

# Mettre en forme des pages Web

## Cascading Style Sheet (CSS)

M1 Architecture de l'Information  
Remise à niveau en informatique  
Lionel Médini



# Objectifs des cours et des TP (partie Web)

- ❑ Découvrir ce qu'est le Web et comment ça marche
  - environnement réseau
  - mécanismes de base du Web
- ❑ Être capable de
  - créer une page Web simple
    - langage XHTML
  - mettre en forme une page ou un ensemble de pages
    - langage CSS niveaux 1 et 2
  - rajouter des contenus dynamiques dans une page
    - scripting côté client
  - créer un site Web basique
    - installation d'un serveur
    - publication de contenus
    - scripting côté serveur

# Position du problème : « styler » des pages Web

## □ Au départ : HTML

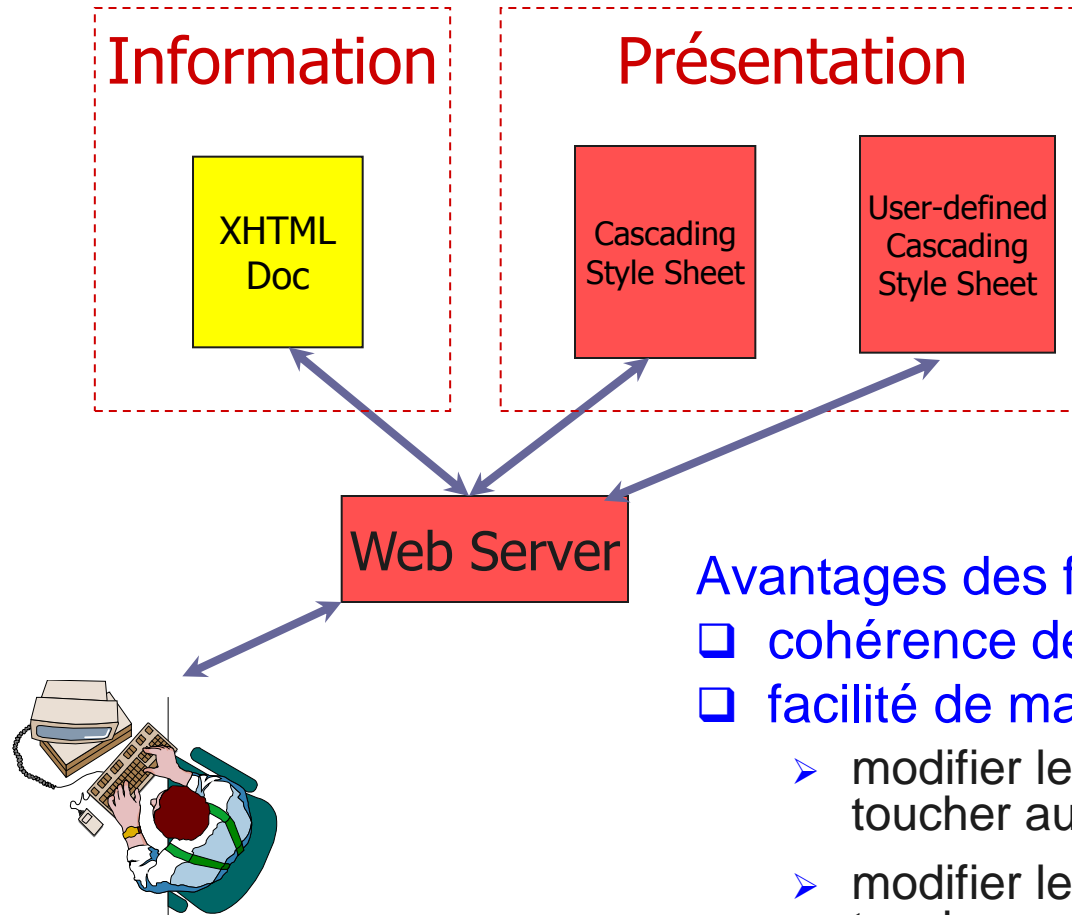
- Rappel : initialement, gestion de la documentation du CERN
  - Pas besoin de styler les pages
- Besoin de mise en forme apparu avec l'essor du Web
  - Au départ : du grand n'importe quoi
    - ❖ Plusieurs façons de faire la même chose
    - ❖ Plusieurs comportements différents pour le même code HTML
  - ➔ Nécessite d'écrire plusieurs versions des pages pour différents navigateurs
- Informations de style dans le code HTML
  - ➔ Nécessite d'indiquer plusieurs fois les informations de style
    - ➔ Difficulté de maintenir une unité graphique cohérente
    - ➔ Mélange du fond et de la forme

# Position du problème : « styler » des pages Web

- **Besoin : mettre en forme des pages Web de façon**
  - Standardisée
  - Réutilisable
  - Séparant clairement le fond et la forme
- **Intervention du W3C**
  - Spécification d'un langage de description de **feuilles de styles** : CSS
  - Avec la spécification XHTML Strict, on se « débarrasse » des éléments et attributs de style hérités de HTML

Remarque : tous les problèmes de compatibilité ne sont pas résolus...

# Présentation de pages Web avec des CSS



## Avantages des feuilles de style

- ❑ cohérence des présentations
- ❑ facilité de maintenance
  - modifier le contenu sans toucher au style
  - modifier le style sans toucher au contenu

# CSS : généralités

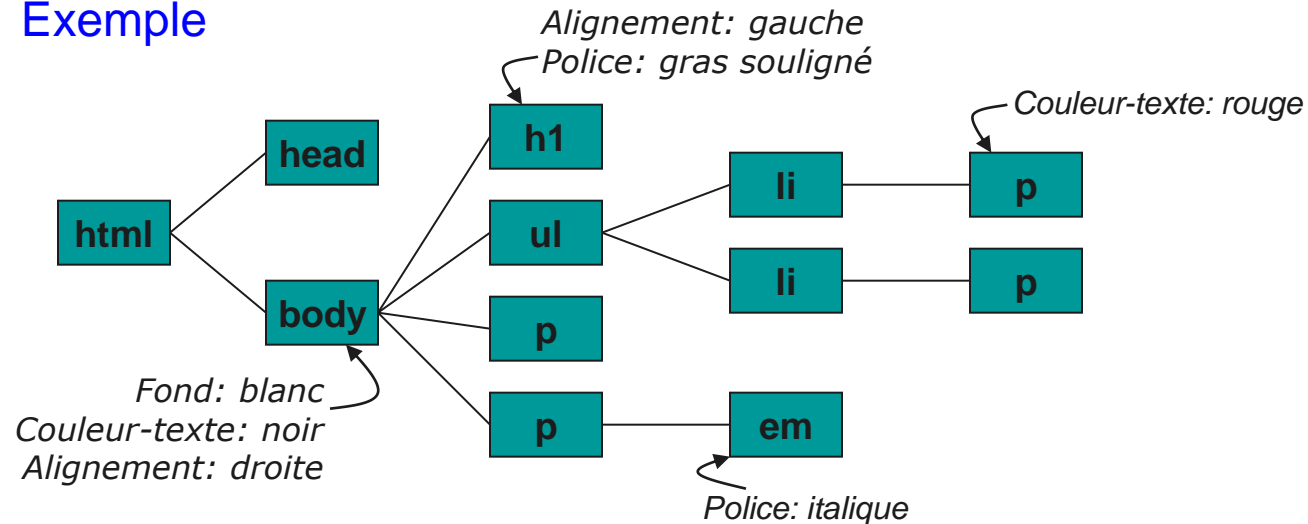
## □ Objectif

- Décrire comment un document XHTML doit être affiché
  - Remplace les éléments d'affichage des anciennes versions de HTML (séparation réelle du contenu et de sa présentation)
  - L'affichage est pris en charge par le navigateur (normalement)

## □ Principe

- Décoration de l'arbre des éléments XHTML
  - Associer des caractéristiques de style à un élément ou un ensemble d'éléments

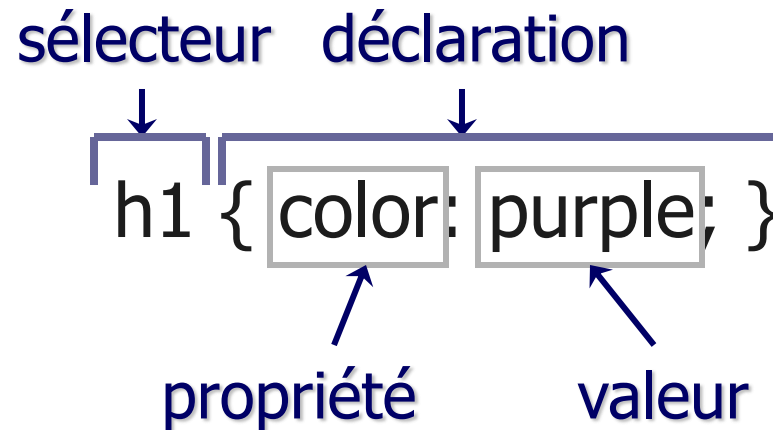
## □ Exemple



# Feuille de style CSS

- ❑ Une feuille de style est composée d'un certain nombre de règles (*rules*)
- ❑ Une règle se compose
  - d'un sélecteur
  - d'une déclaration
- ❑ Une déclaration se compose d'un ensemble de propriétés/valeurs
- ❑ Remarque
  - L'ordre des déclarations est généralement indifférent
  - Tous les styles ne peuvent pas s'appliquer à tous les éléments
    - Ex. : une image n'a pas de style de police

# CSS : structure des règles



```
h1 { font-family: Arial, sans-serif; font-style: italic }
```

séparateur de propriétés/valeurs



# Exemple de règle CSS

```
body {  
    background: #FFFFFF;  
    color: black; /* commentaire */  
    margin-left: 5%;  
    margin-right: 5%;  
    font-family: Tahoma, Optima,  
    Arial, sans-serif;  
}
```

# Styles intégrés

- ❑ Déclarés comme attribut **style** d'un élément

- ❑ Exemple

- `<h1 style="font-family: Arial; font-style: italic;">Un texte qui se retrouvera en arial italique</h1>`

# Feuilles de styles incorporées

- ❑ La feuille de style est déclarée dans l'en-tête (**head**) du document XHTML
- ❑ Elle s'appliquera aux éléments dans **body**
- ❑ Exemple

```
<html>
  <head>
    <title>Texte du titre</title>
    <style type="text/css">
      .important {color: red;}
      h1 {font-family: Arial; font-style: italic; }
      em { color: green }
    </style>
  </head>
  <body>
    <!-- éléments sur lesquels s'appliqueront les styles -->
  </body>
</html>
```

# Feuilles de styles liées

- ❑ Les règles se trouvent dans un fichier extérieur
- ❑ On indique au navigateur où se trouve cette ressource de style, en utilisant un élément **link** dans l'en-tête
  - `<link rel="stylesheet" type="text/css" href="fichier-de-style.css" />`
- ❑ Intérêt
  - Plusieurs documents XHTML peuvent faire appel à la même feuille de style

# Héritage de styles

## □ Par défaut

- Les styles d'un élément sont hérités par ses éléments descendants
- Exemple :
  - `<p style="color: red ;">Du texte <em>mis en évidence</em>, pas mis en évidence.</p>`
  - Du texte *mis en évidence*, pas mis en évidence.

## □ A condition que l'héritage ait un sens

- *i.e.* que les caractéristiques soient applicables à l'élément enfant
  - un positionnement de bloc n'a pas d'intérêt pour un élément `em` qui y est contenu
- si un style est défini spécialement pour un élément (ex. `em { color: blue; }`), l'héritage ne se fait pas
  - Du texte *mis en évidence*, pas mis en évidence.

# Cascading style sheets : cascade

- ❑ On peut avoir concurrence entre plusieurs styles définis dans de multiples endroits
  - styles par défaut (1- navigateur)
  - fichiers CSS externes (2- spécifications globales au site)
  - élément `head` du document XHTML (3- spécification globales au doc.)
  - attributs `style` des éléments (4- spécification locales)
  - style utilisateur (5- spécification de l'utilisateur)
- ❑ Notion de cascade ou ordre de priorité des styles
  - trouver toutes les déclarations qui s'appliquent à un élément
  - les classer par spécificité
    - système de poids qui s'ajoutent
  - les classer par ordre d'apparence
    - plus une déclaration apparaît tard, plus elle a de poids
- ❑ Exemple
  - style (4) > style (4 hérité) > style (3) > style (2) > style (1)

# Types de sélecteurs

- ❑ Simples et groupes
- ❑ Classes
- ❑ Pseudo-classes
- ❑ Pseudo-éléments
- ❑ Contextuels

# Sélecteurs CSS : simple et groupe

## □ Simple

- Lié à un type d'élément HTML
- Utilisation de son nom
- Exemple

- `h1 { text-align: center; }`

## □ Groupe

- Regroupement de règles qui s'appliquent à plusieurs éléments
- Exemple

- `h2, p { font-family: Optima, Arial, sans-serif; }`

## □ Impossibilité de considérer différemment des éléments de même type



# Sélecteurs CSS : id

- ❑ On peut assigner un identifiant à un élément HTML

```
<div id="menu" >
```

- ❑ On peut lui assigner un style par un sélecteur particulier dans la feuille de style

```
#menu { position: fixed; }
```

# Sélecteurs CSS : classes

- ❑ On peut assigner une classe à un élément HTML

```
<h1 class="header" >
```

- ❑ Celle-ci spécifie un sélecteur particulier dans le feuille de style

```
h1.header { text-align: center; }
```

- ❑ Une classe peut s'appliquer à de multiples éléments

```
.header { text-align: center; }
```

s'appliquera aussi à `<h2 class="header">`, *etc.*

# Sélecteurs CSS : pseudo-classes

- ❑ Sélecteurs qui sélectionnent des éléments en fonction de leur état à un moment donné
- ❑ Exemple et intérêt principal
  - `a:link` — lien non visité et inactif
  - `a:hover` — lien sur lequel passe le pointeur de la souris
  - `a:active` — lien sur lequel on clique
  - `a:visited` — lien déjà visité
- ❑ Exemple

```
a:link {color: blue;}  
a:visited {color: magenta;}  
a:hover {color: red; text-decoration:none; font-weight: bold;}  
a:active {color: red;}
```

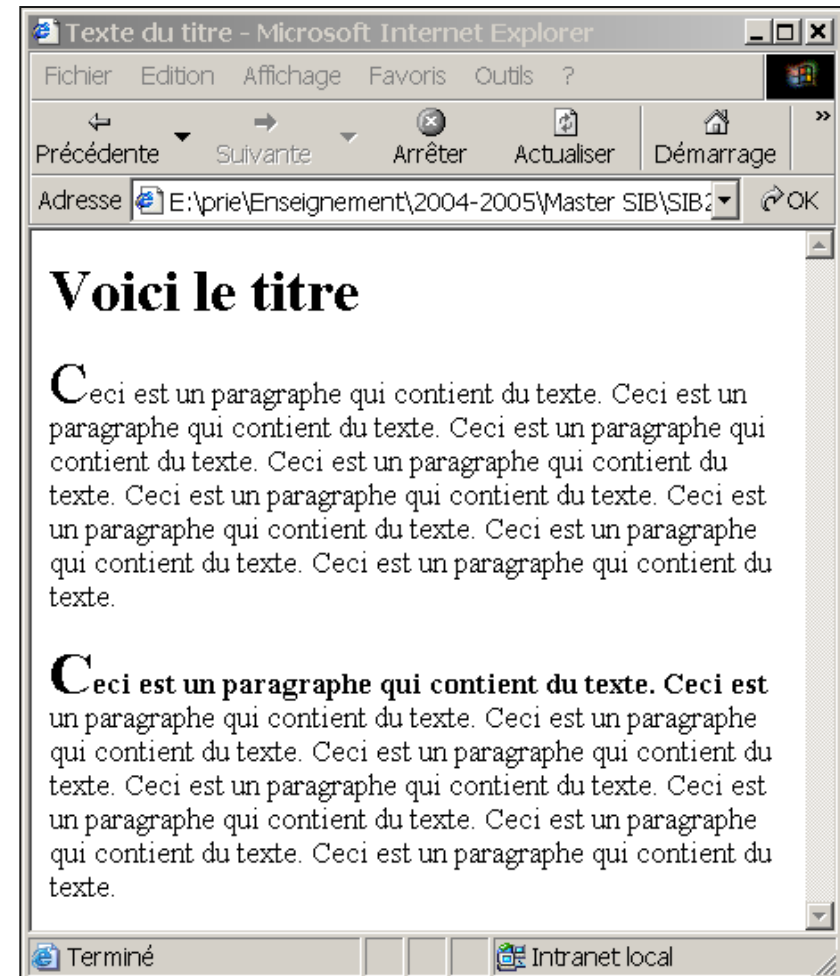
# Sélecteurs CSS : pseudo-éléments

## □ :first-letter

- première lettre dans un élément bloc (ex. p, h1, ...)

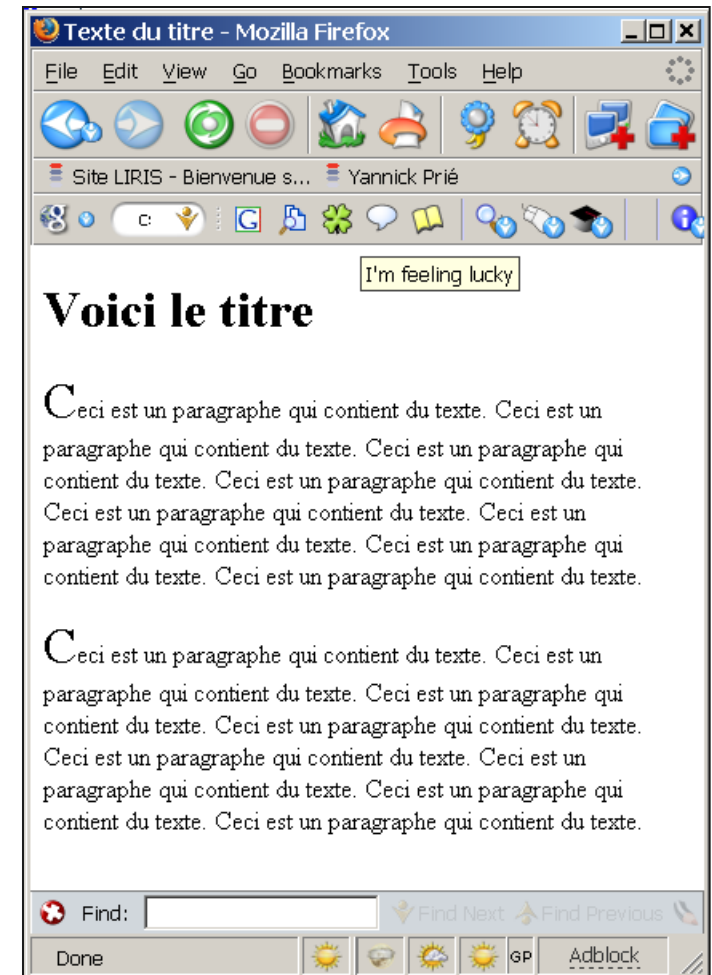
## □ :first-line

- première ligne dans un élément bloc (ex. p, h1, ...)



# Pseudo-éléments : exemple

```
<?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>
<head>
<title>Texte du titre</title>
<style type="text/css">
  p:first-letter {font-size: 200%;}
  p.ligne1:first-line {font: bold;}
</style>
</head>
<body>
<h1>Voici le titre</h1>
<p>Ceci est <!-- ... --> du texte. </p>
<p class="ligne1">Ceci <!-- ... --> qui contient
  du texte. Ceci est un paragraphe qui
  contient du texte. </p>
</body>
</html>
```



# Sélecteurs CSS : contextuels

- ❑ Sélecteurs qui ne sélectionnent que des éléments dans un certain contexte

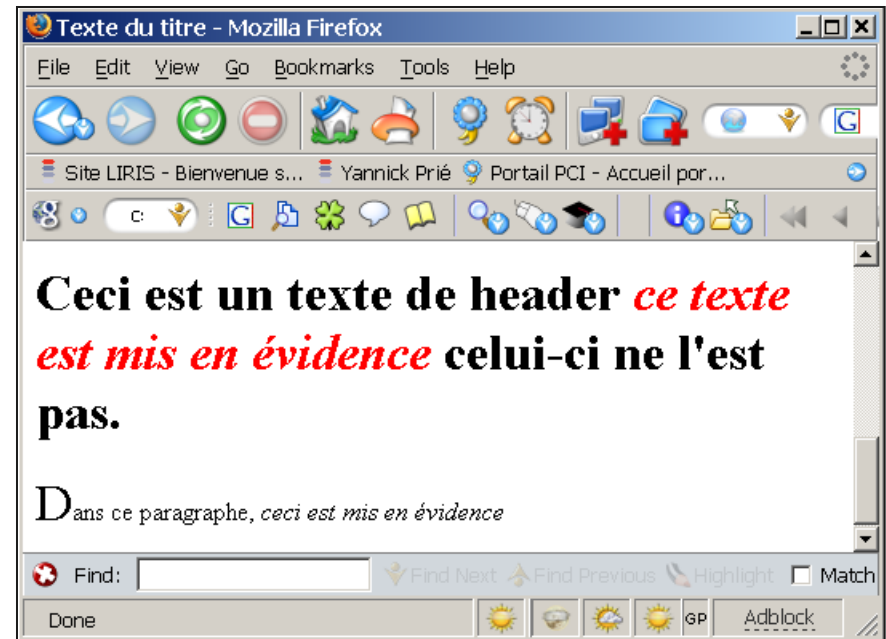
Style :

```
h1 em { color: red; }
```

XHTML :

```
<h1>Ceci est un texte de header <em>ce texte est mis en évidence</em> celui-ci ne l'est pas.</h1>
```

```
<p>Dans ce paragraphe, <em>ceci est mis en évidence</em></p>
```



# Quelques propriétés de textes

- ❑ **font-size:**
  - small | medium... | % | x pt
- ❑ **font-family:**
  - fontname1, fontname2 (*si la première n'est pas disponible*), familyname (*serif, sans-serif, etc.*)
- ❑ **font-weight:**
  - bold | lighter
- ❑ **font-style:**
  - italic, oblique
- ❑ **text-align:**
  - left | center | right | justify
- ❑ **text-indent:** (*retrait de première ligne*)
  - % | x cm

# Les couleurs

## □ color:

- red | blue...| hexcode

## □ background-color:

- red | blue...| hexcode

■ Black = "#000000"

■ Silver = "#C0C0C0"

■ Gray = "#808080"

□ White = "#FFFFFF"

■ Maroon = "#800000"

■ Red = "#FF0000"

■ Purple = "#800080"

■ Fuchsia = "#FF00FF"

■ Green = "#008000"

■ Lime = "#00FF00"

■ Olive = "#808000"

■ Yellow = "#FFFF00"

■ Navy = "#000080"

■ Blue = "#0000FF"

■ Teal = "#008080"

■ Aqua = "#00FFFF"



# Types d'éléments

## □ Éléments blocs

- Prennent la forme d'un bloc dans la page
  - ensemble de lignes
  - ne peuvent être contenus que dans d'autres éléments blocs
- Exemple
  - p, img, ul, table, h1, **div**...
- Propriétés de bloc

## □ Éléments inline

- S'inscrivent dans la continuité des éléments
  - ne forcent pas un changement de ligne
  - peuvent être inclus dans n'importe quel élément
- Exemple
  - a, em, **span**...

## □ Éléments de listes

- éléments HTML qui ont un marqueur (bullet, number) et un ordre

# Rappels sur div et span

## □ div

- élément contenant d'autres éléments, servant à définir un bloc
- servira essentiellement à faire du positionnement de bloc

## □ span

- élément contenant d'autres éléments, inline
- servira essentiellement à regrouper des caractéristiques globales pour des éléments textuels

# Propriétés de styles graphiques

## □ Attribut **clip**

- définit une zone de rognage du bloc
- par exemple pour rogner une image

## □ Attribut **visibility** et **display**

- indiquent si (visible|hidden) et comment un élément est affiché

## □ Attribut **z-index**

- en cas de superposition de blocs d'affichage, indique l'ordre dans lesquels le navigateur doit les afficher (cf. logiciel de dessin)

## □ Remarques

- les distances s'expriment en pixels (**px**), points (**pt**), unités métriques (**cm**, **mm**), ou pourcentages de la taille de la fenêtre (%)

# Propriétés de styles graphiques (2)

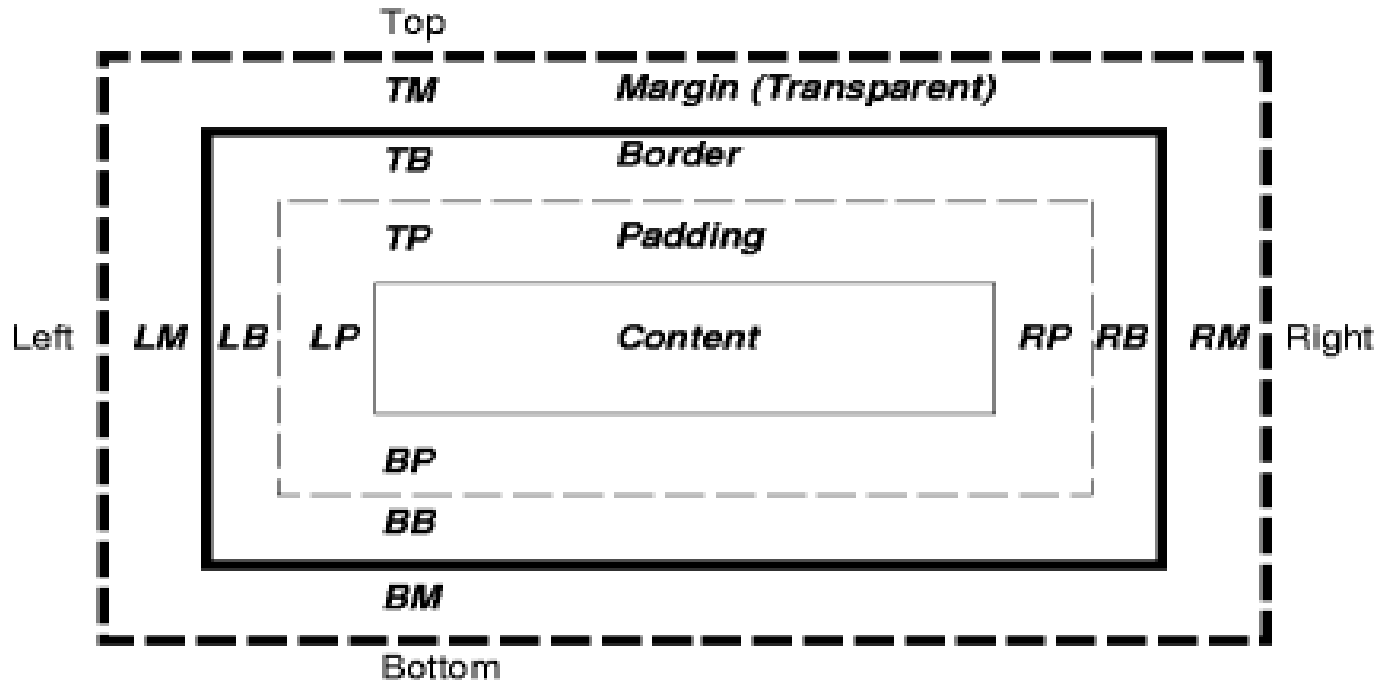
## □ Arrière-plan

- background-color
  - couleur de fond
- background-image
  - image de fond
- background-repeat
  - types de répétition de l'image
- background-attachement
  - arrière-plan fixe quand on utilise l'ascenseur
- background-position
  - position par rapport au coin supérieur gauche
- background
  - attribut de résumé

## □ Exemple

- p { background: image.gif fixed repeat-y }

# Marges, bordures, padding



-  Margin edge
-  Border edge
-  Padding edge
-  Content edge

CSS définit des attributs pour définir précisément ces zones

Source : W3C

# Positionnement des blocs

- ❑ `position: relative`
  - Pas de position particulière : à la suite des autres blocs du bloc le contenant, dans le flot du document
- ❑ `position: fixed`
  - Positionnement par rapport à la page
  - Ne fait plus partie du flot du document
  - Ne défile pas avec la page
- ❑ `position: absolue`
  - Positionnement à l'intérieur du bloc le contenant
  - Ne fait plus partie du flot du document
  - Défile avec la page
- ➔ Utilisation en conjonction avec `top`, `bottom`, `left`, `right`
  - ➔ Attention à ne pas mélanger les unités !
- ➔ Lorsque des blocs « sortent » du flot du document, on peut obtenir un recouvrement (*overlap*) entre les différents éléments
  - ➔ Utiliser la propriété `z-index` pour définir l'ordre d'affichage

À tester sur :

[http://w3schools.com/css/css\\_positioning.asp](http://w3schools.com/css/css_positioning.asp)

# CSS + JavaScript

## ❑ CSS permet de rendre un site beau, mais pas dynamique

### ➤ Nécessite de la programmation

- Scripts déclenchés par des événements utilisateur

### ➔ Appel de fonctions JavaScript dans des attributs XHTML

`onClick, onMouseOver, onMouseOut...`

### ➤ Exemple de code

```
<p onClick='javascript:changerElement("menu");'>abracadabra</p>
```

...

```
<div id="menu">Menu...</div>
```

...

```
function changerElement(mon_element) {  
document.getElementById("mon_element").style.visibility = hidden;  
}
```

Id de l'élément

Nom de la  
caractéristique  
de style

Valeur de la  
caractéristique  
de style

# Quelques astuces

## □ Feuille de style par défaut

- Définit le comportement d'un navigateur en l'absence de style déclaré pour un élément
- Peuvent être hétérogènes d'un navigateur à l'autre
  - ➔ Mises en forme différentes pour les styles non spécifiés

## □ Reset CSS ([Eric Meyer, 2007](#))

- Suppression (écrasement) des styles par défaut
  - Mêmes types et tailles des polices
  - Mise à 0 des marges, bordures
  - ...
- Nécessite de savoir ce que l'on fait
  - Restylage **complet** de tous les éléments utilisés dans les pages
  - En tenant compte du comportement que l'on souhaite en fonction du média
    - Ex. : marges différentes sur écran d'ordinateur, téléphone ou pour impression



## Quelques astuces

- ❑ « Débogage » d'une feuille CSS : le résultat obtenu n'est pas celui que vous souhaitez
  1. Vérifiez la conformité de votre feuille de style : <http://jigsaw.w3.org/css-validator/>
  2. Utilisez l'onglet HTML de la console de votre navigateur pour déterminer d'où vient la caractéristique de style affichée
    - Il est possible de bloquer, modifier, rajouter une caractéristique de style dynamiquement pour tester
  3. Corrigez votre feuille de style et rechargez la page
  4. Si ça ne fonctionne toujours pas, vérifiez que la caractéristique de style est bien prise en charge par le navigateur utilisé
- ❑ Pour tâtonner : <http://w3schools.com/css/>
  - Bouton « Try it yourself »
  - Remarque : fonctionne aussi pour JavaScript...

# Conclusion sur CSS

## ❑ Actuellement : CSS 3

- Recommandation du 29 septembre 2011
- Amélioration des sélecteurs
- Nombreuses nouvelles caractéristiques de style
  - Animations, transitions...
- Media queries

## ❑ Les navigateurs gèrent diversement les feuilles de style !

- Tester avec plusieurs navigateurs

## ❑ Aller voir les références

- <http://www.w3.org/Style/CSS/>

## ❑ Tester, pratiquer

- <http://w3schools.com/css/>