1 Introduction

Un formulaire de page Web permet de collecter des informations de la part du **client** pour les transférer au **serveur** en vue d'un traitement.
Il est fréquent (*mais non obligatoire*) qu'un script Javascript accompagne le formulaire pour vérifier qu'il est correctement rempli (cette tâche s'effectue sur la machine cliente) avant d'envoyer les données du formulaire au serveur.
Sur le serveur, un script écrit dans un langage de programmation (souvent **PHP**, mais d'autres langages comme Python sont possibles) va traiter les données et souvent renvoyer une nouvelle page au client.

2 Formulaire et HTML

2.1 Fonctionnement général

Un formulaire HTML repose sur l'élément `form` et des éléments contrôles.

L'élément principal est `input` qui permet la saisie de données par l'utilisateur.
Son attribut `type` permet de préciser la nature des données à saisir.
Son attribut `name` prend une valeur qui sera le nom de la variable contenant la donnée saisie et qui pourra être traitée au niveau du serveur.

L'élément `form` possède 2 attributs particuliers :

- `action` : sa valeur indique l'URL du fichier script permettant de traiter les données extraites du formulaire.
- `method` : sa valeur indique la façon de transmettre les données du formulaire par le protocole HTTP.

Un formulaire finit en général par un élément `input` avec un attribut `type` à la valeur `submit`, qui a pour effet d'envoyer le formulaire lorsqu'on clique sur le bouton correspondant.

2.2 Exemples

Penser aussi à consulter la référence annexe HTML/CSS.

```
<form action="chemins/vers/script.php" method="POST">
  <label for="nom">Nom : </label> <input type="text" name="nom_data" id="nom"/> <br/>
  <input type="reset" value="RAZ" />
  <input type="button" value="Gérer le clic en JS" onclick="alerte()">
  <input type="submit" value="Envoyer">
</form>
```

Cases à cocher (élément input type checkbox) :

```
<form action="chemins/vers/script.php" method="GET">
  Cochez vos aliments préférés :<br />
  <input type="checkbox" name="frites_data" id="frites" checked />
  <label for="frites">Frites</label><br />
  <input type="checkbox" name="steak_data" id="steak" />
  <label for="steak">Steak haché</label><br />
  <input type="submit" value="Envoyer">
</form>
```

Boutons radio (élément input type radio) :

```
<form action="chemins/vers/script.php" method="POST">
  Quelle tranche d'âge ?<br />
  <!-- les boutons radio partagent le même attribut "name" -->
  <input type="radio" name="age_data" value="enfant" id="15" />
  <label for="15">Moins de 12 ans</label><br />
  <input type="radio" name="age_data" value="ado" id="medium12-18" checked />
  <label for="medium12-18">12-18 ans</label><br />
  <input type="radio" name="age_data" value="adulte" id="plus18" />
  <label for="plus18">Plus vieux  ?</label>
  <input type="submit" value="Envoyer">
</form>
```

Listes déroulantes (élément select) :

```
<form action="chemins/vers/script.php" method="GET">
  <label for="pays">Choisir un pays :</label><br />
  <select name="pays_data" id="pays">
    <optgroup label="Europe">
      <option value="france">France</option>
      <option value="espagne">Espagne</option>
    </optgroup>
    <optgroup label="Amérique">
      <option value="canada">Canada</option>
      <option value="etats-unis">États-Unis</option>
    </optgroup>
  </select>
  <input type="submit" value="Envoyer">
</form>
```

3 Transmission des données : méthodes GET et POST

L'attribut `action` de l'élément `form` permet de définir l'URL vers laquelle les données du formulaire seront transmises.

L'attribut `method` de l'élément `form` indique quelle méthode le protocole HTTP va utiliser pour cette effectuer cette transmission. Les méthodes principales sont **GET** ou **POST**.

3.1 Méthode GET

La méthode **GET envoie les données en les ajoutant à l'URL de destination**. L'ajout se fait après un point interrogation, et les différentes données sont séparées par une esperluette et présentées sous la forme `donnee=valeur`.

Le corps de la requête HTTP reste vide dans le cadre de cette méthode.

Exemple : `https://nom_domaine/script.php?donnee_1=valeur_1&donnee_2=valeur_2`
`donnee_i` est la valeur de l'attribut `name` de chaque élément `input`, et `valeur_i` est la donnée saisie par l'utilisateur ou bien la valeur de l'attribut `value`.

La méthode GET est normalement utilisée lorsqu'aucune modification des données ne doit avoir lieu : elle opère donc une simple lecture.

Elle ne peut être utilisée que pour envoyer des données textuelles ASCII.

3.2 Méthode POST

La méthode **POST envoie les données dans le corps de la requête HTTP**. L'URL de destination ne subit donc aucune modification.

La méthode POST est normalement utilisée lorsqu'un traitement sur les données est nécessaire.

Tout type de données (pas seulement du texte ASCII) peut être transmis par cette méthode.

3.3 Transmission sécurisée ?

On lit parfois que la méthode POST est plus sûre que la méthode GET car on ne peut pas voir en clair dans l'URL les données transmises (*ex* : mot de passe) ; c'est vrai, mais rien n'empêche d'intercepter les paquets sur le réseau et de lire ainsi les données. Ce n'est pas donc pas une bonne méthode pour transmettre des données de façon sécurisée.

Comme on l'a vu au chap. P4-3, la seule façon de transmettre des données sensibles sans risque est d'utiliser le protocole **HTTPS qui chiffre les données avant leur transmission**.