Chap_XVI_Interactions_client-serveur

March 13, 2020

0.1 Interactions client - serveur

1 Modèle

On appelle interaction client $\leftrightarrow$ serveur un mode de communication entre un programme qualifié de *client* qui **envoie des requêtes** et un programme dit *serveur* qui y répond.
Dans le cas du Web, le client peut être un navigateur (*Google Chrome, Firefox, etc*) et le serveur un programme tel que *Apache, Nginx, etc*. Dans le cadre de ce cours, on utilisera un serveur écrit en Javascript (*Node.js* plus exactement). Le protocole utilisé est **HTTP**.

2 Analyse d'un échange

▷ Ouvrir un terminal et taper la commande ci-dessous, en remplaçant la séquence `xxx.xxx.xxx.xxx` par l'adresse ip fournie par le professeur:

```
curl -v http://xxx.xxx.xxx.xxx:4000
```

▷ Commenter le résultat obtenu.

Les lignes précédées du signe $>$ constituent les requêtes du client et celles précédées du signe $<$ sont les réponses du serveur.
Les requêtes ont toujours la forme:

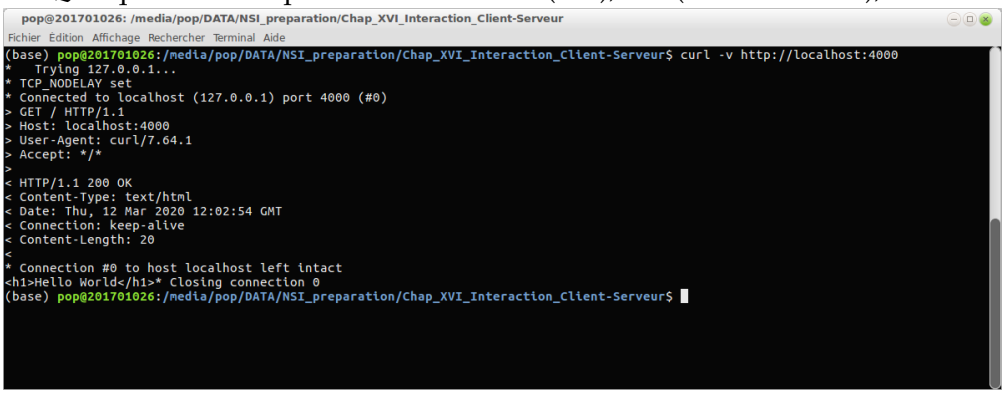
```
commande URL version_protocole
Entête
--> ligne vide <--
corps de la requête
```

`commande` est la méthode à utiliser, elle spécifie le type de requête (voir paragraphe suivant). On peut citer, entre autres: **GET**, **POST** ou **HEAD**.

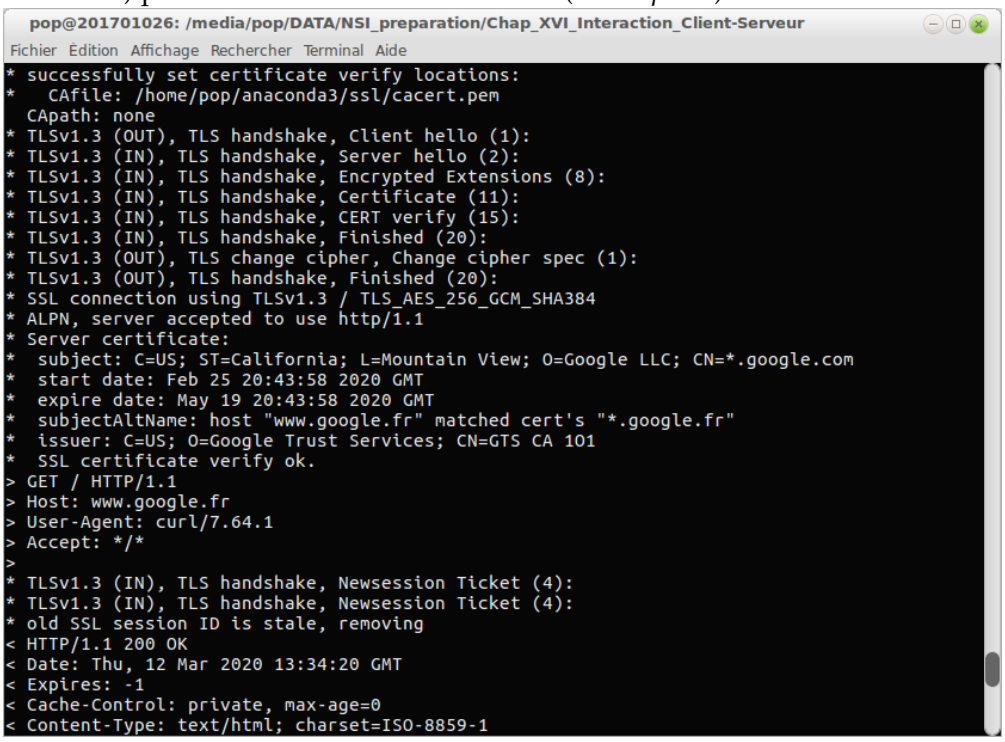
De même, les réponses suivent la syntaxe:

```
version_protocole code_réponse texte_réponse
Entête
--> ligne vide <--
corps de la réponse
```

Quelques codes réponse courants: 200 (OK), 404 (NOT FOUND), etc.

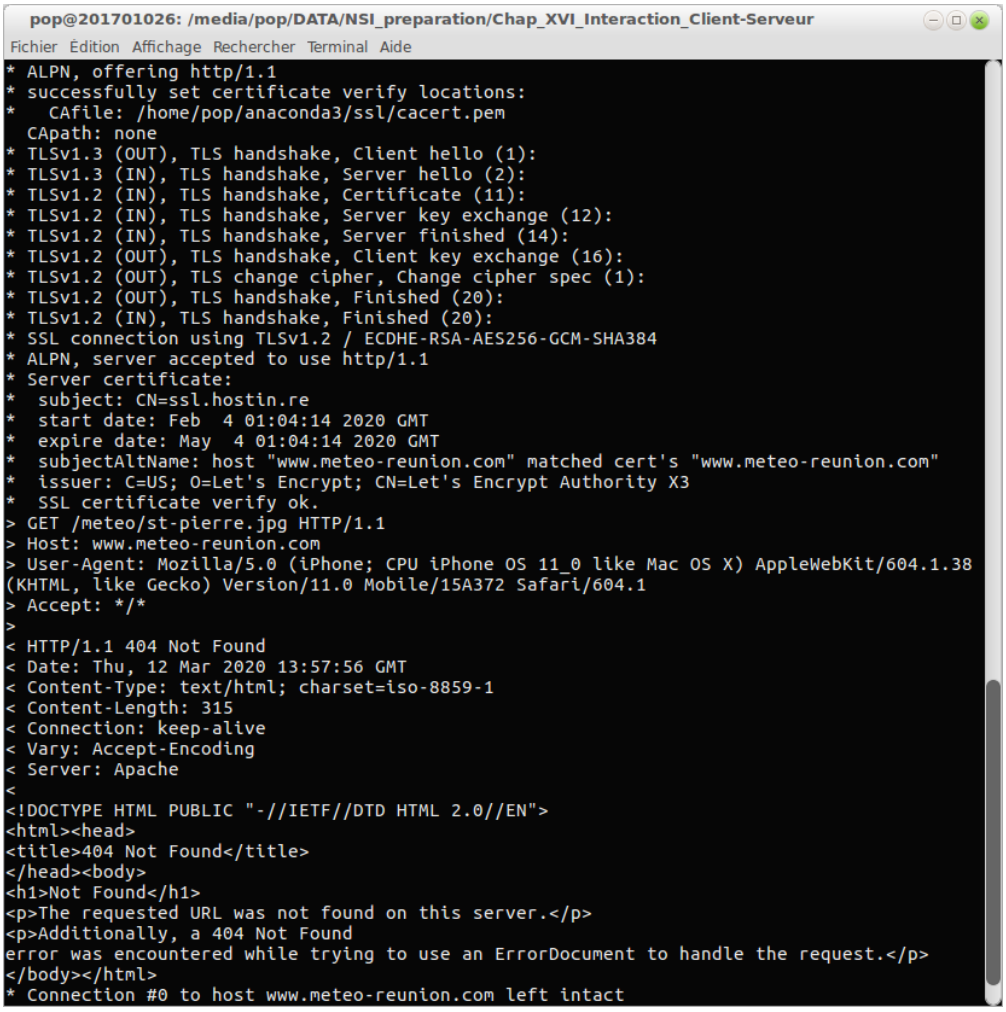


De plus en plus, les échanges entre client et serveur sont chiffrés. De telles communications utilisent la version *sécurisée* **HTTPS** du protocole. Les négociations de chiffrement (on dit aussi le *handshake*) précèdent les envois de données (*voir capture*).

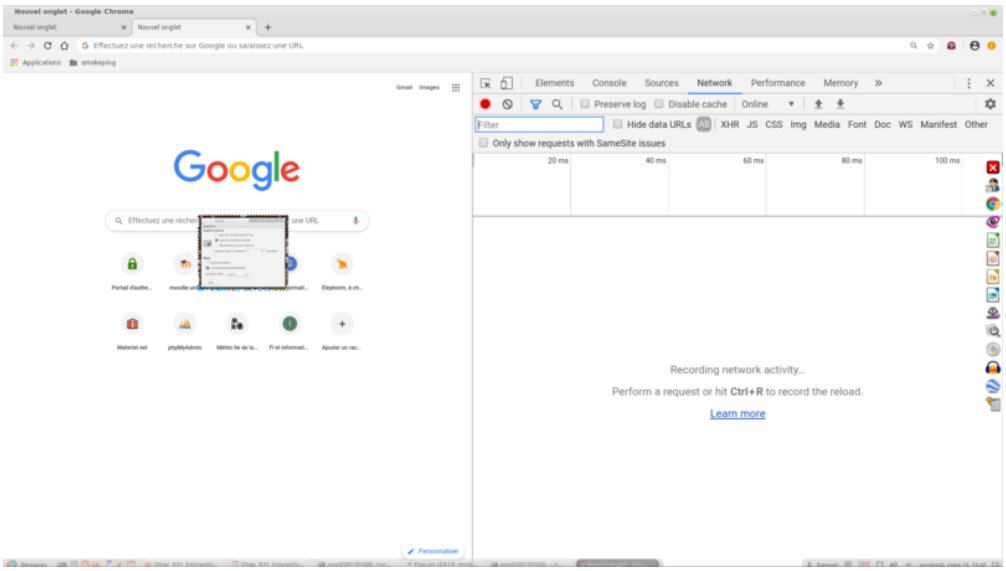


2.0.1 Exercice

A partir de la capture ci-dessous, retrouver les informations suivantes:
* la méthode de la requête ainsi que l'url de la ressource demandée; * le type de serveur et le code de réponse retourné. Quelle en est la signification ?
* Quel est le type de document renvoyé par le serveur et quel est sa taille ?


pop@201701026: /media/pop/DATA/NSI_preparation/Chap_XVI_Interaction_Client-Serveur
Fichier Édition Affichage Rechercher Terminal Aide
* ALPN, offering http/1.1
* successfully set certificate verify locations:
*   CAfile: /home/pop/anaconda3/ssl/cacert.pem
  CApath: none
* TLSv1.3 (OUT), TLS handshake, Client hello (1):
* TLSv1.3 (IN), TLS handshake, Server hello (2):
* TLSv1.2 (IN), TLS handshake, Certificate (11):
* TLSv1.2 (IN), TLS handshake, Server key exchange (12):
* TLSv1.2 (IN), TLS handshake, Server finished (14):
* TLSv1.2 (OUT), TLS handshake, Client key exchange (16):
* TLSv1.2 (OUT), TLS change cipher, Change cipher spec (1):
* TLSv1.2 (OUT), TLS handshake, Finished (20):
* TLSv1.2 (IN), TLS handshake, Finished (20):
* SSL connection using TLSv1.2 / ECDHE-RSA-AES256-GCM-SHA384
* ALPN, server accepted to use http/1.1
* Server certificate:
* subject: CN=ssl.hostin.re
* start date: Feb 4 01:04:14 2020 GMT
* expire date: May 4 01:04:14 2020 GMT
* subjectAltName: host "www.meteo-reunion.com" matched cert's "www.meteo-reunion.com"
* issuer: C=US; O=Let's Encrypt; CN=Let's Encrypt Authority X3
* SSL certificate verify ok.
> GET /meteo/st-pierre.jpg HTTP/1.1
> Host: www.meteo-reunion.com
> User-Agent: Mozilla/5.0 (iPhone; CPU iPhone OS 11_0 like Mac OS X) AppleWebKit/604.1.38
(KHTML, like Gecko) Version/11.0 Mobile/15A372 Safari/604.1
> Accept: */*
>
< HTTP/1.1 404 Not Found
< Date: Thu, 12 Mar 2020 13:57:56 GMT
< Content-Type: text/html; charset=iso-8859-1
< Content-Length: 315
< Connection: keep-alive
< Vary: Accept-Encoding
< Server: Apache
<
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>404 Not Found</title>
</head><body>
<h1>Not Found</h1>
The requested URL was not found on this server.
Additionally, a 404 Not Found
error was encountered while trying to use an ErrorDocument to handle the request.
</body></html>
* Connection #0 to host www.meteo-reunion.com left intact

exercice



chrome

3 Transmission de paramètres

3.1 Methode GET

▷ Ouvrir deux onglets différents dans le navigateur Chrome, en activant la console de développement web (touche F12) et sélectionnt 'Network' dans cette console (à droite).

▷ Entrer les deux adresses suivantes (dans lesquelles on modifiera la chaine `xxx`), chacune dans un onglet:

```
http://xxx.xxx.xxx.xxx:4000/date
http://xxx.xxx.xxx.xxx:4000/date?format=svg
```

▷ Commenter les résultats

▷ Dans les consoles de développement, retrouver (onglet *Headers* puis *Request Headers*) la méthode de requête utilisée ainsi que l'url demandée.

A RETENIR On peut passer des paramètres au serveur via l'url. Les paramètres d'une URL servent à influencer la représentation d'une ressource ou information. La méthode GET permet de transmettre des paramètres.

Remarque: on peut passer plusieurs paramètres dans l'url; par exemple pour transmettre deux paramètres `para1` et `para2` on ajoute à l'url:

```
?para1=valeur&para2=valeur2
```

3.2 Méthode POST - Cas de formulaire

3.3 Qu'est-ce qu'un formulaire ?

Les formulaires HTML sont les principaux outils d'interaction entre un utilisateur et un serveur. Ils permettent **d'envoyer des données** au serveur web. ## Syntaxe Pour ajouter un formulaire à une page HTML, on utilisera la balise `form`:

```
<form action="url_de_traitement" method="POST">
    ...
    ...
</form>
```

`url_de_traitement` est à adapter ! A l'intérieur d'un formulaire, on place généralement des *contrôles* repérés par des balises `<input>` dont le rendu dans le navigateur va dépendre de son attribut `type`. Les contrôles `<input>` sont souvent associés à une légende dont le contenu est fixée par une balise `<label>`. Par exemple, le code

```
<label for="nom">Nom (entre 4 et 8 caractères):</label>
<input type="text" id="nom" name="nom" minlength="4" maxlength="8" size="10">
```

fournira le rendu:

```
In [2]: from IPython.core.display import HTML
        HTML("""
        <form action="localhost:4000" method="POST">
            <label for="nom">Nom (entre 4 et 8 caractères):</label>
            <input type="text" id="nom" name="nom" minlength="4" maxlength="8" size="10">
        </form>
        """)

Out[2]: <IPython.core.display.HTML object>
```

▷ Ouvrir le fichier `index.html` avec Visual Studio Code. Repérer le formulaire. Combien de contrôles sont présents ?
▷ Modifier l'url de traitement de la balise `<form>` selon les instructions du professeur.
▷ Ouvrir le fichier `index.html` avec le navigateur Chrome ainsi que la console de développement web.
▷ Valider le formulaire. Commenter les résultats, côté client puis côté serveur.

```
In [ ]:
```

```
In [ ]:
```