

# Cours Web n°4

## Introduction à HTTP — Formulaires XHTML

Sandrine-Dominique Gouraud (gouraud@lri.fr)  
Pierre Senellart (pierre@senellart.com)



Semaine du 10 octobre 2005

# Plan du cours

- 1 HTTP
- 2 Formulaires XHTML
- 3 Références
- 4 Application

## Rappel

- Un client C (p.ex. un navigateur Web), sur une machine X.
- Un serveur Web S sur une machine Y.
- C se connecte à Y.
- C demande à S une URL, accompagnée de **paramètres**.
- S répond à C en lui renvoyant p.ex. une page Web. Cette page peut être un document statique (p.ex. un fichier XHTML) ou une page **dynamique** produite par un programme (p.ex. un script PHP).
- S ferme la connexion à C.

- Protocole de communication à la base du World Wide Web.
- Requête du client :

```
GET /MarkUp/ HTTP/1.1
Host: www.w3.org
```

- Réponse du serveur :

```
HTTP/1.1 200 OK
...
Content-Type: text/html; charset=utf-8

<!DOCTYPE html ...>
<html ...>
...
</html>
```

- Deux **méthodes** HTTP (types de requêtes) : GET et POST.
- Possibilité de passer des paramètres (paires clef/valeur).

- Type de requête la plus simple.
- Les paramètres (éventuels) sont passés à la fin de l'URL, après un ' ?'
- Ne convient pas quand il y a beaucoup de paramètres ou que leurs valeurs sont très longues.
- Méthode utilisée quand on tape une URL, quand on suit un lien...

### Exemple (Recherche Google)

URL : `http://www.google.fr/search?hl=fr&q=hello`

Requête HTTP GET correspondante :

```
GET /search?hl=fr&q=hello
```

```
Host: www.google.fr
```

- Méthode pouvant être utilisée uniquement pour des formulaires.

## Exemple

```
POST /php/test.php HTTP/1.1
```

```
Host: pierre.senellart.com
```

```
Content-Type: application/x-www-form-urlencoded
```

```
Content-Length: 100
```

```
type=recherche&titre=Le+Dictateur&format=long&pays=US
```

- Par défaut, les paramètres sont passés (en GET ou en POST) sous la forme : `name1=valeur1&name2=valeur2...`, et les caractères spéciaux (caractères accentués, espaces...) sont remplacés par des codes comme `+,%20...`. Cette manière de passer les paramètres est nommée `application/x-www-form-urlencoded`.
- Pour la méthode POST, on peut aussi choisir un encodage plus lourd (plusieurs lignes par paramètre), similaire à la façon dont les e-mails sont construits; c'est surtout utile pour passer de grandes quantités d'information. On parle d'encodage `multipart/form-data`.

# Plan du cours

- 1 HTTP
- 2 Formulaires XHTML**
- 3 Références
- 4 Application

- Un formulaire XHTML est placé à l'intérieur d'une balise <form>.
- Celle-ci prend les attributs suivants :

**action** URL du script auquel sera soumis le formulaire.  
**method** Méthode HTTP, valant soit GET soit POST.  
**enctype** Encodage HTTP. Peut valoir  
application/x-www-form-urlencoded (valeur par défaut) ou multipart/form-data.

### Exemple (Formulaire élémentaire)

```
<form action="action.php" method="GET">  
  <div><input type="submit" /></div>  
</form>
```

En XHTML, il est interdit de mettre des champs de formulaire directement à l'intérieur d'un `<form>`. Il faut d'abord les regrouper :

- Dans des paragraphes `<p>` si les champs de formulaires sont à l'intérieur de paragraphes de textes (rare).
- Dans des ensembles de champ `<fieldset>` pour regrouper des champs de formulaire de sémantique proche. On pourra alors donner une légende à l'ensemble de champs avec la balise `<legend>`.
- Dans des divisions `<div>` sans contenu sémantique sinon.

### Exemple (Ensemble de champ)

```
<fieldset>
  <legend>Mensurations</legend>
  <input type="text" name="taille" />
  <input type="text" name="poids" />
</fieldset>
```

- La plupart des champs sont naturellement accompagnés d'une **étiquette** (`<label>`).
- On peut la placer où on veut, en général juste à gauche ou à droite du champ.
- Son attribut `for` référence l'attribut `id` du champ correspondant.

### Exemple (Étiquette)

```
<label for="taille">Taille :</label> <input type="text"
name="taille" id="taille" />
```

- La balise `<input>` a une utilisation très vaste dans les formulaires. Elle représente un champ de saisie.
- L'attribut `type` détermine le type (texte, mot de passe, liste, etc.) du champ.
- L'attribut `name` (nom du paramètre de la requête PHP) est **obligatoire** (sauf pour `reset` et `submit`); il permet de préciser au serveur à quelle saisie on fait référence.

### Exemple (Zone de texte pour écrire des commentaires)

```
<input type="text" name="Commentaires" />
```

- `type="text"` est utilisé pour la saisie d'un texte dont la taille est inférieure à une ligne.
- L'attribut `value` permet de préciser la valeur par défaut.
- La taille maximale de la chaîne de caractères à saisir peut être spécifiée à l'aide de l'attribut `maxlength`.

### Exemple

```
<input type="text" name="prenom" value="Jordy"
maxlength="50" />
```

- `type="password"` est utilisé pour la saisie d'un texte dont les caractères sont remplacés par des astérisques : c'est généralement utilisé pour la saisie des mots de passe. Le mot de passe est quand même transmis en clair au serveur !
- L'attribut `value` permet de préciser la valeur par défaut.
- La taille maximale de la chaîne de caractères à saisir peut être spécifiée à l'aide de l'attribut `maxlength`.

### Exemple

```
<input type="password" name="password" value="12345678"
maxlength="8" />
```

- `type="checkbox"` permet de choisir plusieurs éléments parmi une liste de possibilités.
- Cela se matérialise sous forme de cases à cocher.
- La valeur retournée est **obligatoirement** précisée à l'aide de l'attribut `value`.
- L'attribut `checked="checked"` permet de cocher la case par défaut.
- Pour que les choix multiples soient traités correctement par PHP, on donnera toujours un nom se terminant par `[]`.

## Exemple

```
<input type="checkbox" name="pub[]" value="externe"
checked="checked" id="pub-ex" />
<label for="pub-ex">Recevoir des offres de notre
site</label>
<input type="checkbox" name="pub[]" checked="checked"
value="site" id="pub-site" /> <label for="pub-site">Recevoir
des offres de nos partenaires</label>
```

- `type="radio"` permet de choisir un seul élément parmi une liste de possibilités.
- Cela se matérialise sous forme de boutons radio.
- La valeur retournée est **obligatoirement** précisée à l'aide de l'attribut `value`.
- L'attribut `checked="checked"` permet de préciser la valeur par défaut.

## Exemple

Recevoir de la pub :

```
<input type="radio" name="pub" value="oui" checked="checked"
/> <label for="pub-oui">oui</label>
<input type="radio" name="pub" value="non" id="pub-non" />
<label for="pub-non">non</label>
```

- `type="file"` permet de joindre au formulaire un fichier.
- À cause de la taille de la requête due au téléchargement (**upload**) du fichier, il faut impérativement utiliser la méthode POST et l'encodage `multipart/form-data`.

Exemple (Extrait de <http://validator.w3.org/>)

```
Local File : <input type="file" name="uploaded_file" />
```

- `type="hidden"` permet de cacher des champs au client mais leur contenu est envoyé avec le formulaire.
- Ceci permet de préciser des informations, en utilisant l'attribut `value`, concernant l'interaction client/serveur.
- C'est à utiliser **avec précaution** car cela peut être à l'origine de problèmes de sécurité assez graves : ne pas oublier que le client peut éditer la page à la main pour changer la valeur de ces champs !

## Exemple

```
<input type="hidden" name="monnaie utilisee" value="EUR" />  
<input type="hidden" name="customerCB"  
value="c2415-345-8563" />
```

- `type="reset"` permet de réinitialiser le formulaire en affectant aux différents champs leur valeur par défaut.
- L'attribut `value` permet de donner un nom différent de *Reset* au bouton correspondant.

### Exemple

```
<input type="reset" value="Tout effacer" />
```

- `type="submit"` permet de soumettre le formulaire.
- Le client envoie le contenu du formulaire à l'adresse précisée par l'attribut `action` de la balise `form`.
- L'attribut `value` permet de spécifier l'étiquette du bouton.

### Exemple

```
<input type="submit" value="Envoyer" />
```

- Pour les saisies multiligne, on utilise la balise `textarea`
- Le texte délimité par cette balise permet d'initialiser la valeur par défaut du champ.
- La balise fermante est **obligatoire** même si le champ est vide.
- Les attributs `rows` et `cols` permettent de spécifier la taille en lignes et colonnes de la fenêtre de saisie.

## Exemple

```
<textarea name="bio" cols="40" rows="5">Fille de Josiane  
Balasko, Marilou Berry fait ses premiers pas à l'écran à 8  
ans...</textarea>
```

- La balise `select` permet d'afficher un menu de sélection (ou menu déroulant) :
  - ▶ L'attribut facultatif `size` permet de préciser le nombre de choix apparaissant sur la page Web. Par défaut, ce nombre est initialisé à 1.
  - ▶ L'attribut `multiple="multiple"` permet d'autoriser des réponses multiples. Dans ce cas, pour PHP, on donnera toujours un nom se terminant par `[]`.
- Les choix du menu sont indiqués à l'aide de la balise `option` :
  - ▶ L'attribut `selected="selected"` permet de spécifier le choix par défaut
  - ▶ L'attribut `value` permet de spécifier la valeur associée au choix

## Exemple

```
<select name="age">
  <option value="20">Moins de 20 ans</option>
  <option value="35" selected="selected">21 à 35 ans</option>
  <option value="50">36 à 50 ans</option>
  <option value="51">Plus de 51 ans</option>
</select>
```

- Traditionnellement, on utilisait des tables pour mettre en page un formulaire. **Interdit** désormais! Les tables servent à présenter du contenu intrinséquement tabulaire; CSS est là pour la mise en page.
- Le flottement, le positionnement absolu par rapport au formulaire, le centrage. . . sont autant d'outils utiles pour mettre en page un formulaire avec CSS.
- Les feuilles de styles peuvent utiliser avec profit les éléments sémantiques existants (`fieldset`, `legend`, `label`).

### Astuce

Il peut parfois être utile de considérer certains éléments normalement en-ligne (p.ex. les `label`) comme des éléments blocs (p.ex. pour les faire flotter). Pour cela, on utilisera `label { display: block; }`.

# Plan du cours

- 1 HTTP
- 2 Formulaires XHTML
- 3 Références**
- 4 Application

- Request For Comment 2616, Hypertext Transfer Protocol — HTTP/1.1  
<http://www.faqs.org/rfcs/rfc2616.html>
- Spécification de XHTML 1.0  
<http://www.w3.org/TR/xhtml1/>
- Spécification de HTML 4.01  
<http://www.w3.org/TR/REC-html40/>
- *HTML et XHTML : La Référence*, O'Reilly

# Plan du cours

- 1 HTTP
- 2 Formulaires XHTML
- 3 Références
- 4 Application**

- Reproduire le formulaire modèle de la page du cours (d'abord la version sans CSS, ensuite la version avec).
- Utiliser comme action le script exemple disponible à l'URL `http://pierre.senellart.com/php/test.php`.
- Comparer à la fois le formulaire et la page obtenue en soumettant le formulaire.
- Valider !