Technologie Web

HTML5 et CSS

Alexandre Pauchet

INSA Rouen - Département ASI

BO.B.RC.18, pauchet@insa-rouen.fr

Plan

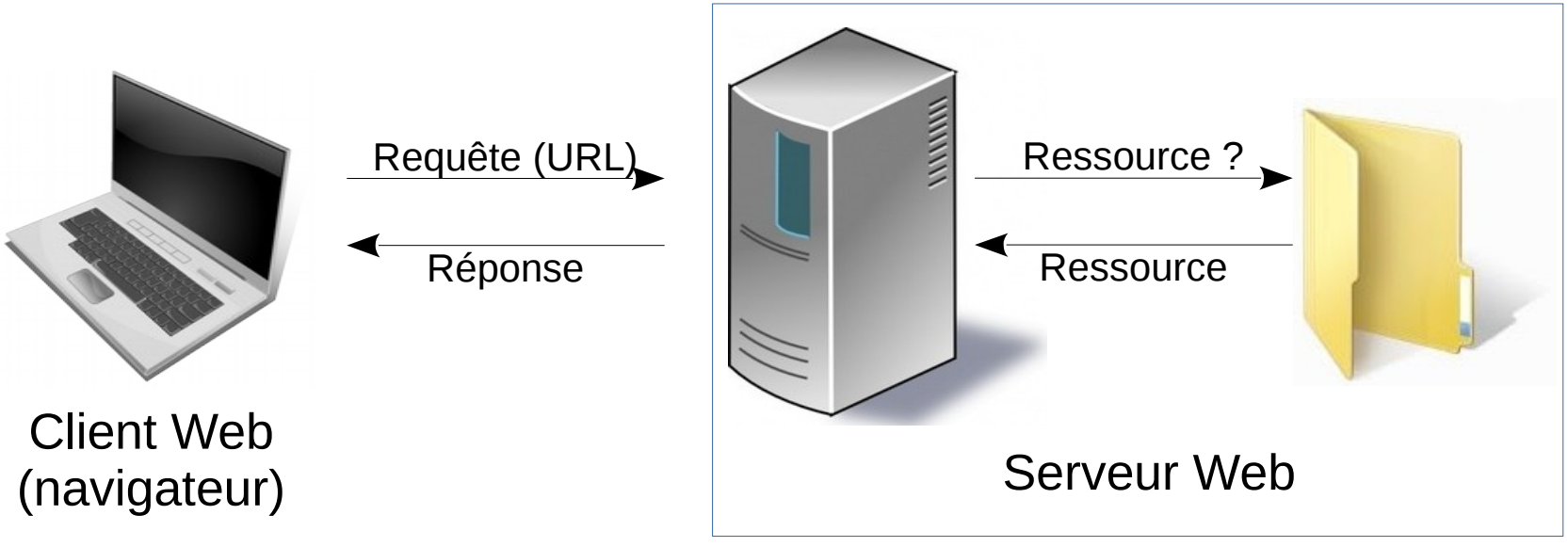
Introduction (1/6)
Historique

- Années 1990 : HTML est créé par Tim Berner-Lee au Centre Européen de Recherche Nucléaire (CERN)
- 1995 : HTML 2.0 normalisation par l'IETF [1]
- 1996 : HTML 3.2 ajout des tables, des applets (Java), *etc.*
- 1998 : HTML 4.01 ajout des feuilles de styles, des frames, *etc.*
- 2000 : XHTML 1.0 reformulation de HTML 4 en XML 1.0
- 2002-2006 : XHTML 2.0 en cours de spécification
- 2007-maintenant : HTML5

Normalisation par le W3C [2] depuis 1996.

1. Internet Engineering Task Force
2. World Wide Web Consortium `http://www.w3c.org`

Introduction (2/6)
Principe de fonctionnement



1. Le navigateur effectue une requête spécifiée à travers l'URL
2. Le serveur retourne un flot typé de données
3. Le navigateur interprète le flot de données et l'affiche

Introduction (3/6)
Langages à balises

- Un fichier HTML/XHTML est un fichier **texte** (*cf.* protocole HTTP) contenant des **balises** appelant des commandes, dont l'action est limitée au texte contenu entre la **balise de début** et la **balise de fin**.
- Extension HTML : `.htm` ou `.html` ; XHTML : `.xhtml`
- Balise de début : `<commande>`
- Balise de fin : `</commande>`
- Balise auto-fermante : `<commande/>`
- Commentaires : `<!--Ceci est un commentaire-->`

Remarque : retours chariot, succession d'espaces et/ou de tabulations ne sont pas pris en compte.

Introduction (4/6)
Squelette d'un document XHTML

XHTML (Hello.xhtml)

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
      strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head>
   <title>Page XHTML Type</title>
   <meta http-equiv="content-type" content="text/html;charset=utf-8" />
   <meta http-equiv="Content-Style-Type" content="text/css" />
</head>
<body>
   <p>Hello world!</p>
</body>
</html>
```

Introduction (5/6)
Squelette d'un document HTML 5

HTML 5 (Hello.html)

```
<!DOCTYPE html>
<html>
   <head>
      <title>Page HTML 5 Type</title>
      <meta http-equiv="content-type" content="text/html; charset=utf-8" />
   </head>
   <body>
      <p>Hello world!</p>
   </body>
</html>
```

Introduction (6/6)
Pourquoi HTML 5 ?

- **Objectifs**
  - Réduire les besoins en plugins externes (ex : Flash)
  - Mieux gérer les erreurs
  - Davantage de contrôle sans utilisation de Javascript
  - Indépendance du terminal (PC, mobile, etc.)

- **Mise en œuvre**
  - Élément `<canvas>` pour les dessins 2D
  - Les éléments `<video>` et `<audio>` pour le multimédia
  - Gestion d'un stockage local
  - Nouvelles balises : `<section>`, `<article>`, `<header>`, ...
  - Contrôle pour les formulaires : calendar, date, time, email, url, search

Langage à balise (1/5)
3 types d'éléments

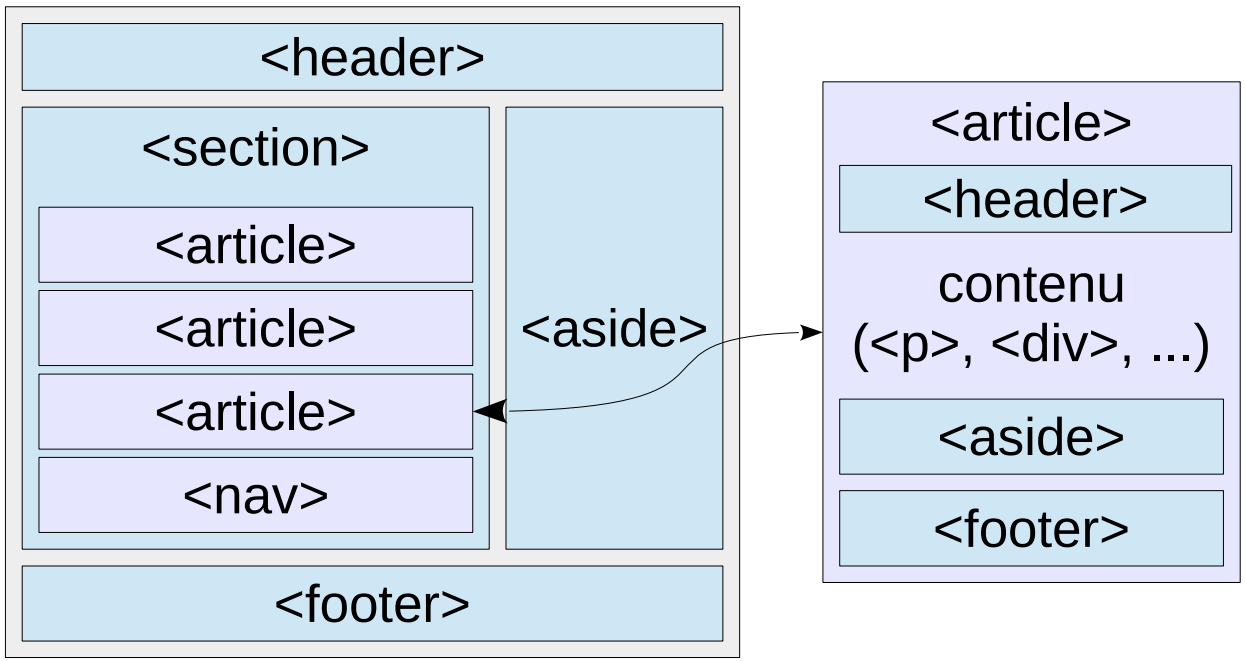
- **Élément bloc (div) :** élément précédé et suivi d'un saut de ligne. Il forme une boite dans lequel est inclus du texte ou d'autres éléments.
  Exemples : les paragraphes, les tableaux, ...
- **Élément inline (span) :** élément qui s'insère dans le fil du texte et ne peut contenir que du texte ou d'autres éléments inlines.
  Exemples : les éléments typographiques italique, gras, ...
- **Élément auto-fermant :** élément qui est une balise ouvrante et fermante à la fois. Elle n'a donc pas de contenu. Ce sont soit des balises de type bloc, soit de type inline.
  Synthaxe : `<balise>` ou `<balise/>`
  Exemples : saut de ligne, séparation horizontale ...

Langage à balise (2/5)
Les balises

- Apporte du sens
  - `<title></title>` : titre de la page
  - `<h1></h1>` : grand titre
  - `<h2></h3>` : titre de second niveau
  - `<p></p>` : paragraphe
  - `<code></code>` : portion de code informatique

- Mise en forme
  - `<br />` : génère un saut de ligne
  - Structuration d'un document : `<section>`, `<article>`, `<header>`, `<footer>`, `<aside>`, `<nav>`

Langage à balise (3/5)
Exemple de document structuré



Langage à balise (4/5)
Notion de flux

Le flux HTML :

- Les balises sont lues séquentiellement, ...
- ... sont affichées au fur et à mesure par le navigateur, ...
- ... les unes en dessous des autres

Remarque

L'affichage se modifie au fur et à mesure du chargement de la page et de ses composants (texte, images, etc.)

Langage à balise (5/5)
Attributs

Il est possible de transmettre des informations à traiter aux balises :

- Balise de début :
  ```
  <balise[ attribut1=valeur1[ attribut2=valeur2 ...]]>
  ...
  </balise>
  ```
- Balise auto-fermante :
  ```
  <balise[ attribut1=valeur1[ attribut2=valeur2 ...]]/>
  ```

Exemples

```
<code langage="java">System.out.println("Alerte");</code>
<br class="double"/>
```

Éléments HTML 5 (1/10)

Titres (éléments bloc)

Il y a 6 niveaux de titre :

- `<h1> ... </h1>`
- `<h2> ... </h2>`
- `<h3> ... </h3>`
- ...
- `<h6> ... </h6>`

Éléments HTML 5 (2/10)
Paragraphes (éléments bloc)

- l'élément de bloc ... permet de construire des paragraphes et par un attribut `align` spécifie la justification.
- l'élément inline <br/> permet de contrôler les sauts de lignes.

Exemple

```
<!DOCTYPE html>
<html>
   <head>
      <title>Paragraphe</title>
      <meta http-equiv="content-type" content="text/html;charset=utf-8" />
      <style type="text/css">
         p {text-align: justify}
      </style>
   </head>
   <body>
      <p>paragraphe paragraphe paragraphe paragraphe
         paragraphe paragraphe paragraphe paragraphe <br/> paragraphe
         paragraphe paragraphe paragraphe paragraphe paragraphe</p>
   </body>
</html>
```

Éléments HTML 5 (3/10)

Texte structuré (éléments inline)

em : mise en exergue

strong : mise en exergue plus importante

cite : extrait ou référence à une autre source

code : portion de code informatique

samp : exemple de résultat issu d'un programme

kbd : frappe au clavier devant être effectuée par l'utilisateur

var : instance d'une variable ou le paramètre d'un programme

dfn : terme encadré a une définition

abbr : forme abrégée

...

Éléments HTML 5 (4/10)

Listes (éléments bloc)

Chaque item d'une liste est déclaré par `<li> ... </li>`.

Liste simple :

```
<ul>
  <li> item </li>
  <li> item </li>
  <li> item </li>
</ul>
```

Liste numérotée :

```
<ol>
  <li> item </li>
  <li> item </li>
  <li> item </li>
</ol>
```

Il existe aussi des listes de définitions (`<dl> ... </dl>`), les items sont déclarés par les balises `<dt>` pour le terme et `<dd>` pour la définition associée.

Éléments HTML 5 (5/10)

Exemples de liste

Exemple

```
<!DOCTYPE html>
<html>
   <head>
      <title>Listes</title>
      <meta http-equiv="content-type" content="text/html;charset=utf-8" />
   </head>
   <body>
    <h5>Liste 1</h5>
    <ul>
      <li>item</li>
      <li>item</li>
    </ul>
    <h5>Liste 2</h5>
    <ol>
      <li>item</li>
      <li>item</li>
    </ol>
    <h5>Liste 3</h5>
    <dl>
      <dt>item</dt><dd>définition</dd>
      <dt>item</dt><dd>définition</dd>
    </dl>
   </body>
</html>
```

Éléments HTML 5 (6/10)

Tableaux (éléments bloc) : squelette général

```
<table>
  <caption>
    <!-- titre du tableau -->
  </caption>
  <thead>
    <!-- entete de table -->
  </thead>
  <tfoot>
    <!-- pied de table -->
  </tfoot>
  <tbody>
    <!-- corps de la table -->
  </tbody>
</table>
```

- Les éléments `<thead>` et `<tfoot>` permettent de répéter l'élément dans les tableaux sur plusieurs pages (impression).
- La balise `<tr>` déclare une ligne
- Les balises `<td>` (cellule normale) ou `<th>` (cellule titre/grasse) déclarent les cellule dans la ligne
- Les attributs `rowspan` et `colspan` fusionnent les cellules

Éléments HTML 5 (7/10)

Exemple de tableau

Exemple

```
<table border="1">
   <caption>
      <h2>Titre du tableau</h2>
   </caption>
   <thead>
      <tr>
         <th>Titre 1</th><th>Titre 2</th><th>Titre 3</th>
      </tr>
   </thead>
   <tfoot>
      <tr>
         <th>Titre 1</th><th>Titre 2</th><th>Titre 3</th>
      </tr>
   </tfoot>
   <tbody>
      <tr>
         <td>cellule 1</td><td>cellule 2</td><td>cellule 3</td>
      </tr>
      <tr>
         <td>cellule 4</td>
         <td>cellule 5</td>
         <td rowspan="2">cellule 6</td>
      </tr>
      <tr>
         <td colspan="2">cellule 7</td>
      </tr>
```

Éléments HTML 5 (8/10)

Images (éléments inline)

La balise `<img/>` permet d'insérer une image

Les attributs suivants sont obligatoires :

- `src` : l'URI où se situe l'image
- `alt` : courte description de l'image

Exemple

```
<img src="debian.png"alt="le logo Debian"/>
```

Remarque : en spécifiant la taille des images, on accélère le chargement (attributs `width` et `heigth`).

Éléments HTML 5 (9/10)

Liens (éléments inline)

- L'élément `<a href="...">...</a>` permet d'insérer un lien
- Le contenu de l'élément est celui qui sera affiché en tant que lien.
- L'attribut `href` contient l'URI vers laquelle le lien pointe.
  - URL : `http://www.google.com`
  - URL(mail) : `mailto:alexandre.pauchet@insa-rouen.fr`
  - Fichier local avec chemin relatif : `./dossier/autre_page.html`
  - Fichier local avec chemin absolu : `/www/dossier/autre_page.html`

Exemple

```
<a href="lienversuneautrepage.html">Texte affiché</a>
```

Éléments HTML 5 (10/10)
Encodage des caractères

- Les anciennes versions d'HTML nécessitent l'utilisation d'entités :

é	`\&eacute;`	è	`\&egrave;`	ê	`\&ecirc;`
à	`\&agrave;`	É	`\&Eacute;`	...	...

- Maintenant l'encodage est supporté. Bonne pratique : `UTF-8`
- L'encodage doit être défini à plusieurs endroits :
  - HTML : `<meta http-equiv="content-type"content="text/html;charset=utf-8"/>`
  - HTTP : `Content-Type: text/html; charset=utf-8`

Les formulaires (1/9)
Déclaration d'un formulaire

L'élément `<form> ... </form>` déclare un formulaire

Les attributs :

- `action` : URL spécifiant le traitement des données (CGI, *etc.*)
- `method` : spécifie la méthode d'acheminement des données (`GET` par défaut ou `POST`)
- `enctype` spécifie la méthode d'encodage pour un envoi en `POST`
  - `application/x-www-form-urlencoded` : encodage par défaut
  - `multipart/form-data` : aucun encodage, (utilisé notamment pour le file-upload)
  - `text/plain` : seul les espaces sont remplacés par des '+'

Les formulaires (2/9)
Contenu d'un formulaire

L'élément `<fieldset>...</fieldset>` permet de définir un regroupement dans un formulaire.

L'élément `<legend>...</legend>` permet de donner une légende à un fieldset.

L'élément `<label>...</label>` permet de définir une étiquette.

L'élément `<input/>` contient les attributs suivant :

- `type` : spécifie le type d'élément à utiliser
- `name` : donne un nom à l'élément
- `value` : donne une valeur à l'élément

Les formulaires (3/9)
Exemple de balises <input/>

```
<input type="text" name="champs" size="10" value="texte"/>
<input type="password" name="mdp" size="10" maxlength="8"/>
<input type="hidden" name="steak" value="haché"/>

<input type="checkbox" name="chk1" value="ok"/>
<input type="checkbox" name="chk2" value="ok" checked="checked"/>

<input type="radio" name="choix" value="rd1"/>
<input type="radio" name="choix" value="rd2" checked="checked"/>
<input type="radio" name="choix" value="rd3"/>
```

Remarques :

- pour les types checkbox/radio si l'attribut value n'est pas spécifié, la valeur par défaut est on
- le type hidden permet de passer des valeurs d'une page à une autre

Les formulaires (4/9)
Boutons

Un élément `<input/>` de type :

- `submit` : affiche un bouton et permet l'envoi des données du formulaire au serveur
- `image` : affiche une image et permet l'envoi des données du formulaire au serveur
- `reset` : affiche un bouton et permet de restaurer les valeurs par défaut du formulaire
- `file` : affiche un bouton permettant d'ouvrir une boite de recherche de fichier

Exemple

```
<input type="submit" name="action" value="Insert"/>
<input type="file" name="unFichier" id="fichier" />
<input type="image" src="images/croix.jpg" name="action" value="Delete"/>
<input type="reset" value="Reset"/>
```

Les formulaires (5/9)

Champs texte

Un élément `<textarea>` permet de créer un champs texte

Exemple

```
<textarea rows="4" cols="50">
Ce texte est éditable et sera envoyé lors du submit
</textarea>
```

Les formulaires (6/9)
Listes

La balise `<select>` permet de définir une liste

Attributs :

- `multiple` : permet de sélectionner plusieurs éléments de la liste
- `code` : si $>$ 2 affiche un tableau, sinon un menu déroulant

Exemple

```
<select name="laliste" size="3" multiple="multiple">
  <option value="1">toto</option>
  <option selected="selected" value="2">titi</option>
  <optgroup label="les autres">
    <option value="3">tata</option>
    <option value="4">tutu</option>
    <option value="5">tete</option>
  </optgroup>
</select>
```

Les formulaires (7/9)
Nom des champs

La balise `<label>` sert à nommer des champs

- Attribut `for` indique champs décrit (attribut `id`)
- Utile sur les radio et checkbox : augmente la surface d'activation

Exemple

```
<form>
  <label for="h">Homme</label>
  <input type="radio" name="genre" id="h" />
  <br />
  <label for="f">Femme</label>
  <input type="radio" name="genre" id="f" />
</form>
```

Les formulaires (8/9)

Exemple de formulaire

Formulaire.html

```
<form action="GET">
   <fieldset>
      <legend>Exemple de formulaire</legend>
      <label>Nom :</label> <input type="text" name="monNom" id="nom" />
      <label>Prénom :</label> <input type="text" name="monPrenom" id="prenom" />
      <hr/>
      <input type="checkbox" name="maNewsletter" id="newsletter" /> <label for="newsletter">Une
            checkbox</label>
      <input type="radio" name="monSexe" id="homme" /><label for="homme">Homme</label>
      <input type="radio" name="monSexe" id="femme" /><label for="femme">Femme</label><br/>
      <label for="photo">Fichier :</label> <input type="file" name="maPhoto" id="photo" /><br/>
      <select name="laliste" size="3" multiple="multiple">
      <option value="1">toto</option>
      <option selected="selected" value="2">titi</option>
      <optgroup label="les autres">
         <option value="3">tata</option>
         <option value="4">tutu</option>
         <option value="5">tete</option>
      </optgroup>
      </select>
      <hr/><textarea name="texte" rows="10" cols="80">Raconte-moi une histoire...</textarea><hr/>
      <input type="submit" name="maSoumission" id="soumission" />
      <input type="submit" name="action" value="Insert"/>
      <input type="submit" name="action" value="Update"/>
      <input type="image" src="Images/logoasi.png" alt="logoasi.png" name="action" width="75"/>
      <input type="reset" value="Reset"/>
   </fieldset>
</form>
```

Les formulaires (9/9)
Exemple de formulaire

Formulaire.html

Exemple de formulaire

Nom : Prénom :

Une checkbox Homme Femme

Fichier : Choisissez un fichier Aucun fichier choisi

toto
titi
les autres

Raconte-moi une histoire...

Valider Insert Update Reset

Les CSS (1/12)
Description

Cascading Style Sheets
Feuilles de style en cascade

- Un document en mode texte
- Utilisable par des documents HTML ou XML
- Présentation identique de tous les documents HTML ou XML
- Apporte la modularité du formatage
- Séparation du contenu et de la présentation
- 3 normes : CSS 1, CSS 2, CSS 3

Les CSS (2/12)
Inclusion d'une CSS

Trois possibilités d'inclusion :

- Directement dans les balises (à éviter)

```
<h2 style="color:red">Titre en rouge</h2>
```

- Définition de la CSS dans le fichier *via* la balise `<style>`

```
<head>
  <style type="text/css">
    déclaration des styles
   </style>
</head>
```

- Déclaration d'un lien vers la CSS via la balise `<link>`

```
<head>
  <link href="fichier.css" rel="stylesheet" type="text/css"/>
</head>
```

Les CSS (3/12)
Déclaration d'une règle

```
sélecteur {
    propriété1: valeur1;
    ...
    propriétéN: valeurN;
}
```

- On peut mettre autant de couples propriété/valeur que voulu, séparés par des ;
- Commentaire : `/* Commentaire */`

Exemple

```
h2 {color:red;}
```

Les CSS (4/12)
Déclaration d'une règle (suite)

- Sélecteur de type : nom de balise

  `h1 {color: red; background-color: yellow}`
- Sélecteur universel : `*`
- Sélecteur d'ID : #

  `#important {color: red}` ou `p#important {font-size: 30pt}`

  Sélection de : `<p id="important">paragraphe</p>`
- Sélecteur de classe : .

  `.special {font-size: 20pt}` ou `h1.special {font-size: 60pt}`

  Sélection de : `<h1 class="special">Titre spécial</h1>`

Remarque

Les id sont uniques sur une même page.
Les class s'appliquent à plusieurs balises.

Les CSS (5/12)
Déclaration d'une règle (suite)

- Sélecteur de descendant

  `p h2 {color: green}` : "les h2 qui sont dans un p"
- Sélecteur d'enfant

  `p > h2 {font-size: 30pt}` : "les h2 qui sont directement dans un p"
- Sélecteur d'adjacent :

  `p + h2 {font-size: 10pt}` : "les h2 qui sont directement après un p"
- Sélecteur de pseudo-classe

```
a:visited {color: brown}
p em:first-child {font-weight: bold }
```

Les CSS (6/12)
Déclaration d'une règle (suite et fin)

- Sélecteur de pseudo-élément "première lettre"

```
p:first-letter {font-size: 200%; font-weight: bold;
                float: left; text-transform: capitalize}
```

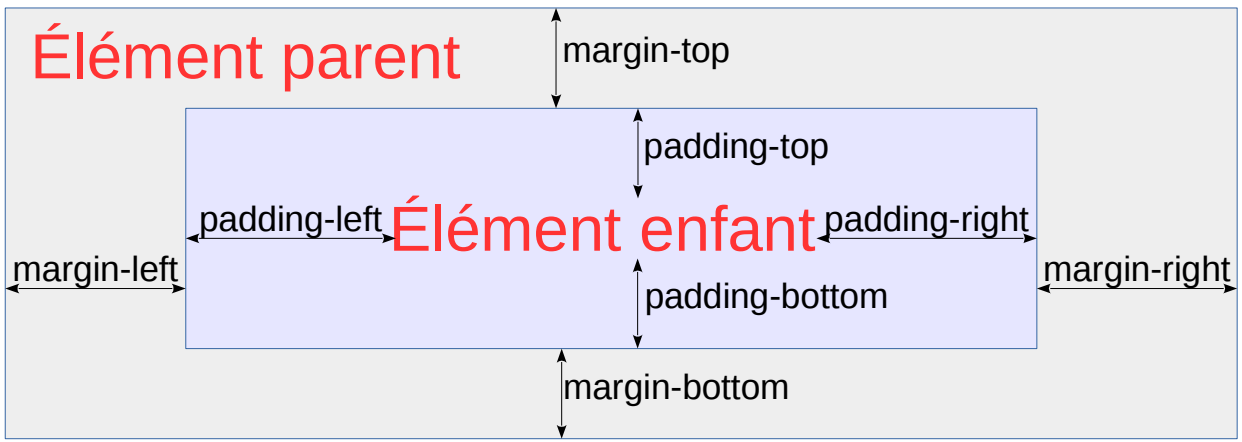
- Sélecteurs de pseudo-élément "avant" et "après"
  - `:before` et `:after` génèrent du contenu avant et après un élément
  - Le contenu généré est déclaré par la propriété `content`
  - Les pseudo-éléments générés héritent du style de l'élément référence

```
dt:before { content: "->"; }
```

- Éléments génériques : Div/Span
  - Élément de bloc générique : `<div> ... </div>`
  - Élément inline générique : `<span> ... </span>`

Les CSS (6/12)
Éléments et propriétés

- Propriétés (`http://www.w3schools.com/cssref/`)
  - polices de caractères :
    `font-size, font-style, font-family, font-weight`
  - paragraphes :
    `line-height, text-align, text-indent, text-transform`
  - blocs :
    `height, width, margin-right, margin-left, margin-top, margin-bottom, padding-right, border-style, border-top-width, ...`



Les CSS (7/12)
Cascade

Tous les styles, peu importe où ils sont définis, se **fusionnent** dans l'**ordre de chargement**, en une seule feuille de style avec un système d'**héritage** et peuvent s'**écraser**.

```
h1 {
color : red;
font-size : 18px;
}
h2 {
color : green;
}
```

+

```
h1 {
color : blue;

}
h2 {
color : green;
font-size : 12px;
}
```

=

```
h1 {
color : blue;
font-size : 18px;
}
h2 {
color : green;
font-size : 12px;
}
```

Les CSS (8/12)
Cascade (suite)

Tous les styles applicables sont appliqués.

```
h1 {color : red;}
p { backgroung-color:grey; }
.untitre { font-size : 64px; }
```

+

```
<p>
    <h1 class= "untitre">Le titre qui a du style</h1>
</p>
```

=

Le titre qui a du style

Les CSS (9/12)
Positionnement

- Fonctionnement par défaut :
  Dans le flux, les éléments les uns en dessous des autres
- Positionnement flottant (par rapport au bloc contenant) :
  `float: left;`
- Positionnement absolu (par rapport à la fenêtre, hors flux) :
  `position: absolute; top: x px; left: y px;`
- Positionnement relatif (par rapport à sa position dans le flux) :
  `position: absolute; top: x px; left: y px;`

`http://www.w3schools.com/css/css_positioning.asp`

Les CSS (10/12)
Exemple d'utilisation

Exemple

```
<!DOCTYPE html>
<html>
   <head>
      <title>Exemple d'utilisation des CSS</title>
      <meta http-equiv="content-type" content="text/html;charset=utf-8" />
      <link href="Livres.css" rel="stylesheet" type="text/css"/>
   </head>
   <body>
      <dl>
         <dt>CSS pour les nuls</dt><dd>Yvon Tengrece</dd>
         <dt>XML pour pas mieux</dt><dd>Alain Stigateur</dd>
         <dt>XHTML pour <em>rien</em></dt><dd>Arnie Ka</dd>
      </dl>
      <p>Voici un petit texte juste pour <strong>montrer</strong> la différence avec une <span id="
            important">liste</span>. On remarquera que la largeur est fixée à <em>500px.</em></p>
      <p>et un second paragraphe derrière.</p>
      <div>Et un span dans un div pour tester br/i : <span class="i"><span class="br">i puis br (on
            favorise i)</span></span> et <span class="br"><span class="i">br puis i (on tient compte de i
             et br)</span></span>.</div>
   </body>
</html>
```

Les CSS (11/12)
Exemple d'utilisation

```
html { font-family: arial; font-size: 20px; font-style: normal; color: #000000; background: #cccccc;
       margin: 5px; width: 500px; }

p { text-align: justify }

p:first-letter { text-transform: capitalize }

p + p { margin-top: 10mm }

dl { margin-left: 50px; }

dt { font-style: italic; }

dt em { font-weight: bold; }

dd { color: #ff6600; }

#important { color: red }

.i>.br { font-weight: normal; font-style: italic; }

.br>.i { font-weight: bold; font-style: italic; }
```

Les CSS (12/12)
Exemple d'utilisation

Sans CSS :

CSS pour les nuls
Yvon Tengrece
XML pour pas mieux
Alain Stigateur
XHTML pour *rien*
Arnie Ka

Voici un petit texte juste pour **montrer** la différence avec une liste. On remarquera que la largeur est fixée à *500px*.

et un second paragraphe derrière.

Et un span dans un div pour tester br/i : i puis br (on favorise i) et br puis i (on tient compte de i et br).

Avec CSS :

CSS pour les nuls
Yvon Tengrece
XML pour pas mieux
Alain Stigateur
XHTML pour ***rien***
Arnie Ka

Voici un petit texte juste pour **montrer** la différence avec une liste. On remarquera que la largeur est fixée à *500px*.

Et un second paragraphe derrière.

Et un span dans un div pour tester br/i : *i puis br (on favorise i)* et ***br puis i (on tient compte de i et br)***.

Documentation et liens

- **HTML 5**
  w2schools : `http://www.w3schools.com/html5/`
  Toutes les balises :
  `http://www.w3schools.com/html5/html5_reference.asp`
- **CSS**
  w2schools : `http://www.w3schools.com/html5/`
  Balises :
  `http://www.w3schools.com/html5/html5_reference.asp`
- **Validation**
  W3C (DTD based) : `http://validator.w3.org`
  Validator.ne (non-DTD) : `http://html5.validator.nu`
  L'indispensable Firebug pour le débuggage !
- **Compatibilité navigateurs**
  `http://caniuse.com`