Université Lille1 Sciences et Technologies

Séances 5a/5b - Technologies du Web

FORMATIONS EN INFORMATIQUE DE LILLE 1

février 2013

javascript

Exercice 1 : En XHTML, la balise `blockquote` est utilisée pour les citations longues, en particulier quand elles comportent plusieurs paragraphes et des retours à la ligne. L'attribut `cite` sert à donner des informations sur l'origine de la citation, mais n'est pas utilisé par la plupart des navigateurs. Voici un exemple d'utilisation de blockquote :

```
<blockquote cite="http://www.w3.org/DOM/">
    <p> "Dynamic HTML" is a term used by some vendors to describe the combination of HTML,
    style sheets and scripts that allows documents to be animated.</p>
</blockquote>
```

Question 1.1 : Écrire un script javascript qui permet de rendre "visible" la référence qui apparaît dans l'attribut `cite` en ajoutant un paragraphe (élément `p`) à l'élément blockquote, à l'intérieur duquel figure un lien vers la page citée.
Sur l'exemple précédent, on obtient :

```
<blockquote cite="http://www.w3.org/DOM/">
    <p> "Dynamic HTML" is a term used by some vendors to describe the combination of HTML,
    style sheets and scripts that allows documents to be animated.</p>
    <p><a href="http://www.w3.org/DOM/">Source</a></p>
</blockquote>
```

Exercice 2 : Soit un document contenant un menu permettant de se diriger vers les différentes parties du document :

```
<body>
 <h1>Titre principal</h1>
 <nav>
   <ul id="menu">
     <li>
       <a href="#section1">vers section 1</a>
     </li>
     <li>
       <a href="#section2">vers section 2</a>
     </li>
     ...
   </ul>
 </nav>
 <section id="section1">
 ...
 </section>
 <section id="section2">
 ...
 </section>
 ...
</body>
```

Question 2.1 : La structure du menu est similaire à celle du document abordé dans la fiche d'exercices CSS (« aide mémoire ... ») Dans cet exercice, on avait donné en CSS un aspect de bouton à chacun des items de la liste. Une différence persistait cependant avec les boutons «classiques» : seul le texte du lien était cliquable mais pas le rectangle autour de ce lien.

- Écrire une fonction javascript qui rend cliquable chaque item du menu. L'action à déclencher sera de charger la page dont l'URL est dans l'attribut `href` de l'élément `a` correspondant.
  (NB : Pour forcer le navigateur à charger un nouvelle page, vous pouvez utiliser l'instruction `window.location.assign(urlACharger)`)
- Dans un fichier `itemCliquables.js` (à lier au fichier HTML) vous mettrez cette fonction ainsi que tout code javascript nécessaire au lancement de cette action au chargement de la page.

Question 2.2 : *Cette question est indépendante de la précédente.*
On souhaite faire en sorte que **seule** la section que désire consulter l'utilisateur soit affichée. Quand l'utilisateur clique sur un lien du menu, la section qu'il vient de choisir s'affiche et celle qu'il consultait précédemment (le cas échéant) se masque.
Initialement toutes les sections sont masquées (seul le menu apparaît).
Écrire un programme javascript qui met en œuvre ce comportement. Vous le développerez dans un fichier `afficherCacher.js`, à lier au fichier HTML.

Exercice 3 : Dans l'exercice précédent, chaque section se trouve dans un élément `section`. Malheureusement, il se produit souvent que les documents HTML ne soient pas aussi bien structurés et que les éléments `section` soient absents. Seuls la présence des titres `h2` permet alors de distinguer les différentes parties. (voir exemple ci-dessous)
La structuration du source HTML est ici insuffisante et il est difficile d'isoler une section complète dans une règle CSS (pour lui donner une couleur de fond particulière, l'encadrer ou la masquer, par exemple). La bonne solution consisterait à mieux concevoir le code HTML mais cela n'est pas toujours possible (si l'on ne maîtrise pas la manière dont le HTML est produit, par exemple).
Question 3.1 : Écrire un programme javascript qui permet d'ajouter dans l'arbre DOM les éléments `section` encadrant chaque section (chaque partie commençant par un `h2`) de façon à obtenir une structure de document similaire à celle de l'exercice précédent.
Le programme modifiera les `id` des éléments : l'identifiant du `h2` sera reporté sur l'élément `section` auquel il appartient.
En résumé, il s'agit de constituer un arbre DOM équivalent au HTML suivant :

Source HTML

```
<body>
  <h1>Titre Principal</h1>
  ...   un menu ..
  <h2 id="section1">Titre section 1</h2>
  ... contenu de la section 1
  <h2 id="section2">Titre section 2</h2>
  ... contenu de la section 2
  ...
  <h2 id="sectionN">Titre section N</h2>
  ... contenu de la section N
 </body>
```

Structure à construire en JS

```
<body>
  <h1>Titre Principal</h1>
  ...   un menu ..
  <section id="section1">
     <h2 >Titre section 1</h2>
    ... contenu de la section 1
  </section>
  <section id="section2">
    <h2>Titre section 2</h2>
    ... contenu de la section 2
  </section>
   ...
```