supprimer le chemin `/hello`.
   (b) Ajouter le chemin accessible à la racine de l'application (`/`) qui saluera les nouveaux venus.
   (c) Ajouter le chemin `/about` qui vous présentera (en une phrase).
   (d) Ajouter le chemin `/bottle` qui vous donnera le lien (cliquable) vers la documentation de Bottle.
   (e) (*) On peut créer des chemins dynamiques en suivant la syntaxe suivante

```python
@get('/page/<utilisateur>')
def page_utilisateur(utilisateur):
   return "Bonjour ", utilisateur, ". Comment allez vous?"
```

Ajouter ce chemin à votre application. Proposez plusieurs URL qui l'utilisent.

(f) On peut aussi configurer des query

```python
from botte import request, get

@get('/age')
def age_de():

   name = request.query.name
   age = request.query.age

   if age > 1:
      return name, " a ", age, " ans."
   elif age <= 0:
      return "Mouai..."
   else:
      return name, " a ", age, " an."
```

Ajouter ce chemin à votre application sans oublier d'ajouter les nouveaux imports. En reprenant votre cours sur les URL, trouver l'URL qui permet d'avoir sur votre navigateur le message

Bob a 5 ans.

(g) Pour le moment, le code renvoyé par notre application n'est pas du code HTML valide. Il est possible d'écrire des modèles (template en anglais) dans lesquels on va pouvoir injecter des variables python pour les rendre modulables.
À côté du fichier de votre application, créer un dossier `views/` puis un fichier `utilisateur.html` dans lequel vous mettrait le code HTML suivant

```html
<!DOCTYPE html>
<html lang=fr>
      <head>
            <meta charset="UTF-8">
            <meta name="Author" content="">
      <title>Page de {{utilisateur}}</title>
      </head>

      <body>
      <h1>{{ utilisateur }}</h1>
      <p>Cette page change en fonction de l'utilisateur.</p>
      <p>Elle sait même faire des calculs 1+1 = {{1+1}}</p>
      </body>
</html>
```

Ajoutez la fonction `template` dans les imports de bottle. Puis modifier le chemin `page_utilisateu`
pour avoir le contenu suivant

```python
@get('/page/<utilisateur>')
def page_utilisateur(utilisateur):
   return template("utilisateur.html", utilisateur=utilisateur)
```

Visitez plusieurs pages utilisateur et constatez les différences.
Comment fait-on pour utiliser une variable python dans modèle ?