Première NSI

Chapitre XI - Interaction client - serveur

I. Généralités

Un utilisateur qui veut consulter un site internet fait une requête sur son ordinateur, appelé **client**.
Cette dernière est envoyée grâce au protocole HTTP(S) à un autre ordinateur, appelé **serveur**.

Une des requêtes possibles est l'accès à une page Web :

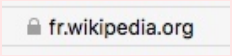
- `http://` ou `https://` qui correspond au protocole de communication entre le client et le serveur ;
- un nom de domaine ;
- un chemin qui pointe vers une page précise.

Pour faire ces requêtes, on peut alors utiliser des méthodes dont les méthodes `GET` et `POST`. Elles sont utilisées principalement dans les formulaires des pages Web.
Elles envoient au serveur des paramètres qui seront utiles pour l'affichage de la page Web.
Le serveur reçoit la requête (et les éventuels paramètres) et répond sous forme de texte qui sera mémorisé par le client et affiché par son navigateur (contenu en HTML, mise en page en CSS, dynamisation de la page en Javascript).

Le serveur peut faire certaines actions grâce à la programmation en PHP, langage équivalent au javascript mais du côté serveur.

Remarque

Si on veut chiffrer le texte envoyé par le serveur, on utilise le protocole HTTPS (HyperText Transfer Protocol Secure). On reconnaît l'utilisation du protocole HTTPS par le petit cadenas devant l'adresse URL dans les navigateurs.



HTTPS permet au visiteur de vérifier l'identité du site web auquel il accède, grâce à un certificat d'authentification émis par une autorité tierce, réputée fiable. Il garantit théoriquement la confidentialité et l'intégrité des données envoyées par l'utilisateur (notamment des informations entrées dans les formulaires) et reçues du serveur.
Il peut permettre de valider l'identité du visiteur, si celui-ci utilise également un certificat d'authentification client.

Pour plus d'informations : voir HTTP.

II. Les formulaires

Les formulaires d'une page Web permettent, via le protocole HTTP, d'envoyer des paramètres au serveur. Les deux méthodes que nous utiliserons, sont les méthodes `GET` et `POST`.

II.1. La methode GET

La méthode `GET` ' va encoder les données du formulaire dans l'URL de la page. Elle seront visibles après le `?` et séparées par un `&`.
On peut récupérer ces données grâce au langage PHP mais aussi au Javascript.
On rappelle que la taille limite d'une URL est de 2000 caractères. Donc la méthode `GET` est limitée.

II.2. La methode POST

La methode `POST` envoie un en-tête et un corps de message au serveur. Le corps est généralement constitué des données entrées dans le champ de formulaire par l'utilisateur. Les données du formulaire n'apparaissent pas dans l'URL. En conséquence, il n'est pas possible de récupérer directement les données en JavaScript , il faut ajouter du code PHP dans la page.
La méthode `POST` est plus sécurisée que la méthode `GET` puisqu'on ne voit pas les données dans l'URL.

II.3. Création d'un formulaire

II.3.a. La balise <form>

Selon la méthode choisie, on écrira :

- Méthode `GET` avec page HTML

```
1 <form method="get" action="page.html">
2
3 </form>
```

- Méthode `GET` avec page PHP

```
1 <form method="get" action="page.php">
2
3 </form>
```

- Méthode `POST` avec page PHP

```
1 <form method="post" action="page.php">
2
3 </form>
```

Le client fait une requête au serveur afin d'ouvrir la page indiquée dans l'attribut `action` de la balise `<form>`.
Il envoie aussi tous les données du formulaire. Ces dernières sont indiquées dans les balises `<input>`.

II.3.b. La balise <input>

Les balises `<input>` qui ne possède pas de balise fermante.
Les balises `<input>` ont un attribut `type` qui indique le type d'affichage pour la balise, un attribut `name` qui donne un nom à la balise et un attribut `value` qui indique la valeur envoyée pour le nom.

Les différents types sont :

- un texte :

```
1 <input type="text" name="un_nom" value="">
```

Rendu :

Remarque

La valeur de cette balise correspond à ce que va rentrer l'utilisateur dans la zone de texte.

- un bouton radio :

```
1 <input type="radio" name="unNom" value="uneValeur"><br>
2 <input type="radio" checked name="unNom" value="uneAutreValeur">
```

Rendu :

Remarque

L'attribut `checked` coche automatiquement le bouton radio.
Si plusieurs balises `<input>` sont de type radio et ont le même nom, alors une seule pourra être choisie par l'utilisateur.
Seule la valeur du bouton radio coché sera envoyé.

- une case à cocher :

```
1 <input type="checkbox" name="unNom" value="uneValeur"><br>
2 <input type="checkbox" checked name="unAutreNom" value="uneAutreValeur">
```

Rendu :

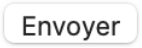
Remarque

L'attribut `checked` coche automatiquement la case à cocher.
Il est nécessaire d'indiquer des noms différents à chacune des cases à cocher.
Seules les valeurs des cases cochés seront envoyées.

- un bouton de soumission :

```
<input type="submit" name="unNom" value="Envoyer"><br>
```

Rendu :



Remarque

La valeur d'un bouton de soumission est la chaîne de caractères affichée sur ce bouton.
Lorsque l'utilisateur appuie sur ce bouton, la requête est envoyée.

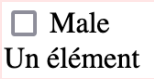
- D'autres types à voir sur W3schools.

Remarque

Il n'est pas forcément aisé de cliquer sur un bouton radio(ou une boite à cocher). On peut alors utiliser la balise `<label>...</label>` qui permet de cliquer sur l'élément de cette balise pour cocher le bouton radio(ou la boite à cocher) correspondant. Cette balise utilise un attribut `for` dont la valeur doit être égale à l'identifiant du bouton (ou de la boite à cocher).

```
<input type="checkbox" name="unAutreNom" id="unIdentifiant" value="valeur1"> Male<br>
<label for="unIdentifiant">Un élément</label>
```

Rendu :



Cliquer sur « Male » ne produira rien, mais cliquer sur « Un élément » cochera la case à cocher.

III. Exercices

III.1. Exercice 1

1. Créer une page `index.html` dans laquelle un formulaire pointe, avec la méthode `GET` vers la page `reponse.html`. Le formulaire devra contenir :
   - la demande du nom et du prénom
   - le genre de l'utilisateur
   - si l'utilisateur aime la purée, les saucisses et/ou les petits pois
2. Créer une page `reponse.html` qui affichera le nom, le prénom et le genre de l'utilisateur ainsi que les aliments qu'il aime grâce à un fichier `formulaire.js`.

III.2. Exercice 2

1. Créer une page `index.html` dans laquelle un formulaire pointe, avec la méthode `POST` vers la page `reponse.php`. Le formulaire devra contenir :
   - la demande du nom et du prénom
   - le genre de l'utilisateur
   - si l'utilisateur aime la purée, les saucisses et/ou les petits pois
2. Créer une page `reponse.php` qui affichera le nom, le prénom et le genre de l'utilisateur ainsi que les aliments qu'il aime.