Conception d'Interfaces Web

Licence 1

3. Créer un site Web avec HTML5 et CSS3

Bruno Jobard

Présentation de pages Web par les feuilles de style

CSS Zen Garden

The Beauty of CSS Design

A demonstration of what can be accomplished through CSS-based design. Select any style sheet from the list to load it into this page.

Download the example html file and css file

The Road to Enlightenment

Littering a dark and dreary road lay the past relics of browser-specific tags, incompatible DOMs, broken CSS support, and abandoned browsers.

We must clear the mind of the past. Web enlightenment has been achieved thanks to the tireless efforts of folk like the W3C, WaSP, and the major browser creators.

The CSS Zen Garden invites you to relax and meditate on the important lessons of the masters. Begin to see with clarity. Learn to use the time-honored techniques in new and invigorating fashion. Become one with the web.

So What is This About?

There is a continuing need to show the power of CSS. The Zen Garden aims to excite, inspire, and encourage participation. To begin, view some of the existing designs in the list. Clicking on any one will load the style sheet into this very page. The HTML remains the same, the only thing that has changed is the external CSS file. Yes, really.

CSS allows complete and total control over the style of a hypertext document. The only way this can be illustrated in a way that gets people excited is by demonstrating what it can truly be, once the reins are placed in the hands of those able to create beauty from structure. Designers and coders alike have contributed to the beauty of the web; we can always push it further.

Participation

Strong visual design has always been our focus. You are modifying this page, so strong CSS skills are necessary too, but the example files are commented well enough that even CSS novices can use them as starting points. Please see the CSS Resource Guide for advanced tutorials and tips on working with CSS.

You may modify the style sheet in any way you wish, but not the HTML. This may seem daunting at first if you've never worked this way before, but follow the listed links to learn more, and use the sample files as a guide.

Download the sample HTML and CSS to work on a copy locally. Once you have completed your masterpiece (and please, don't submit half-finished work) upload your CSS file to a web server under your control. Send us a link to an archive of that file and all associated assets, and if we choose to use it we will download it and place it on our server.

Benefits

Why participate? For recognition, inspiration, and a resource we can all refer to showing people how amazing CSS really can be. This site serves as equal parts inspiration for those working on the web today, learning tool for those who will be tomorrow, and gallery of future techniques we can all look forward to.

Requirements

HTML seul

CSS ZEN GARDEN

The Beauty of CSS Design

A DEMONSTRATION OF WHAT CAN BE ACCOMPLISHED THROUGH CSS-BASED DESIGN. SELECT ANY STYLE SHEET FROM THE LIST TO LOAD IT INTO THIS PAGE.

DOWNLOAD THE EXAMPLE HTML FILE AND CSS FILE

The Road to Enlightenment

Littering a dark and dreary road lay the past relics of browser-specific tags, incompatible DOMs, broken CSS support, and abandoned browsers.

We must clear the mind of the past. Web enlightenment has been achieved thanks to the tireless efforts of folk like the W3C, WaSP, and the major browser creators.

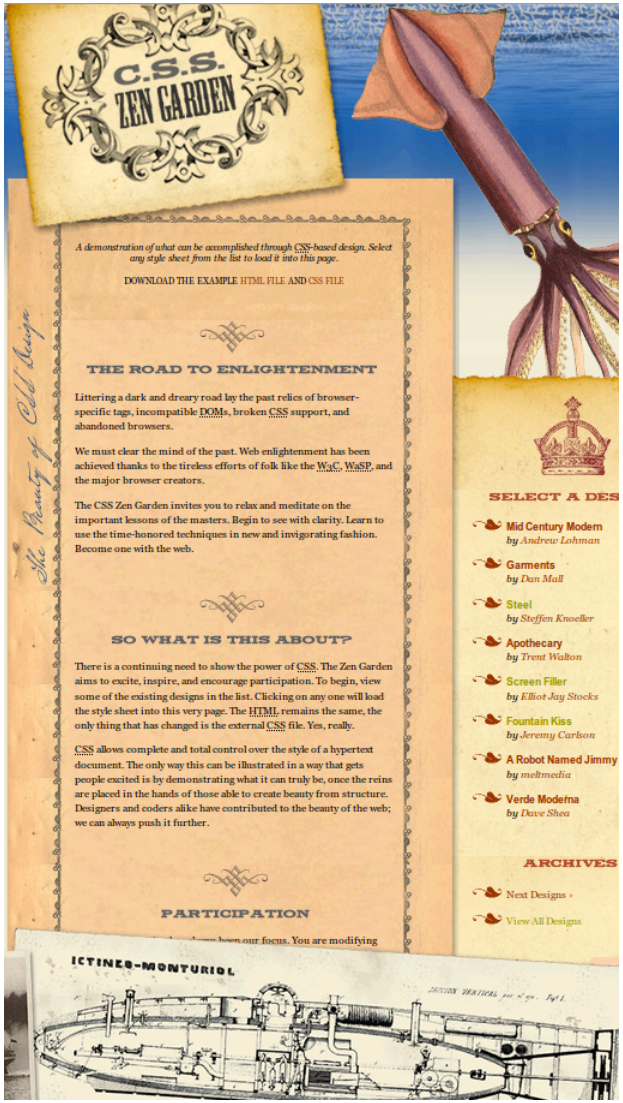
The CSS Zen Garden invites you to relax and meditate on the important lessons of the masters. Begin to see with clarity. Learn to use the time-honored techniques in new and invigorating fashion. Become one with the web.

So What is This About?

There is a continuing need to show the power of CSS. The Zen Garden aims to excite, inspire, and encourage participation. To begin, view some of the existing designs in the list. Clicking on any one will load the style sheet into this very page. The HTML remains the same, the only thing that has changed is the external CSS file. Yes, really.

CSS allows complete and total control over the style of a hypertext document. The only way this can be illustrated in a way that gets people excited is by demonstrating what it can truly be, once the reins are placed in the hands of those able to create beauty from structure. Designers and coders alike have contributed to the beauty of the

HTML+CSS



Démo : CSS Zen Garden

Les feuilles de style CSS

CSS est un langage pour décrire des règles de présentation pour des pages HTML.

Ces règles sont rassemblées dans des fichiers qui constituent des « feuilles de style ».

Avant l'invention du langage CSS en 1996, la présentation des pages Web était gérée par des balises HTML. Le mélange des balises de structure et de présentation alourdissait le code et rendait les pages complexes à maintenir.

Avantages des feuilles de style

La séparation de la structure et de la présentation dans les pages Web amène plusieurs avantages :

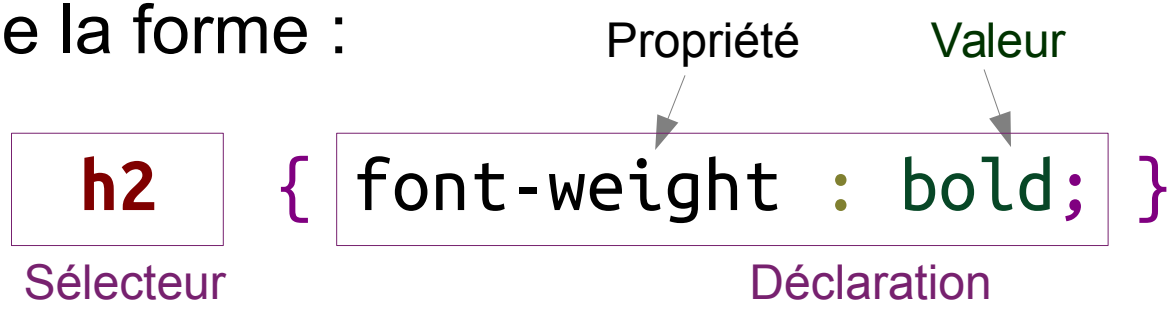
- Utiliser une même feuille de style pour plusieurs pages Web (mutualisation et uniformité)
- Faire évoluer séparément contenu et présentation (évolutivité et maintenance des sites Web)
- Consulter un même document dans des contextes différents (écran, imprimante, mobile...)

CSS : Cascading Style Sheet

- La « cascade » utilisée par CSS repose sur 3 niveaux de définitions de styles
  1. Le niveau **agent**, c.à.d **navigateur** : il s'agit du style par défaut du navigateur
  2. le niveau **auteur** : style défini par l'auteur de la page. Ce style est référencé dans le document HTML
  3. Le niveau **utilisateur** : directives de style définies par celui qui va consulter la page (réglage de « préférences » par exemple)
- Le style **utilisateur** l'emporte sur le style **auteur** et lui-même sur le style par défaut du **navigateur**
- La cascade repose aussi sur différents niveaux de priorités de règles de styles

Les règles CSS

Une règle CSS est de la forme :



Le sélecteur indique à quels éléments et dans quelles circonstances s'applique la déclaration qui est à droite.

Pour des raisons de concision, on peut grouper plusieurs sélecteurs et plusieurs déclaration :

```
h1, h2, h3 { font-weight : bold;
             color :  red;
           }
```

Les sélecteurs sont séparés par des virgules, les déclarations par des point-virgules.

Les propriétés CSS

Les versions successives de CSS ont défini un nombre important de propriétés. Le but n'est pas ici d'en faire un inventaire exhaustif. Quelques exemples :

- **`font-size:12pt;`** (ou **`larger`**, ou **`120%`**)
- **`font-weight`** (ex : **`bold`**)
- **`text-align`** (ex : **`center`**, **`justify`**, ...)
- **`text-transform`** (ex : **`uppercase`**)
- **`list-style-type`** (ex : **`circle`**)
- **`color, background-color`** (ex : **`blue`**, **`rgb(53,0,132)`**)

Les types de propriétés CSS

- **Les dimensions** peuvent être
  - absolues : mm, cm, in, pt (1/72 in), pc (12 pt)
  - relatives au contexte : %, em (largeur d'un M), ex (hauteur d'un x)
  - relatives à l'écran : px (pixel)
- **Les couleurs** s'expriment par une donnée RGB (RVB en français) : **`rgb(12, 214, 130)`** ou **`#0CD682`** ou par un des 16 noms de base (**`red`, `black`, `yellow`**, ...)
- **Les chaînes** ou les valeurs énumérées (ex : **`center`**)
- **Les URL** notées par un qualificateur **`url`**
  ex : **`url(http://www.uppa.fr)`**

Les sélecteurs

Un sélecteur commence chaque règle CSS :

```
h2 { font-weight : bold; }
```

Sélecteur

- Il permet de *sélectionner* une **sous-arborescence** du document HTML sur laquelle appliquer le style.

- Le sélecteur peut être simple ou composé :

```
h2 + ul > li em { font-weight : bold; }
```

Sélecteur composé

Les opérateurs précisent le mode de sélection

Les sélecteurs de type

Code HTML

```
<body>
<h2>Desserts aux <em>fruits</em></h2>
<p>Liste des <b>desserts</b> :</p>
<ul>
  <li><em>banana</em> split</li>
  <li>compote de <em>pommes</em></li>
  <li><b>pêche</b> melba</li>
</ul>
</body>
```

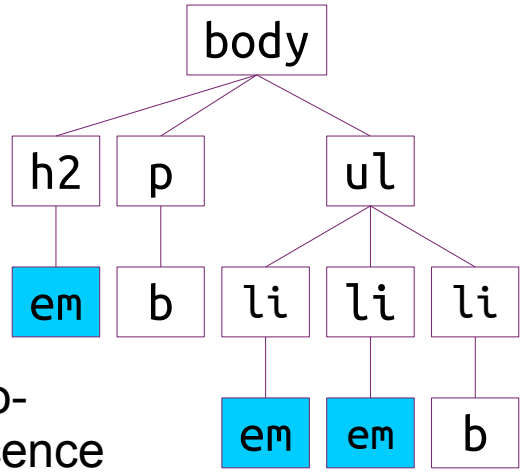
Rendu dans le navigateur

Desserts aux *fruits*

Liste des **desserts** :

- *banana* split
- compote de *pommes*
- **pêche** melba

Arbo-rescence



Code CSS

```
em { color : blue; }
```

Signification du sélecteur de type

« Sélectionne les éléments `em` »

Les sélecteurs de type

Code HTML

```
<body>
<h2>Desserts aux <em>fruits</em></h2>
<p>Liste des <b>desserts</b> :</p>
<ul>
  <li><em>banana</em> split</li>
  <li>compote de <em>pommes</em></li>
  <li><b>pêche</b> melba</li>
</ul>
</body>
```

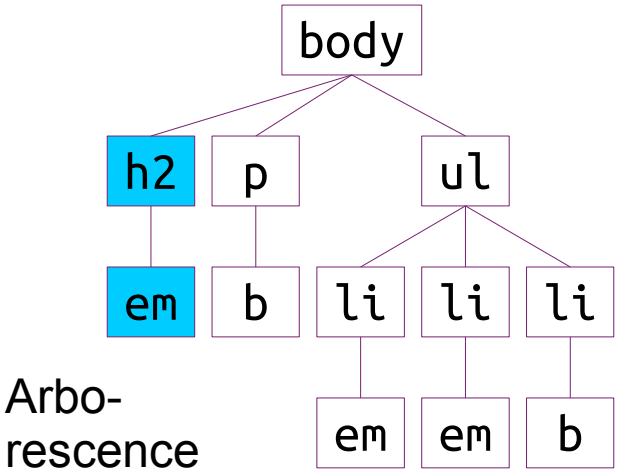
Rendu dans le navigateur

Desserts aux *fruits*

Liste des **desserts** :

- *banana* split
- compote de *pommes*
- **pêche** melba

Arbo-rescence



Code CSS

```
h2 { color : blue; }
```

Signification du sélecteur de type

« Sélectionne les éléments `h2` »

Les sélecteurs de type

Code HTML

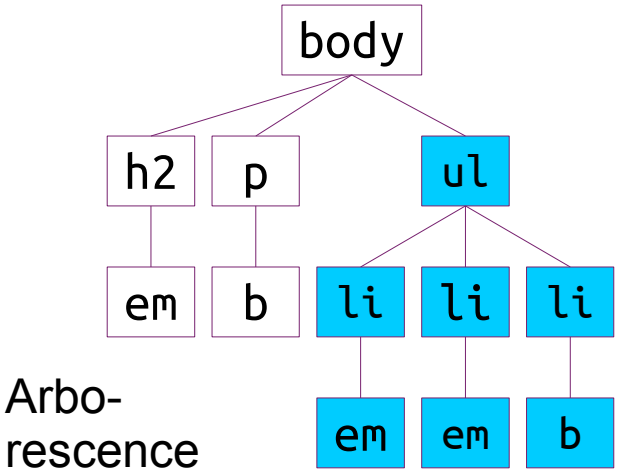
```
<body>
<h2>Desserts aux <em>fruits</em></h2>
<p>Liste des <b>desserts</b> :</p>
<ul>
  <li><em>banana</em> split</li>
  <li>compote de <em>pommes</em></li>
  <li><b>pêche</b> melba</li>
</ul>
</body>
```

Rendu dans le navigateur

Desserts aux *fruits*

Liste des **desserts** :

- *banana* split
- compote de *pommes*
- **pêche** melba



Arbo-rescence

Code CSS

```
ul { color : blue; }
```

Signification du sélecteur de type

« Sélectionne les éléments `ul` »

Les sélecteurs descendants

Code HTML

```
<body>
<h2>Desserts aux <em>fruits</em></h2>
<p>Liste des <b>desserts</b> :</p>
<ul>
  <li><em>banana</em> split</li>
  <li>compote de <em>pommes</em></li>
  <li><b>pêche</b> melba</li>
</ul>
</body>
```

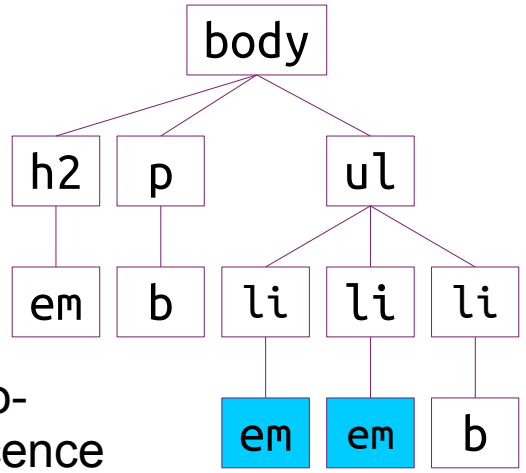
Rendu dans le navigateur

Desserts aux *fruits*

Liste des **desserts** :

- *banana* split
- compote de *pommes*
- **pêche** melba

Arbo-rescence



Code CSS

```
ul em { color : blue; }
```

Signification du sélecteur descendant : ' ' (espace)

« Sélectionne les éléments `em` **descendants** d'éléments `ul` »

Les sélecteurs d'enfants

Code HTML

```
<body>
<h2>Desserts aux <em>fruits</em></h2>
<p>Liste des <b>desserts</b> :</p>
<ul>
  <li><em>banana</em> split</li>
  <li>compote de <em>pommes</em></li>
  <li><b>pêche</b> melba</li>
</ul>
</body>
```

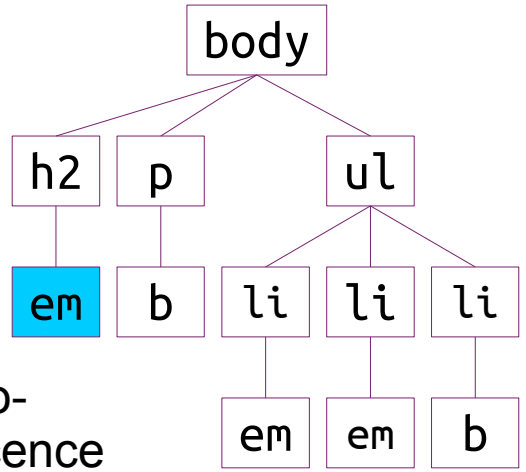
Rendu dans le navigateur

Desserts aux *fruits*

Liste des **desserts** :

- *banana* split
- compote de *pommes*
- **pêche** melba

Arbo-rescence



Code CSS

```
h2 > em { color : blue; }
```

Signification du sélecteur d'enfants : '>'

« Sélectionne les éléments **em enfants de** l'élément **h2** »

Les sélecteurs adjacents directs

Code HTML

```
<body>
<h2>Desserts aux <em>fruits</em></h2>
<p>Liste des <b>desserts</b> :</p>
<ul>
  <li><em>banana</em> split</li>
  <li>compote de <em>pommes</em></li>
  <li><b>pêche</b> melba</li>
</ul>
</body>
```

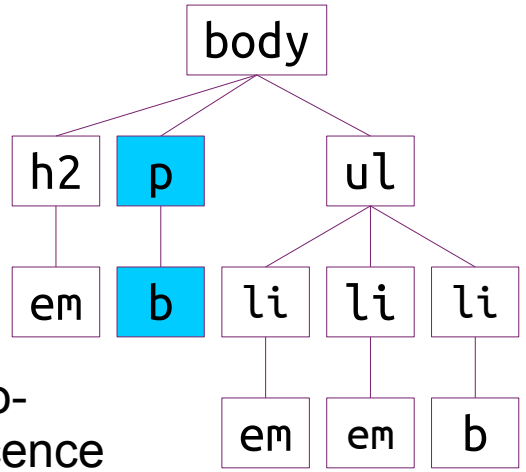
Rendu dans le navigateur

Desserts aux *fruits*

Liste des **desserts** :

- *banana* split
- compote de *pommes*
- **pêche** melba

Arbo-rescence



Code CSS

```
h2 + p { color : blue; }
```

Signification du sélecteur adjacent direct : '+'

« Sélectionne les éléments **p** **frères directs** d'un élément **h2** »

Les sélecteurs adjacents indirects

Code HTML

```
<body>
<h2>Desserts aux <em>fruits</em></h2>
<p>Liste des <b>desserts</b> :</p>
<ul>
  <li><em>banana</em> split</li>
  <li>compote de <em>pommes</em></li>
  <li><b>pêche</b> melba</li>
</ul>
</body>
```

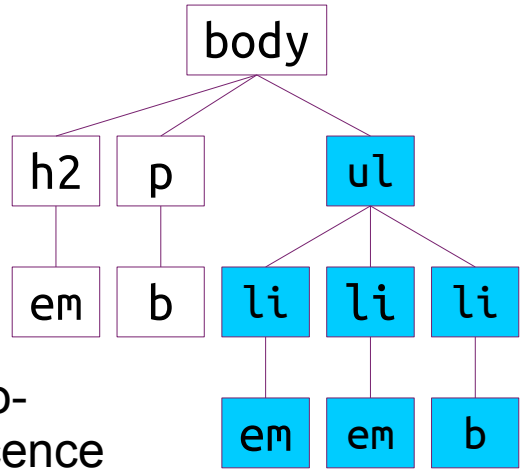
Rendu dans le navigateur

Desserts aux *fruits*

Liste des **desserts** :

- *banana* split
- compote de *pommes*
- **pêche** melba

Arbo-rescence



Code CSS

```
h2 ~ ul { color : blue; }
```

Signification du sélecteur adjacent indirect : '~'

« Sélectionne les éléments `ul` **frères suivants** d'un élément `h2` »

Les sélecteurs composés

Code HTML

```
<body>
<h2>Desserts aux <em>fruits</em></h2>
<p>Liste des <b>desserts</b> :</p>
<ul>
  <li><em>banana</em> split</li>
  <li>compote de <em>pommes</em></li>
  <li><b>pêche</b> melba</li>
</ul>
</body>
```

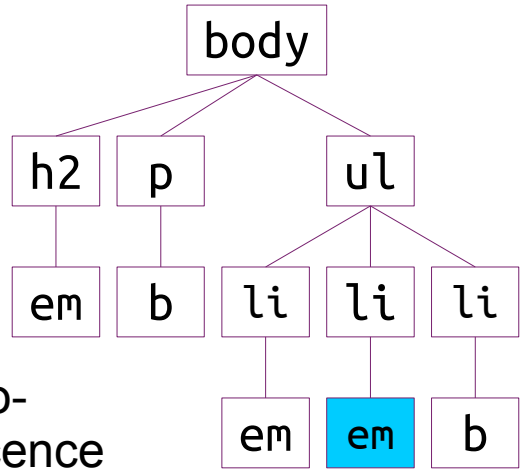
Rendu dans le navigateur

Desserts aux *fruits*

Liste des **desserts** :

- *banana* split
- compote de *pommes*
- **pêche** melba

Arbo-rescence



Code CSS

```
h2~ul>li+li em { color : blue; }
```

Signification du sélecteur composé

« Sélectionne les `em` **imbriqués** dans un `li` qui **suit** un `li` **fils** d'un `ul` **frère suivant** d'un `h2` »

Les sélecteurs de classe

- Pour chacun des éléments HTML, on peut spécifier l'attribut universel **`class`** dont la valeur est un identificateur. Plusieurs éléments peuvent recevoir la même valeur pour cet attribut. Exemple de code HTML :

```
<div class="question"> ...?    </div>
<div class="réponse" > ...!    </div>
<div class="question"> Really? </div>
<div class="réponse" > Sure!!  </div>
```

- La feuille de style pourra préciser des styles différents pour les deux classes :

```
div.question { font-style:italic; color:gray; }
div.reponse  { font-weight:bold; }
```

- Ou encore de manière abrégée :
  - `.question { ... }` et `.reponse { ... }`

Les sélecteurs d'identifiant

- Pour chacun des éléments HTML on peut spécifier l'attribut universel **id** dont la valeur est un identificateur. Un seul élément au plus peut recevoir un identifiant donné.
- Exemple de code HTML :

```
<table id="fruits"> ... </table>
<table id="legumes"> ... </table>
```

- La feuille de styles :

```
#fruits { color : yellow; }
#legumes { color : green; }
```

Sélection par pseudo-classes et pseudo-éléments

- Quelques pseudo-classes
  - `:first-child` Premier enfant d'un élément
  - `:lang(fr)` Élément dont la langue correspond à symb exemple : `:lang(fr)`
  - `:hover` L'élément est survolé par la souris
  - `:active` L'élément est sélectionné
  - `:visited` Liens déjà visités
  - `:link` Liens non encore visités
- Quelques pseudo-éléments
  - `:first-line` Première ligne d'un élément
  - `:first-letter` Première lettre d'un élément

Récapitulatif des sélecteurs

*	Tout élément
Elt	Tout élément Elt
Elt1 Elt2	Tout élément Elt2 qui est imbriqué dans un élément Elt1
Elt1>Elt2	Tout élément Elt2 qui est fils d'un élément Elt1
Elt1+Elt2	Tout élément Elt2 qui suit immédiatement un élément Elt1
Elt1~Elt2	Tout élément Elt2 qui suit un élément Elt1 (même parent)
Elt[att]	Tout élément Elt dont l'attribut est positionné
Elt[att=val]	Tout élément Elt dont l'attribut est positionné à val
Elt.c	Tout élément Elt dont l'attribut classe="c"
Elt#identité	L'élément Elt dont l'attribut id="identité"
Elt :prop	Tout élément Elt vérifiant la propriété prop (pseudo-classe ou pseudo-élément)

Où se définissent les styles ?

- Dans une feuille externe (meilleure solution). Cette feuille est référencée dans le document HTML par :

```
<link rel="stylesheet" href="monstyle.css" />
```

- Dans l'entête (head) du document HTML, au sein d'un élément style :

```
<head>
  <style>
    h1 { color:blue; }  /* du CSS au milieu du HTML */
  </style>
  <title> ... </title>
</head>
```

- Les deux peuvent se combiner. Les règles définies dans le document s'ajoutent à la suite des règles externes.

L'attribut style

- On peut également mentionner des déclaration de style pour chaque élément du code HTML, via un attribut universel **style** :

```
<td style="text-align:center; color:blue"> ... </td>
```

  - Par cette méthode, on perd tous les avantages de la séparation forme/fond (factorisation, maintenance, évolutivité)
  - la méthode doit être réservée à des mises-en-page très spécifiques (présentation de certains tableaux par exemple)
  - Les valeurs des propriétés fixées de cette manière surpassent celles définies ailleurs (voir priorités)

Priorités des sélecteurs

- Les déclarations sont prises en compte dans l'ordre suivant (priorités décroissantes) :
  - règle par sélecteur par identifiant (id)
  - règle avec sélecteur par attribut, classe ou pseudo-classe
  - règle avec sélecteur par élément
  - héritées d'un élément englobant

Priorités des sélecteurs

- À priorités égales, la règle appliquée dépend de l'endroit où elle a été définie (par ordre de priorité décroissante) :
  - Localement à l'élément (attribut style)
  - Défini dans l'entête du document HTML (balise style)
  - Dans une feuille de style extérieure
- En cas de d'égalité et de même endroit
  - la dernière déclaration prévaut.

Priorité : exemple

- De quelle couleur sera le contenu des tableaux après application des styles ?

```
table#resultats  { color : green; }
table[border]    { color : blue;  }
table.importante { color : red;   }
table            { color : black; }
```

```
<table>                                  ... </table>
<table border="1" id="resultats">        ... </table>
<table border="1">                       ... </table>
<table border="1" class="importante">    ... </table>
```

- Si on inverse l'ordre des règles « table[border] » et « table.importante », la dernière table s'affiche en ?

Priorité : exemple

- De quelle couleur sera le contenu des tableaux après application des styles ?

```
table#resultats  { color : green; }
table[border]    { color : blue;  }
table.importante { color : red;   }
table            { color : black; }
```

```
<table>                                  ... </table>  <!-- noir  -->
<table border="1" id="resultats">        ... </table>  <!-- vert  -->
<table border="1">                       ... </table>  <!-- blue  -->
<table border="1" class="importante">    ... </table>  <!-- rouge -->
```

- Si on inverse l'ordre des règles « table[border] » et « table.importante », la dernière table s'affiche en bleu !

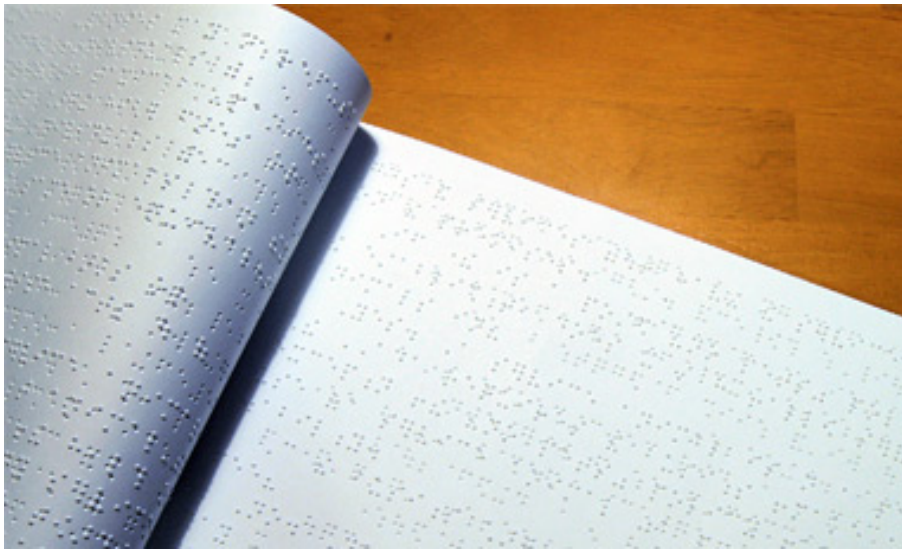
Adaptation aux médias

- CSS distingue plusieurs types de médias. Parmi eux :
  - `screen` moniteurs couleurs en mode graphique
  - `print` imprimantes
  - `tty` terminaux texte
  - `braille` appareils tactiles ou imprimantes Braille
- Une feuille de style peut comporter des sections différentes pour plusieurs médias. Par exemple :

```
@media print {
    body { font-size : 12pt }
}

@media screen {
    body { font-size : 10pt }
}

@media screen, print {
    body { line-height : 1.2ex }
}
```



Une impression en braille

Adaptation aux médias

- Il est possible d'attacher à un document HTML plusieurs feuilles de styles en précisant le type de média pour lequel elles s'appliquent :

```
<link href="style1.css" rel="stylesheet"
media="screen"/>

<link href="style2.css" rel="stylesheet" media="print"/>
```

Adaptation aux médias

- Mais comment gérer plusieurs tailles d'écran ?
  - CSS3 apporte les ***requêtes de médias*** pour s'adapter plus précisément.

Requêtes de médias

- ***Media Queries CSS3*** en anglais.
- Récupération d'informations fines sur le dispositif d'affichage
- Dans le HTML :

```
<link  rel="stylesheet"
       media="screen and (max-width: 640px)"
       href="smallscreen.css" />
```

- Dans le CSS :

```
@media screen and (min-width: 200px) and (max-width: 640px) {
  .bloc { display: block;
          clear  : both;
  }
}
```

Requêtes de médias

- Quelques caractéristiques interrogeable :
  - `color` support de la couleur
  - `height` hauteur en nombre de pixels
  - `orientation` portrait ou landscape
  - `monochrome` monochrome ou niveaux de gris
  - `resolution` du périphérique (en dpi)
  - `width` largeur en nombre de pixels

- Les critères soulignés peuvent être préfixé par `min-` ou `max-` (`max-width: 640px`)

Requêtes de médias

- Ex : orientation de l'écran :

```
<link  rel="stylesheet"
       media="(orientation:portrait)"
       href="portrait.css">

<link  rel="stylesheet"
       media="(orientation:landscape)"
       href="paysage.css">
```

- Ex : impression sur une page > 10 cm

```
<link  rel="stylesheet"
       media="print and (min-width: 10cm)"
       href="medium_print.css">
```