Cours CSS/JavaScript

E.Coquery

emmanuel.coquery@liris.cnrs.fr

CSS

Cascading Style Sheets

- Principe : séparation du style et de la structure
- HTML → Structure
  - paragraphes, tableaux, liens, ...
- CSS → Style
  - polices de caractères, couleurs, position, bordures ...

CSS : syntaxe

Fichier CSS indiqué par :
`<link rel="stylesheet" type="text/css" href="fichier.css">` dans `<head>...</head>`

```
sélecteur {
   nom-propriété : valeur;
   nom-propriété2 : valeur2;
   ...
}

sélecteur2 {
...
}
```

CSS : dans le HTML

Possibilité d'inclure un style particulier dans une balise :

- Attribut `style`
- Valeur : chaîne de caractères contenant les propriétés.
  - Ce qu'il y a dans les { }

Exemple :

```
<p style="color : #00FF00 ; font-style : italic ;"
>...</p>
```

Sélecteurs simples

- nom de tags : *tag*
  - li p div
  - correspond à la balise indiquée
- nom de classes : *.classe*
  - .titre .article .vert
  - correspond aux balises dont l'attribut `class` est *classe*
  - par exemple `<div class="titre">`
  - si la classe est collée à un tag, ce sélecteur ne fonctionne que pour ce tag.
    - `div.titre` fonctionne pour `<div class="titre">` mais pas pour `<p class="titre">`
- nom d'objet : *#nom*
  - #menu #logo
  - correspond aux balises dont l'attribut `id` est *nom*
  - au maximum une balise pour un nom donné

Sélecteurs - 2

- Possibilité de partager des propriétés entre sélecteurs :
  - `selecteur1, selecteur2 { ...`
  - les propriétés vont s'appliquer sur les balises correspondant à `selecteur1` ou à `selecteur2`
- Possibilité de cumuler des sélecteurs en les séparant avec des espaces :
  - `selecteur1 selecteur2 { ...`
  - les propriétés vont s'appliquer sur les balises correspondant à `selecteur2` qui sont dans des balises correspondant à `selecteur1`
  - exemple :
    `div a { ...` s'applique sur la balise a dans :
    `<div>...<p>...<a>...</a>...</p>...</div>`

Pseudo-classes

Correspondent à un état d'un lien :

- `a :link {...`
  - lien non visité
- `a :visited {...`
  - lien visité
- `a :hover {...`
  - activé lors du passage de la souris sur le lien
  - a mettre après `link` et `visited` dans le fichier css
- `a :active {...`
  - activé lors du clic
  - a mettre après `hover` dans le fichier css

Autres pseudo-classes : first-child, lang, focus

- Non limités aux liens

Cascades

- Une balise se voit appliquer les propriétés de tous les sélecteurs qui lui correspondent
- Une balise hérite les propriétés de son contenant, sauf quelques exception
- En cas de conflit conteneur/contenu sur une propriété, c'est le contenu qui gagne
- En cas de conflit entre deux sélecteurs, on applique la priorité dans l'ordre (lexicographique) suivant :
  - celui ayant le plus de noms (#)
  - celui ayant le plus de classes
  - celui ayant le plus de tags
  - si un des deux voit sa valeur suivit de `!important`

Polices

- `font-size` taille de la police
  - valeurs : small, medium, large, smaller, larger, *xx*%
- `font-style` : normal, italic, oblique
- `font-variant` :normal, small-caps
- `font-weight` : normal, bold, bolder, lighter, ...
- `font-family` : times, helvetica, ...

Texte

- `color` : #AABB22 (RVB en hexadécimal), Blue, LightBlue, ...
- `text-align` : left, right, center, justify
- `text-decoration` : none, underline, overline, line-through, blink
- ...

Fonds

- `background-attachment` : scroll, fixed
- `background-color` : couleur
- `background-image` : url(URL),none
- `background-position` : top left, top center, top right, center left, bottom right, *x*% *y*%, *xpos ypos*
- `background-repeat` : repeat, repeat-x, repeat-y, no-repeat

Tailles

Hauteur

- `height` : auto, taille
- `line-height` : normal, taille
- `max-height` : none, taille
- `min-height` : taille

Idem pour la largeur (`width`)

Unités : `em`, `ex`, `%`, ...

Divers

- `float` : left, right, none
- `position` : static, relative, absolute, fixed
- `visibility` : visible, hidden, collapse
- `vertical-align` : top, middle, bottom, x%, ...

JavaScript

- Langage pour programmer des comportement dynamiques côté client
- Permet la manipulation du HTML à travers sa représentation DOM (Document Object Model)

Exemple DOM

Le langage JavaScript

Dans l'idée proche du PHP, avec les différences suivantes :

- Variables : *nomVar*
- Concaténation : +
- Pas de variables dans les chaînes de caractères
- Création de tableaux : `tab = new Array()`
- Tailles des tableaux : `tab.length`

Intégration à une page :

```
<script src="monfichier.js" type="text/javascript">
</script>
<script type="text/javascript">mon script</script>
```

ou dans des attribut de gestion d'événement (ex : onclick)

Le langage JavaScript - suite

Langage à objets :

- var.champ
  - accès à un champ d'un objet
  - `crayon.couleur = "rouge"`
- var.func(args)
  - utilisation d'une méthode d'un objet
  - `crayon.taille()`

Objets node

- Correspond à un noeud (balise, texte, attribut) dans l'arbre DOM
- Ensemble de champs et de méthodes permettant de les manipuler
- `noeud = document.getElementById("menu")`
  - met dans `noeud` un objet `node` correspondant à la balise nommée `menu`
- `noeud.className = ...`
  - Change la classe de la balise
- `noeud.style.ppte = ...`
  - Change la valeur d'une propriété CSS du noeud (la valeur doit être une chaîne de caractères).
  - `ppte` est le nom de la propriété CSS où les - sont supprimés et où les lettres suivant des - sont mises en majuscule (ex : `backgroud-color` → `backgroundColor`)

Fonctions utiles

- alert(message) crée une fenêtre de dialogue dans laquelle le message est affiché
- back() permet de retourner à la dernière page visitée
- close(nom_de_la_fenetre) détruit une fenêtre du client
- confirm(message) crée une fenêtre de dialogue pour confirmer une action : elle permet le choix entre OK et Annuler
- open(URL,nom_de_la_fenetre,options_de_la_fenetre) crée une nouvelle fenêtre client
- prompt(message,valeur_defaut) crée une fenêtre de dialogue permettant la saisie et dans laquelle le message est affiché

Événements

- Attributs (donc à mettre dans le HTML) associés à des événements
  - clic de souris, sélection d'un composant de saisie
- Valeur des attributs = code JavaScript à exécuter si l'événement arrive
  - En général, un appel de fonction

Exemple :

```
<script type="text/javascript">
   function toto() {
      document.getElementById("toto")
         .style.backgroundColor="#0000FF";
   }
</script> ...
<p id="toto" onmouseover="toto()">blabla</p>
```

Quelques évémenements

- `onclick, ondblclick`
  - Lors d'un (double) clic sur la zone d'affichage de la balise.
  - Clic annulé si la fonction renvoie faux
- `onmouseover, onmouseout`
  - Lorsque la souris passe sur, ou bien sort de, la zone d'affichage de la balise.
- `onfocus, onblur`
  - Lorsqu'un composant de saisie obtient/perd le *focus*
- `onchange`
  - Après un changement de valeur pour un coposant de saisie
- `onsubmit, onreset`
  - Lors de l'envoi/de la remise à zéro d'un formulaire
  - Envoi annulé si la fonction renvoie faux

Références

CSS :
`http://www.w3schools.com/css/css_reference.asp`

JavaScript :
`http://www.w3schools.com/js/default.asp`

Evénements :
`http://www.w3.org/TR/html4/interact/scripts.html#h-18.2.3`