Accessibilité des formulaires

L'attribut `action`

Pour qu'un formulaire soit fonctionnel, un seul attribut est obligatoire:
l'attribut "`action`" qui permet de définir l'*URI* du script ou de la page qui traitera les informations soumises par le formulaire.
Si l'attribut est absent ou sans valeur, l'action par défaut est de rappeler la page courante.

L'attribut `method`

Cet attribut définit la façon d'envoyer le formulaire (en **`post`** ou en **`get`**).
Il s'agit de **méthodes du protocole http**.

```
<form action="index.html" method="get">
```

Si l'attribut est absent, la méthode par défaut est **`get`**.

Différences entre get et post

- **`get`**
  les `name` et les `value` des éléments de formulaire sont ajoutés à la *querystring* sous forme de couple **propriété=valeur.**

  Chaque couple est séparé par une esperluette (&).
  Les données sont séparées de l'adresse de la page (la cible de l'attribut action) par un point d'interrogation (?).

  Exemple : http://domaine.ca/formulaire.html**?nom=fred&courriel=fred@gmail.com**

  Note : En `get`, une personne mal intentionnée peut modifier les valeurs directement dans la barre d'adresse de son navigateur et resoumettre les données en rafraichissant la page avec une *querystring* modifiée.

- **`post`**
  les couples de `name` et `value` des éléments de formulaire sont envoyés à la cible de l'attribut `action` **de manière cachée.** Cette méthode est un peu plus sécuritaire que la méthode `get`, toutefois elle n'est pas infaillible.

Éléments HTML pour soumettre un formulaire

1.

```
<input   type="submit"
         name="btnCommander"
         value="Commander"  />
```

La technique traditionnelle pour déclencher l'envoi d'un formulaire est d'ajouter un `input` de type `"submit"`. Celui-ci envoie automatiquement la valeur de son `name` associée à la valeur de sa `value`.
Exemple: `btnCommander=Commander`

2.

```
<input   type="image"
         name="btnCommander"
         value="commander"
         src="images/btnCommander.gif"
         alt="Commander" />
```

Celui-ci envoie ses coordonnées x et y, en plus d'envoyer la valeur de son `name` associée à la valeur de sa `value`.
Exemple: `btnCommander.x=88&btnCommander.y=125&btnCommander=commander`

3.

```
<button name="Commander"
        value="Commander">
    <img src="btnCommander.png" alt="Commander">
</button>
```

L'élément `<button>` ajouté par html5 est de loin la meilleure de ces trois techniques. Cet élément est très polyvalent car il peut être utilisé autant dans un formulaire qu'à l'extérieur pour déclencher un autre type d'action.

S'il est utilisé dans un formulaire, par défaut son `type="submit"`.
Si son rôle est de déclencher une action autre que la soumission du formulaire, il faut définir `type="button"`.

Au besoin, on peut placer à l'intérieur d'un `button` une balise image et le `button` est plus facile à styler que des `input`.

Touche Enter

Lorsque le focus est dans un formulaire et qu'on frappe la touche *Enter*, l'évènement submit du formulaire est automatiquement appelé et le formulaire est envoyé (en post ou get selon la méthode).

En accessibilité, il faut s'assurer que ce mécanisme est fonctionnel.

Balisage sémantique et accessible des formulaires

Placer les éléments en ligne dans des conteneurs blocs.

Le Doctype (.dtd) strict de HTML 4.01 spécifie l'entité de la balise **`form`** ainsi:

```
<! ELEMENT FORM (%block;|SCRIPT)+  -- >
```

Ce qu'on peut traduire par: « La balise **`form`** doit avoir comme enfants des éléments de type bloc ou une balise script. »

```
NON : <form> <label /> <input /> </form>
```

```
OUI : <form> <p> <label /> <input /> </p> </form>
```

Ainsi, les contenus en ligne d'un formulaire doivent être placés dans des conteneurs de type bloc comme un **`<p>`**.

Choisir le bon input

Options : https://www.w3.org/TR/html51/sec-forms.html#the-input-element

```
input type = text
             password
             checkbox
             radio
             button
             submit
             hidden
             image
             date
             time
             number
             range
             email
             url
             search
             tel
             color
             …
```

Le clavier des appareils mobiles s'adapte selon le type de input.

Principaux attributs des inputs.

`type`	
`name`	essentiel pour l'envoi des données
`id`	essentiel pour l'étiquetage
`value`	
`placeholder`	pour proposer un exemple. *À éviter…* selon les experts UX, il est préférable de placer l'information de manière **stable** près du champ de saisie.
`autocomplete`	valeurs possibles : "on" ou "off"
`required`	champ obligatoire
`max`	valeur maximum

`maxlength`	nombre de caractères maximum
`min`	valeur minimum
`minlength`	nombre de caractères minimum
`pattern`	un motif… par exemple, le code postal : A1A 1A1. Voir http://html5pattern.com/
`title`	DOIT être ajouté lorsqu'on utilise l'attribut pattern, pour documenter le motif utilisé
`checked`	
`disabled`	
`readonly`	

`formnovalidate`	pour empêcher la validation html5 du formulaire, ajouter cet attribut au bouton de soumission du formulaire

```
<button name="Commander" formnovalidate>
    Commander
</button>
```

Références

Avant d'utiliser l'un ou l'autre des attributs des input, vous pouvez vérifier si cet attribut s'applique :

https://www.w3.org/TR/html51/sec-forms.html#the-input-element

Pour vérifier le support des navigateurs pour les éléments, les attributs et les valeurs d'attributs html5 des formulaires:

https://www.wufoo.com/html5

Regrouper et étiqueter

Faire des regroupements thématiques

- **Avec un `fieldset`**

  Lorsqu'un groupe d'éléments de formulaire correspond à un sous-ensemble thématique, on doit les placer dans une balise `fieldset`.

- **Avec `optgroup`**

  Dans une liste déroulante (balise `<select>`), un groupe d'options se fait en englobant un ensemble de balises `<option>` dans un élément parent `<optgroup>`

Étiqueter les regroupements

- Un **`fieldset`** est étiqueté par une balise **`legend`** qui doit **obligatoirement être le premier enfant** du **`fieldset`**.

- La balise **`legend`** peut contenir un titre **`h1-h6`**.

- On peut aussi ajouter une étiquette à un groupe d'options d'un **`select`** (balise **`<optgroup>`**) en ajoutant un attribut label à la balise **`<optgroup>`**.

  Exemple :

```
<optgroup label="Canada">
    <option value="AB">Alberta</option>
    <option value="BC">British Columbia</option>
    <option value="QC">Quebec</option>
</optgroup>
```

Étiqueter les éléments de formulaire

- **Avec une balise <label>**

Mauvais exemple :

```
<label>Prénom
    <input type="text" name="prenom" />
</label>
```

Il faut faire un lien explicite entre le `label` et le `input` qu'il documente en utilisant dans le label un attribut `for` ayant comme valeur, l'identifiant du `input`.

Bon exemple :

```
<label for="prenom">Prénom</label>
<input type="text" name="prenom" id="prenom" />
```

Que faire si le visuel impose de cacher l'étiquette ?

Exemple :

- **Avec un label visuellement caché**

```
<label for="recherche" class="screen-reader-only">
   Entrer un terme de recherche </label>
<input    type="search" id="recherche" />
<input    type="submit" value="Rechercher" />
```

- **OU mieux : avec un attribut aria-label**

```
<input    type="search" id="recherche"
          aria-label="Entrer un terme de recherche" />
<input    type="submit" value="Rechercher" />
```

Le cas des boutons radios et des cases à cocher

Dans le cas des boutons radios et des cases à cocher,
la balise **`label`** est préférablement placée **après** la balise **`input`**.

Il faut savoir qu'un groupe de boutons radio partage le même **`name`**.

Le **`name`** d'un groupe de bouton radio est **l'identifiant du groupe.**

C'est parce qu'il partage le même **`name`** que le comportement de sélection (**`checked`**) fonctionne en exclusion mutuelle.

Pour valider un groupe de boutons radio, on se servira du **`name`** du groupe plutôt que des **`id`** individuel.

Pour étiqueter un groupe de boutons radio, on utilisera une balise **`fieldset`** et sa **`legend`**.

```
<fieldset>
   <legend> Quelle est votre boisson chaude préférée?</legend>
   <ul>
      <li>
         <input   name="boisson" id="cafe"
                  type="radio" value="cafe" />
         <label for="cafe">Le café</label>
      </li>
      <li>
         <input   name="boisson" id="chocolat"
                  type="radio" value="chocolat" />
         <label for="chocolat">Le chocolat chaud</label></p>
      </li>
      <li>
         <input   name="boisson" id="capuccino"
                  type="radio" value="capuccino" />
         <label for="capuccino">Le capuccino</label>
      </li>
   </ul>
</fieldset>
```

Lorsqu'il y a seulement 2 options et que les étiquettes de chaque bouton radio rendent évident la nature du groupe, par exemple : Madame / Monsieur, les balises **`fieldset`** et **`legend`** sont inutiles.

Naviguer au clavier

S'assurer que le formulaire est **navigable logiquement à l'aide du clavier**

Tab pour avancer
Majuscule + Tab pour revenir en arrière

Flèches gauche et droite pour passer d'un bouton radio à l'autre

Espace pour cocher une case à cocher
ou un bouton radio
ou sélectionner une option d'un select

Enter lorsque le focus est dans un select, la touche Enter permet de sélectionner une option

Enter s'assurer que le formulaire peut-être soumis avec la touche "Enter"

Naviguer au clavier (suite)

Si le formulaire est navigable logiquement à l'aide du clavier,

l'attribut **`tabindex`** NE doit PAS être utilisé.

C'est seulement dans le cas où l'ordre de tabulation naturel (celui du code HTML) ne suffit pas que l'on peut recourir aux valeurs suivantes :

- **forcer l'ajout d'un élément dans l'ordre de tabulation**

  ```
  tabindex="0"
  ```

- **sortir un élément de l'ordre naturel de tabulation**

  ```
  tabindex="-1"
  ```

Fournir des textes alternatifs aux images fonctionnelles

Lorsqu'un élément de formulaire contient une image et que celle-ci donne son sens à l'élément de formulaire, il est indispensable de fournir un équivalent textuel.

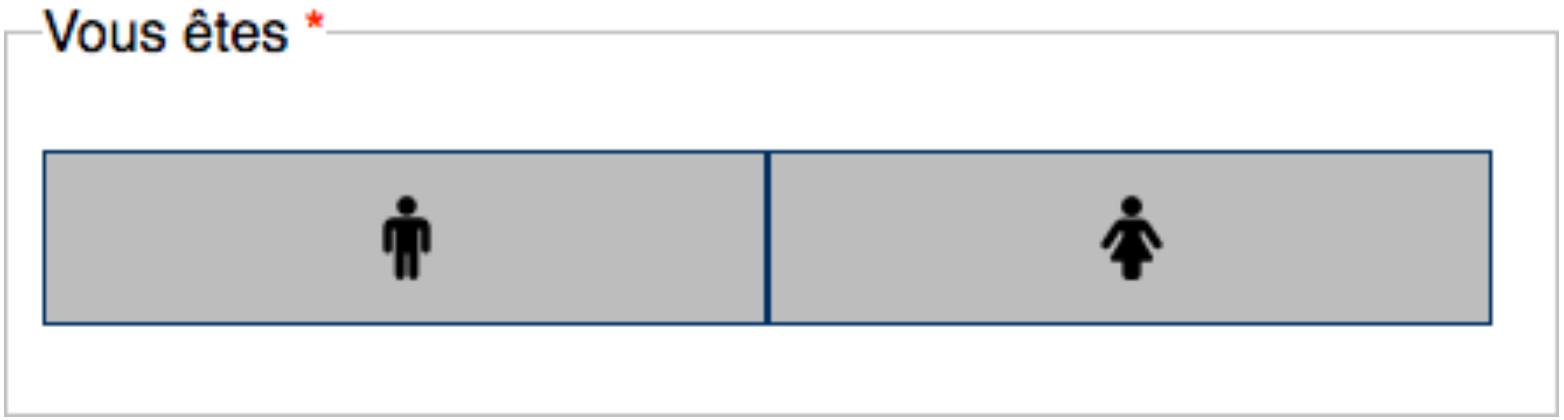
Exemple 1. Bouton de soumission d'une zone de recherche :

```
<button>

    <img    src="images/search.png"
            width="74" height="29"
            alt="Rechercher" />

</button>
```

Exemple 2. Boutons radio :




```
<fieldset id="genre">
    <legend> Vous êtes </legend>
    <p class="ctnFlex">
        <input type="radio" name="sexe" id="Mme" value="mme"
               class="screen-reader-only"
               required>
        <label for="Mme">Madame</label>
        <input type="radio" name="sexe" id="M" value="m"
               class="screen-reader-only">
        <label for="M">Monsieur</label>
    </p>
</fieldset>
```

Comment fournir une aide efficace

Parfois, les données à saisir par l'usager doivent respecter un motif particulier, ou encore, certains caractères sont permis et d'autres interdits et il faut renseigner l'utilisateur à ce sujet.

Une manière efficace de fournir ces renseignements est de les joindre à l'étiquette de l'élément de formulaire. Par exemple :