LIF4 - Programmation Web - cours CSS

Fabien Duchateau

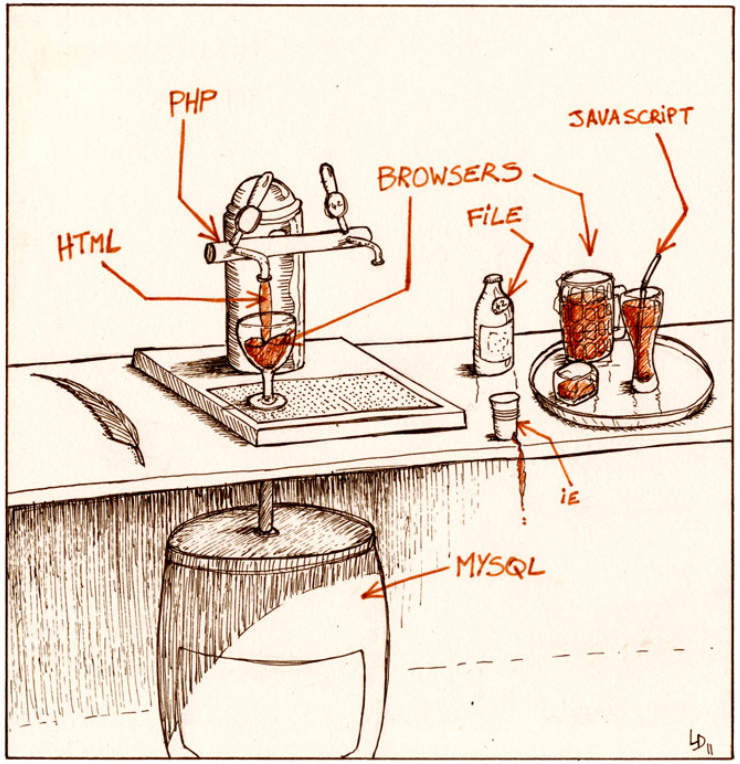
fabien.duchateau [at] univ-lyon1.fr

Université Claude Bernard Lyon 1

Automne 2012

http://liris.cnrs.fr/~fduchate/ens/2012-2013/LIF4/

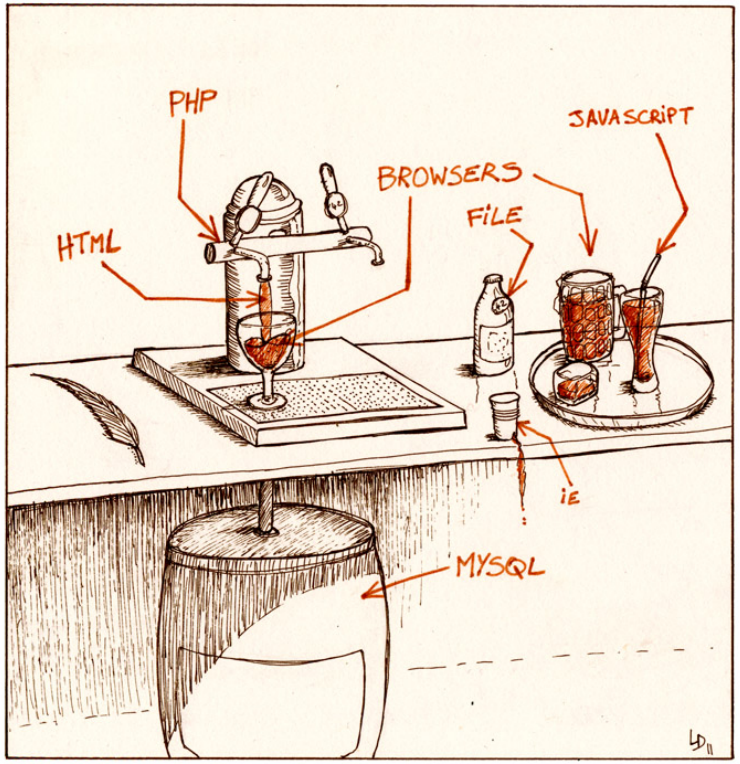
Rappel du cours (ou du week-end) précédent



Et le CSS dans tout ça ?

http://www.luc-damas.fr/humeurs/

Rappel du cours (ou du week-end) précédent



Et le CSS dans tout ça ?

Glaçons, parasol, alcool/sirop à mélanger avec la bière, etc.

`http://www.luc-damas.fr/humeurs/`

Mise en forme

Avant : mise en forme dans le code HTML, au milieu des données (balises `<center>`, `<font>`, attribut `align`, etc., devenus obsolètes)

Aujourd'hui, séparation des données et de la mise en forme :

- Facilité de maintenance
- Facilité de portage pour un autre site
- Simplification et lisibilité du code
- Uniformisation du code entre les pages du même site et entre différents médias (e.g., impression, écran)
- Améliorer l'accessibilité

Mise en forme

Avant : mise en forme dans le code HTML, au milieu des données (balises `<center>`, `<font>`, attribut `align`, etc., devenus obsolètes)

Aujourd'hui, séparation des données et de la mise en forme :

- Facilité de maintenance
- Facilité de portage pour un autre site
- Simplification et lisibilité du code
- Uniformisation du code entre les pages du même site et entre différents médias (e.g., impression, écran)
- Améliorer l'accessibilité

⇒ Objectifs des Cascading Style Sheets

Cascading Style Sheets

CSS pour Cascading Style Sheets :

- Langage de mise en forme et mise en page
- Origine : 1994-1995 (standard W3C en 1996)
- Toujours en développement (CSS3 depuis 2010)
- Extension d'un fichier CSS : `.css`
- Tutoriaux et wikilivres disponibles

---

http://fr.wikipedia.org/wiki/Feuilles_de_style_en_cascade
http://css.mammouthland.net/
http://fr.html.net/tutorials/css/

Plan

Différents niveaux d'insertion

On peut placer du code CSS à cinq niveaux :

- Dans une balise, via un attribut (**inline**)
- Dans un script intégré dans l'entête de la page web (**interne**)
- Dans **une feuille externe**
- Dans **plusieurs feuilles externes**
- L'utilisateurice possède sa feuille (cas très rare)

Différents niveaux d'insertion

On peut placer du code CSS à cinq niveaux :

- Dans une balise, via un attribut (**inline**)
- Dans un script intégré dans l'entête de la page web (**interne**)
- Dans **une feuille externe**
- Dans **plusieurs feuilles externes**
- L'utilisateurice possède sa feuille (cas très rare)

Meilleure solution ?

Différents niveaux d'insertion

On peut placer du code CSS à cinq niveaux :

- Dans une balise, via un attribut (**inline**)
- Dans un script intégré dans l'entête de la page web (**interne**)
- Dans **une feuille externe**
- Dans **plusieurs feuilles externes**
- L'utilisateurice possède sa feuille (cas très rare)

Meilleure solution ?

Idéalement dans plusieurs feuilles externes (e.g., pour chaque média), mais généralement dans une seule feuille externe

Différents niveaux d'insertion (2)

- Dans une balise (via un attribut `style`)

CSS dans une balise (inline)

<balise **style**="propriété : valeur ;">...</balise>

- Dans un script intégré dans l'entête de la page web

CSS dans l'entête (interne)

<head>
<style type="text/css" title="styles" media="all">
...
</style>
</head>

Différents niveaux d'insertion (2)

- Dans une balise (via un attribut `style`)

CSS dans une balise (inline)

<balise **style**="propriété : valeur ;">...</balise>

- Dans un script intégré dans l'entête de la page web

CSS dans l'entête (interne)

```
<head>
<style type="text/css" title="styles" media="all">
...
</style>
</head>
```

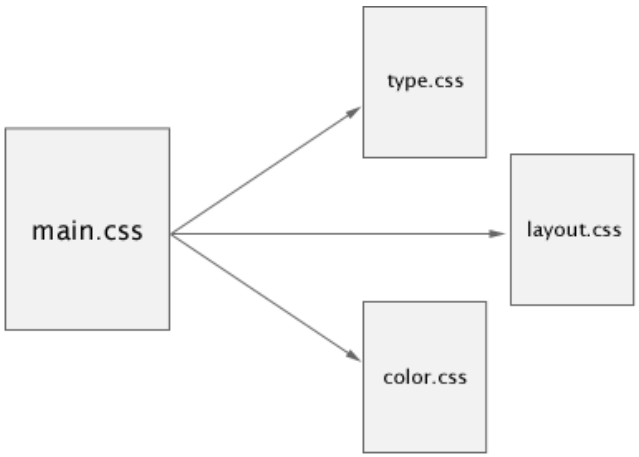
⇒ Limiter l'utilisation de ces deux niveaux d'insertion !

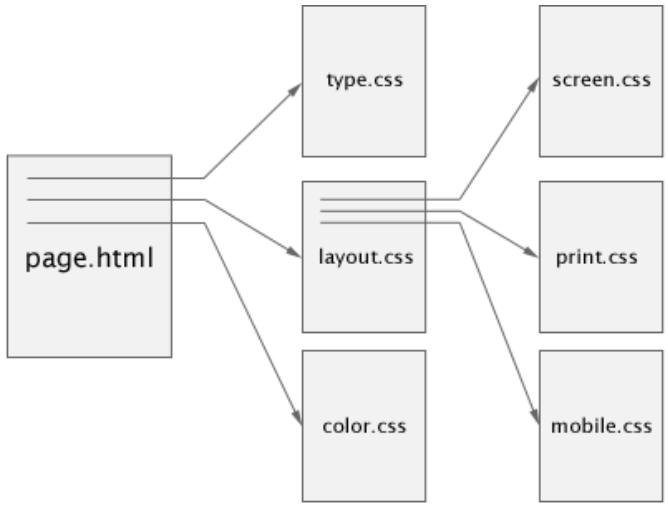
Différents niveaux d'insertion (3)

- Dans plusieurs feuilles externes

CSS dans plusieurs feuilles externes

```
<link rel="stylesheet" type="text/css" href="types.css">
<link rel="stylesheet" type="text/css" href="layout.css">
<link rel="stylesheet" type="text/css" href="color.css">
```





http://www.alistapart.com/articles/progressiveenhancementwithcss/

Différents niveaux d'insertion (4)

- Dans une seule feuille externe

CSS dans une feuille externe

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

Différents niveaux d'insertion (4)

- Dans une seule feuille externe

CSS dans une feuille externe

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

⇒ Solution à privilégier (pour les projets) :

- Bénéficie des avantages du CSS (séparation données/mise en forme, facilité de maintenance, etc.)
- Moins complexe que l'utilisation de plusieurs feuilles
- Possibilité de découper en plusieurs feuilles externes plus tard

Notion de cascades

Comme le nom l'indique, les styles peuvent être "en cascades", i.e., la possibilité de combiner des déclarations de styles externe, interne ou inline

- Priorité accordée à la déclaration la plus proche de l'élément à styliser, en cas de conflit sur une même propriété
- Ordre décroissant de priorité : [feuille utilisateurice,] inline, interne, externe

Exemple de cascades

Si on déclare une balise `paragraphe` avec des propriétés de style *fond rouge, alignement centré*, et qu'une feuille externe contraint les balises paragraphes au style *fond vert, police taille 12*, quelle est le style du paragraphe ?

Notion de cascades

Comme le nom l'indique, les styles peuvent être "en cascades", i.e., la possibilité de combiner des déclarations de styles externe, interne ou inline

- Priorité accordée à la déclaration la plus proche de l'élément à styliser, en cas de conflit sur une même propriété
- Ordre décroissant de priorité : [feuille utilisateurice,] inline, interne, externe

Exemple de cascades

Si on déclare une balise `paragraphe` avec des propriétés de style *fond rouge, alignement centré*, et qu'une feuille externe contraint les balises paragraphes au style *fond vert, police taille 12*, quelle est le style du paragraphe ?
⇒ fond rouge, alignement centré, police taille 12

Héritage

Imbrication des éléments HTML :

- La balise `html` contient les balises `head` et `body`, la balise `body` contient des balises `p`, `ul`, etc.

⇒ En CSS, un élément imbriqué **hérite des propriétés de son/ses parents**, sauf les propriétés redéfinies pour cet élément

Exemple d'héritage

Si on déclare une balise `body` avec une couleur de fond bleue, un paragraphe sans style spécifique et dont le parent direct est la balise `body` aura une couleur de fond bleue

Syntaxe du CSS

Syntaxe de base du CSS

```
sélecteur {
  propriete1 : valeur1 ;
  propriete2 : valeur2 ;
  ...
}
```

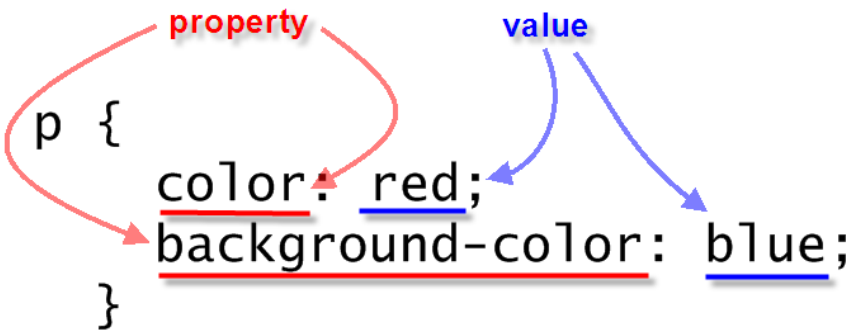
⇒ Déclaration des valeurs *valeur1* et *valeur2* pour les propriétés *propriete1* et *propriete2* d'un sélecteur dans une feuille externe ou interne

Syntaxe du CSS

Syntaxe de base du CSS

```
sélecteur {
  propriete1 : valeur1 ;
  propriete2 : valeur2 ;
  ...
}
```

⇒ Déclaration des valeurs *valeur1* et *valeur2* pour les propriétés *propriete1* et *propriete2* d'un sélecteur dans une feuille externe ou interne



Sélecteurs

Sélecteur : "pattern" qui sélectionne l'élément ou le groupe d'éléments sur lesquels va s'appliquer le style

Types de sélecteur possibles :

- Une balise
- Un nom de classe (introduit par .)
- Un identifiant d'élement (introduit par #)

Sélecteurs (3)

Sélecteur de type balise

```
balise {
  propriete1 : valeur1 ;
  propriete2 : valeur2 ;
}
```

⇒ Sélecteur de type **balise** :

- Sélectionne tous les éléments dont la balise est *balise*

Sélecteurs (3)

Rappel : les balises HTML ont un attribut `class`

Sélecteur de type nom de classe

```
.classe {
  propriete1 : valeur1 ;
  propriete2 : valeur2 ;
}
```

⇒ Sélecteur de type **nom de classe** :

- Sélectionne tous les éléments de la page dont l'attribut `class` vaut *classe*
- Plusieurs éléments (et de balises différentes) peuvent appartenir à la même classe

```
.ninja {
   color:#000000;
   visibility:hidden;
}
```

Sélecteurs (4)

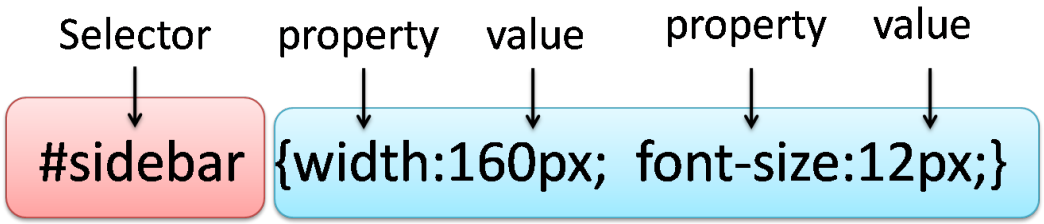
Rappel : les balises HTML ont un attribut `id`

Sélecteur de type identifiant

```
#identifiant {
  propriete1 : valeur1 ;
  propriete2 : valeur2 ;
}
```

⇒ Sélecteur de type **identifiant** :

- Sélectionne l'élément de la page dont l'attribut `id` vaut *identifiant*
- Unique pour chaque page !



Sélecteurs (5)

Combinaison et comparaison avec les sélecteurs :

- `balise1, balise2` : tous les éléments `balise1` et `balise2`
- `balise1 balise2` : tous les éléments `balise2` imbriqués dans un élément `balise1`
- `balise1>balise2` : tous les éléments `balise2` dont le parent est un élément `balise1`
- `[attrib]` : tous les éléments ayant l'attribut `attrib`
- `[attrib = "val"]` : tous les éléments ayant la valeur `val` pour l'attribut `attrib`
- ...

---

http://www.w3schools.com/cssref/css_selectors.asp

Sélecteurs (5)

Combinaison et comparaison avec les sélecteurs :

- `balise1, balise2` : tous les éléments `balise1` et `balise2`
- `balise1 balise2` : tous les éléments `balise2` imbriqués dans un élément `balise1`
- `balise1>balise2` : tous les éléments `balise2` dont le parent est un élément `balise1`
- `[attrib]` : tous les éléments ayant l'attribut `attrib`
- `[attrib = "val"]` : tous les éléments ayant la valeur `val` pour l'attribut `attrib`
- ...

Beaucoup d'autres possibilités, voir la liste sur *w3schools.com*

`http://www.w3schools.com/cssref/css_selectors.asp`

Priorités

Plusieurs insertions de CSS, plusieurs types de sélecteurs... qui a la priorité ?

Du plus prioritaire au moins prioritaire :

- Style dans la balise HTML (e.g., *<h1 style="color : black">*)
- Utilisation d'un identifiant (e.g., *h1#id {color : black ;}*)
- Utilisation d'une classe (e.g., *h1.noir {color : black ;}*)
- Imbrication d'éléments (e.g., *div h1 {color : black ;}*)
- Elément HTML (e.g., *h1 {color : black ;}*)

Exception avec la mention ! `important` en fin de déclaration (qui devient la déclaration à appliquer) :

Syntaxe de la mention *!important*

propriete **:** valeur **!important ;**

En résumé

- Déclaration CSS : `sélecteur {propriété: valeur;}`
- Notion d'héritage : un élément hérite des propriétés de ses parents
- Priorité selon les styles en cascades et les types de sélecteurs

En résumé

- Déclaration CSS : `sélecteur {propriété: valeur;}`
- Notion d'héritage : un élément hérite des propriétés de ses parents
- Priorité selon les styles en cascades et les types de sélecteurs

Plan

Généralités

Quelles propriétés, quelles valeurs pour une déclaration CSS ?

- Nombreuses propriétés disponibles en CSS3
- Une majorité de ces propriété possède une liste restreinte de valeurs
- On peut appliquer ces couples *{propriété, valeur}* à de nombreux éléments HTML pour les besoins suivants :
  - choix des couleurs (texte, arrière plan, ...)
  - choix des polices de caractères
  - ...

⇒ Propriétés de texte, puis quelques exemples pour les bordures, liens, ...

`http://www.w3schools.com/cssref/default.asp`

Propriétés de police et texte

Taille de police

font-size : 100% ;

⇒ Taille de police à 100%
Utiliser les % ou *em* comme mesure (pas de *px*, *cm*, *pt* car non redimensionnables)

Famille de police

font-family : arial, sans-serif ;

⇒ Fonte de police en *Arial*, famille *sans serif*
Important de spécifier la famille, car si la fonte n'est pas trouvée sur le sytème/navigateur, remplacement par une fonte de la même famille

Propriétés de police et texte (2)

Couleur de police

color : #000000 ;

⇒ Police de couleur noire (#000000 en code hexa)

Style de fonte

font-style : normal | italic | oblique ;

⇒ Style de police en normal, italique, oblique

Variante de fonte

font-variant : normal | small-caps ;

⇒ Style de police en petites majuscules ou normal

Propriétés de police et texte (3)

Epaisseur de fonte

font-weight : normal | bold | bolder | lighter ;

⇒ Style de police en normal, gras ou très gras ou peu gras

Alignement (de texte, mais aussi de contenu quelconque)

text-align : left | right | center | justify ;

⇒ Alignement à gauche, droite, centré ou justifié

Décoration du texte

text-decoration : none | underline | overline | line-through | blink ;

⇒ Aucune décoration, souligné, surligné, barré, clignotant

Propriétés de police et texte (4)

Transformation de texte

text-transform : none | capitalize | uppercase | lowercase ;

⇒ Aucune transformation, tout en capitalisé, tout en majuscules, tout en minuscules

Hauteur de ligne

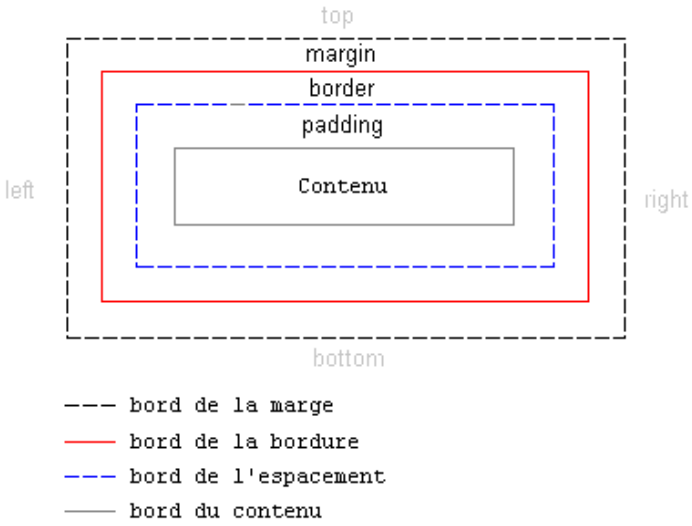
line-height : normal | nombre | % ;

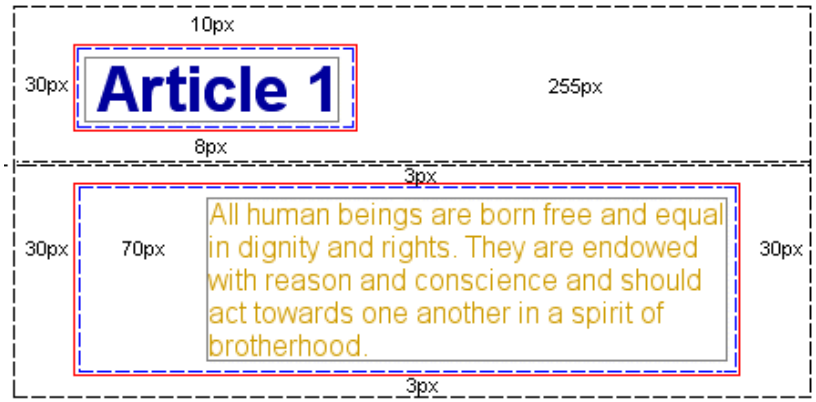
⇒ Hauteur de la ligne, en normal, valeur numérique ou pourcentage

Notion de “boîtes”

Toutes les balises HTML sont contenues dans des “boîtes” :

- Ces boîtes sont invisibles la plupart du temps
- Possibilité de configurer leurs propriétés (bordure, marge, espacements, etc.)





http://fr.html.net/tutorials/css/

Propriétés de bordure

```
<ol style="border-style:dotted; border-width:5px;">
        <li>entrée 1</li>
        <li>entrée 2</li>
        <li>entrée 3</li>
        <li>...</li>
</ol>
```

Que dire sur la bordure ?

Propriétés de bordure

```
<ol style="border-style:dotted; border-width:5px;">
        <li>entrée 1</li>
        <li>entrée 2</li>
        <li>entrée 3</li>
        <li>...</li>
</ol>
```

1. entrée 1
2. entrée 2
3. entrée 3
4. ...

Que dire sur la bordure ?

- Insertion du CSS : inline (par l'attribut `style`)
- Style appliqué à une liste numérotée spécifique
- Deux propriétés de bordure utilisées :
  - style (pointillés)
  - épaisseur (5px)

Propriétés de liens

```
<style type="text/css">
        a:hover {
                color:red;
                text-decoration:none;
                background-color:black;
        }
</style>
```

un lien interne, avec chemin relatif

Quel style pour les liens ?

Propriétés de liens

```
<style type="text/css">
        a:hover {
                color:red;
                text-decoration:none;
                background-color:black;
        }
</style>
```

Quel style pour les liens ?

- Insertion du CSS : interne (balise `<style>`)
- Style appliqué à tous les liens lors de leur survol par la souris (pseudo classe *:hover*)
- Trois propriétés utilisées :
  - couleur du texte (rouge)
  - décoration de texte (aucune)
  - couleur de fond (noir)

Propriétés de liens

```
<style type="text/css">
        a:hover {
                color:red;
                text-decoration:none;
                background-color:black;
        }
</style>
```

Quel style pour les liens ?

- Insertion du CSS : interne (balise `<style>`)
- Style appliqué à tous les liens lors de leur survol par la souris (pseudo classe *:hover*)
- Trois propriétés utilisées :
  - couleur du texte (rouge)
  - décoration de texte (aucune)
  - couleur de fond (noir)

Propriétés d'image de fond

ml ✖ demo1-style.css ✖

```
#tdSpecial {
        color:red;
        background-color:white;
        background-image:url(bluefish.png);
}
```

demo1.html ✖

```
<link href="demo1-style.css" rel="stylesheet" media="all" type="text/css">
</head>
<body>
        <table border="2" width="60%">
        <tr>
        <td id="tdSpecial">case 2<br>qui s'étend sur plusieurs lignes
                <br>grâce aux balises br</td>
        </tr>
        <tr align="right">
                <td>case 4</td>
        </tr>
        </table>
```

Quel style pour la case du tableau ?

Propriétés d'image de fond

ml ✖ | demo1-style.css ✖

```
#tdSpecial {
        color:red;
        background-color:white;
        background-image:url(bluefish.png);
}
```

demo1.html ✖

```
<link href="demo1-style.css" rel="stylesheet" media="all" type="text/css">
</head>
<body>
        <table border="2" width="60%">
        <tr>
        <td id="tdSpecial">case 2<br>qui s'étend sur plusieurs lignes
                <br>grâce aux balises br</td>
        </tr>
        <tr align="right">
                <td>case 4</td>
        </tr>
        </table>
```

Quel style pour la case du tableau ?

- Insertion du CSS : externe (balise `<link>` pour lier la feuille de style à la page HTML)
- Style appliqué à l'élément dont l'attribut `id` vaut *tdSpecial*
- Trois propriétés utilisées :
  - couleur du texte (rouge)
  - image d'arrière plan (bluefish.png)
  - couleur de fond (blanc)

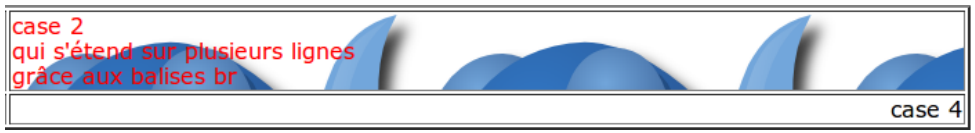
Propriétés d'image de fond

demo1-style.css

```
#tdSpecial {
        color:red;
        background-color:white;
        background-image:url(bluefish.png);
}
```

demo1.html

```
<link href="demo1-style.css" rel="stylesheet" media="all" type="text/css">
</head>
<body>
        <table border="2" width="60%">
        <tr>
        <td id="tdSpecial">case 2<br>qui s'étend sur plusieurs lignes
                <br>grâce aux balises br</td>
        </tr>
        <tr align="right">
                <td>case 4</td>
        </tr>
        </table>
```

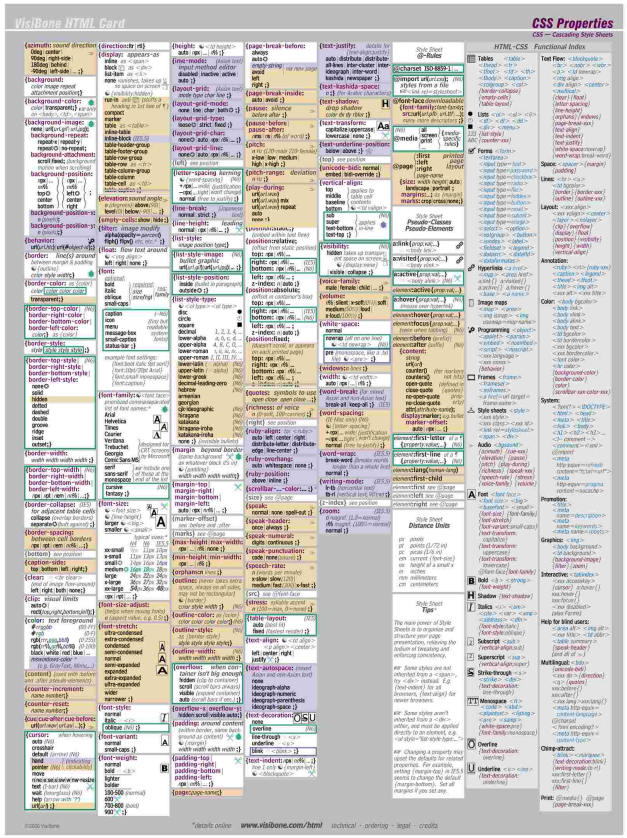


Quel style pour la case du tableau ?

- Insertion du CSS : externe (balise `<link>` pour lier la feuille de style à la page HTML)
- Style appliqué à l'élément dont l'attribut `id` vaut *tdSpecial*
- Trois propriétés utilisées :
  - couleur du texte (rouge)
  - image d'arrière plan (bluefish.png)
  - couleur de fond (blanc)

En résumé

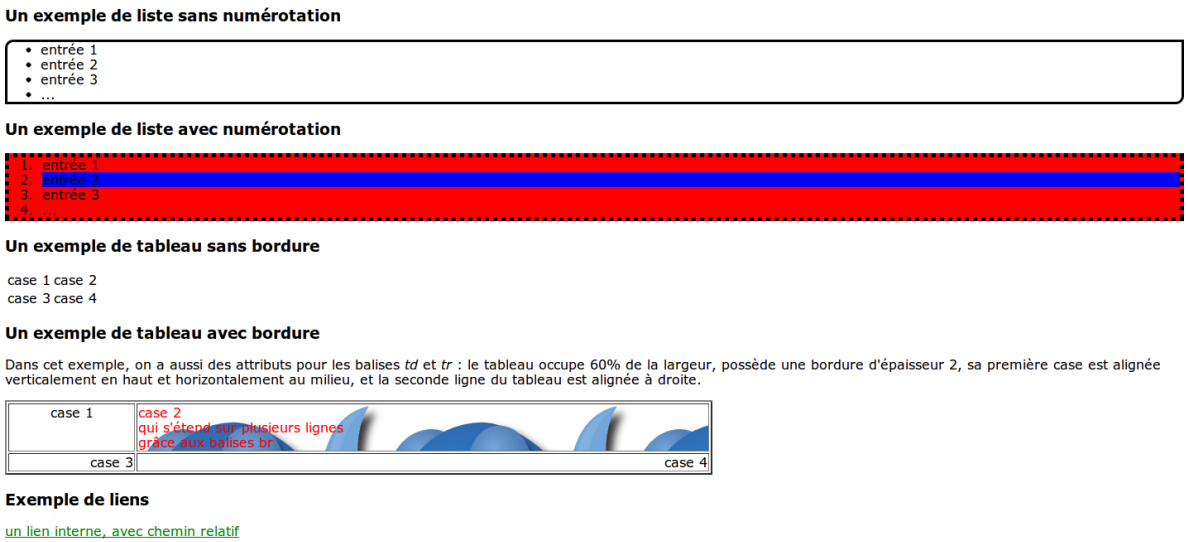
Fiche récapitulative de tous les sélecteurs et propriétés



http://visibone.com/html
http://visibone.com/

En résumé (2)

- Toute la mise en forme peut se faire avec CSS
- Réfléchir à la structure des éléments avant de faire la mise en forme



⇒ Démo avec `demo1-inline.html`, `demo1-interne.html`, `demo1-externe.html` (code source sur la page du LIF4)

Plan du cours

Généralités

Il existe aussi des couples *{propriété, valeur}* pour la mise en page :

- Organisation des différentes parties d'une page (menu, bannière, contenu, pied de page, etc.)
- Positionnement des éléments les uns par rapport aux autres

Rappel sur les types de balises :

- Inline (e.g., `<a>`, `<img>`)
- Block (e.g., `<p>`)

---

http://htmlhelp.com/reference/html40/inline.html
http://htmlhelp.com/reference/html40/block.html

Généralités (2)

Rappel sur les balises neutres et qui servent à regrouper et structurer une page

Syntaxe d'un regroupment

```
<div><p>un texte</p><p>et un second</p></div>
```

⇒ s'utilise pour regrouper des éléments de type bloc ou inline

Syntaxe d'un regroupement

```
<p><span><i>un texte</i></span> et la suite<p>
```

⇒ s'utilise pour regrouper des éléments de type inline

Notion de flux

Flux : ordre dans lequel les éléments HTML sont lus par le navigateur et affichés

- Selon le type d'élément :
  - bloc : éléments les uns en dessous des autres (succession verticale)
  - inline : éléments les uns à côté des autres (succession horizontale)

- Selon les propriétés de style CSS `display`

Il est possible de sortir un élément du flux naturel, dans ce cas, l'espace libéré reste vacant

Notion de flux (2)

Types de rendu pour la propriété `display` :

- block
- inline
- inline-block
- list-item
- table, table-row, table-cell

Les éléments bloc ont par défaut un type de rendu *block*, et les éléments inline ont par défaut un rendu *inline*

Mais un élément peut être “reclassé” grâce à la propriété `display`

`http://openweb.eu.org/articles/initiation_flux/`

Propriété `display`

Type de rendu *block*

display : block ;

⇒ Element placé sous un autre (succession verticale) :

- Occupe toute la largeur de son conteneur
- Redimensionnable avec les propriétés `width`, `height`, `min-width` ou `min-height`
- Peut avoir des marges verticales
- Elements HTML bloc ont par défaut ce type de rendu CSS

Propriété `display` (2)

Type de rendu *inline*

display : inline ;

⇒ Element placé à côté d'un autre (succession horizontale) :

- Largeur déterminée par leur contenu (texte, image, etc.)
- Pas redimensionnables (en théorie)
- Elements HTML inline ont par défaut ce type de rendu CSS

Type de rendu *inline-block*

display : inline-block ;

⇒ Identique au rendu *inline*, mais les éléments sont redimensionnables

- Cas des balises `<input>`

Propriété `display` (3)

Type de rendu *list-item*

display : list-item ;

⇒ Element de type *block* qui bénéficie des propriétés liées aux puces

- Cas de la balise `<li>`

Type de rendu *table, table-row, table-cell*

display : table | table-row | table-cell ;

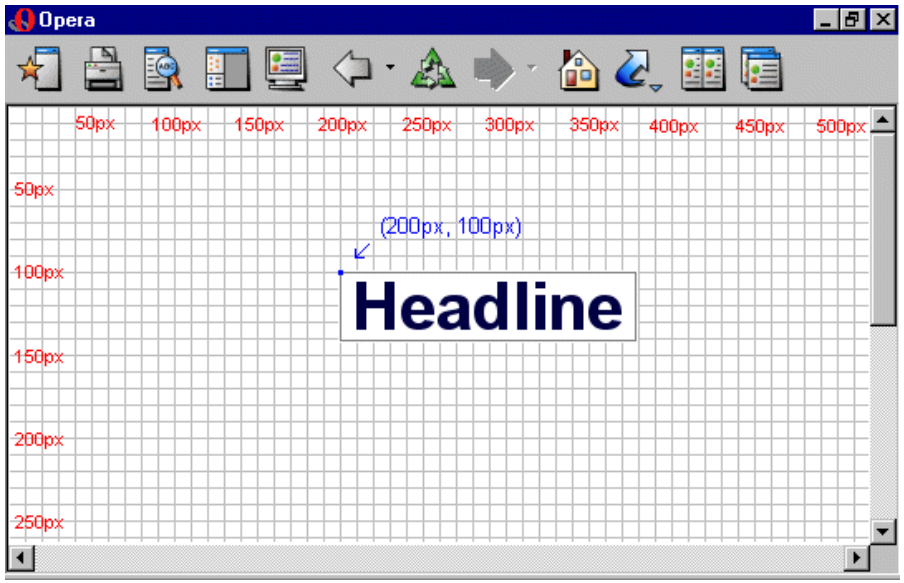
⇒ Même comportement que les balises HTML `table, tr, td`

- Peuvent être utilisées pour la mise en page

Positionnement

On considère la page comme un repère orthonormé :

- Point de coordonnées d'origine (0, 0) en haut à gauche
- Possibilité de placer des éléments en indiquant leurs coordonnées dans ce repère
- Méthode très précise, mais fastidieuse selon le nombre d'éléments



Positionnement absolu

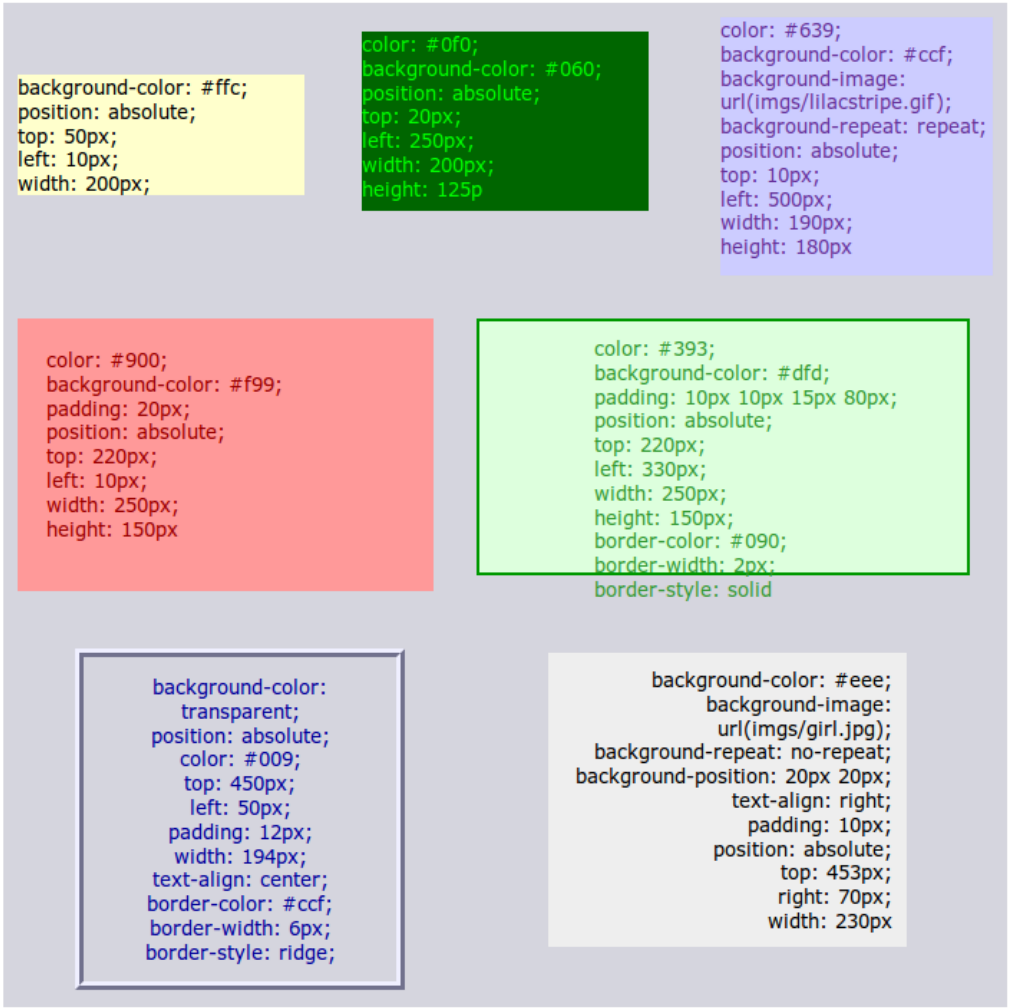
Positionnement absolu

position : absolute ;

⇒ L'élément déclaré en position absolue est totalement retiré du flux :

- Positionnement par rapport au premier ancêtre positionné rencontré (sinon la fenêtre du navigateur)
- Propriétés `top, left, bottom, right` pour indiquer l'écart entre le côté représenté par la propriété et l'élément

Positionnement absolu (2)



http://www.pompage.net/IMG/html/page_modele09.html

Positionnement fixe

Positionnement fixe

position : fixed ;

⇒ Idem que la position absolue, mais :

- Positionnement par rapport à la fenêtre du navigateur suivant un ou plusieurs côtés
- Propriétés `top`, `left`, `bottom`, `right` pour indiquer l'écart entre le côté représenté par la propriété et l'élément
- Même avec une barre de défilement, l'élément fixe gardera toujours la même position dans la page

Positionnement relatif

Positionnement relatif

position : relative ;

⇒ Décale l'élément par rapport à sa position dans le flux

- Les autres éléments considèrent que celui-ci est toujours à sa position initiale dans le flux
- Utile pour servir de référence ancêtre à un élément positionné en absolu
- Utile pour utiliser la superposition d'éléments avec la propriété `z-index`

Propriété `float`

Positionnement flottant

float : left | right ;

⇒ Indiquer qu'un élément "enroule" son voisin à gauche ou à droite

- Effet "d'habillage"
- Largeur dictée par son contenu

Annulation flottant

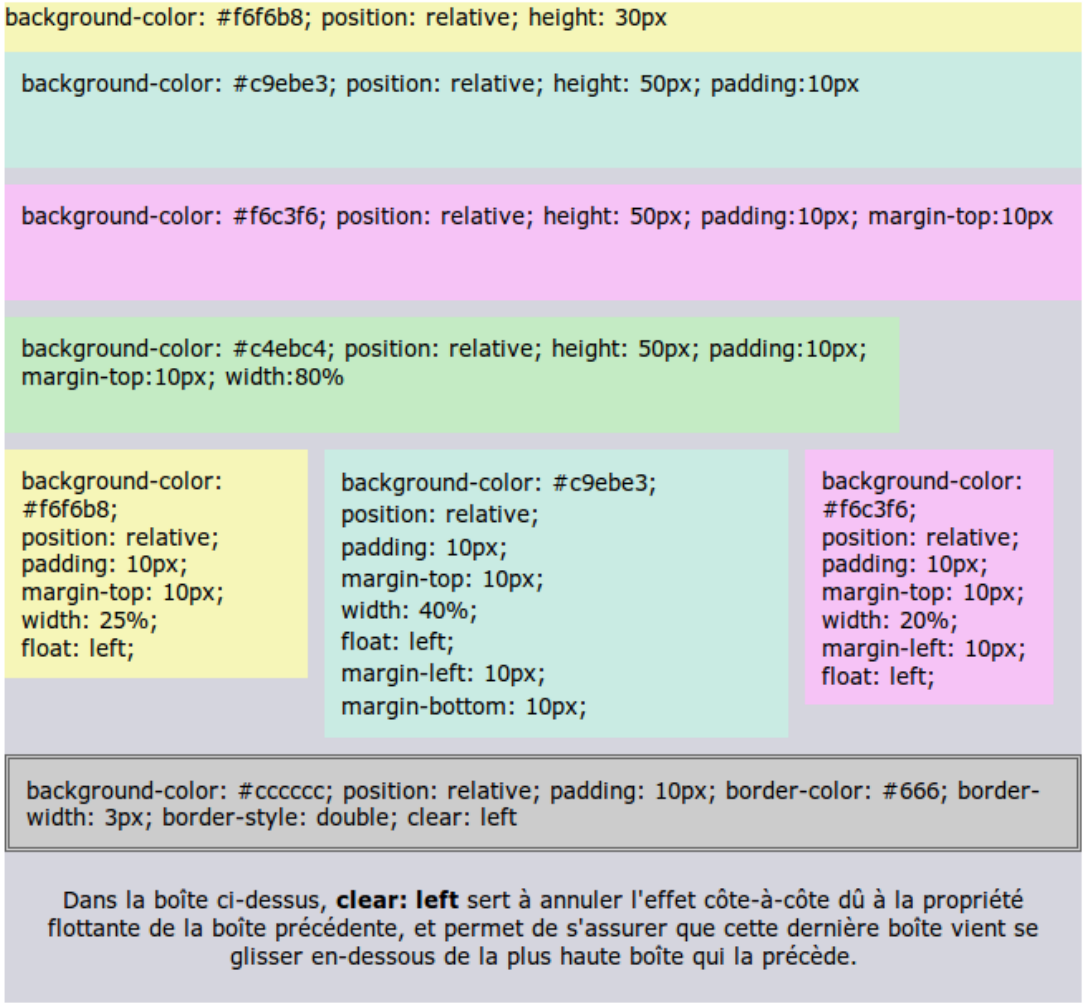
clear : left | right | both ;

⇒ Propriété `clear` pour annuler l'effet côte à côte imposé par un ou des éléments voisins

---

`http://openweb.eu.org/articles/initiation_float/`

Positionnement relatif et flottants



http://www.pompage.net/IMG/html/page_modele10.html

Mise en page par tableaux

Possibilité de faire de la mise en page par tableaux avec CSS :

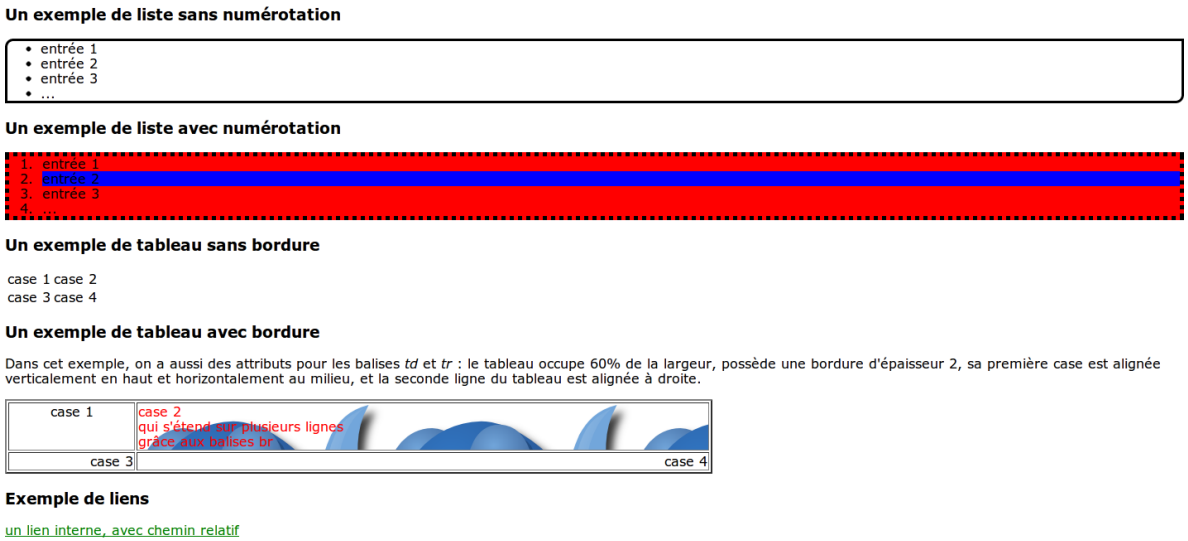
- La déclaration `display: table;` indique que l'élément est un tableau
- La déclaration `display: table-cell;` indique que l'élément est une cellule
- Avantage de ces tableaux émulés : les cellules ont la même hauteur quelque soit le contenu

```
#main {
        display:table;
        margin:auto;
}
#menu  {
        display:table-cell;
        width:240px;
        background-color:#FF3366;
}
#contenu {
        display:table-cell;
        background-color:#9966FF;
}
```

http://css.mammouthland.net/mise-en-page-avec-display-table.php

En résumé

- Positionner les éléments avec CSS avec `display` et en privilégiant selon l'ordre du flux (pas avec des tableaux HTML ou frames)
- Ne pas abuser des balises `div` ou `span`



Un exemple de liste sans numérotation

- entrée 1
- entrée 2
- entrée 3
- ...

Un exemple de liste avec numérotation

1. entrée 1
2. entrée 2
3. entrée 3
4. ...

Un exemple de tableau sans bordure

case 1 case 2
case 3 case 4

Un exemple de tableau avec bordure

Dans cet exemple, on a aussi des attributs pour les balises *td* et *tr* : le tableau occupe 60% de la largeur, possède une bordure d'épaisseur 2, sa première case est alignée verticalement en haut et horizontalement au milieu, et la seconde ligne du tableau est alignée à droite.

case 1	case 2 qui s'étend sur plusieurs lignes grâce aux balises br
case 3	case 4

Exemple de liens

un lien interne, avec chemin relatif

⇒ Démo avec `demo2.html` (code source sur la page du LIF4)

Bilan

Utilisation de CSS pour la mise en forme et la mise en page d'une page Web :

- Utiliser une feuille de style externe
- Sélecteurs pour appliquer une mise en forme commune sur plusieurs pages
- Bien penser à la mise en page (flux, types d'éléments, etc.)

Bilan

Utilisation de CSS pour la mise en forme et la mise en page d'une page Web :

- Utiliser une feuille de style externe
- Sélecteurs pour appliquer une mise en forme commune sur plusieurs pages
- Bien penser à la mise en page (flux, types d'éléments, etc.)

Prochain cours : le langage PHP