

# CSS

Thierry Lecroq

Université de Rouen  
FRANCE

# Plan

- 1 Généralités sur les CSS
- 2 Les sélecteurs
- 3 Les propriétés
- 4 le dimensionnement et le positionnement

# Cascading Style Sheet

## Mise en forme

- Séparation de la forme et du fond
- HTML décrit le fond
- CSS décrit la forme
- Centralisation de l'aspect visuel
- Validation automatique : <http://jigsaw.w3.org/css-validator>
- On insère du CSS entre :

```
<style type="text/css">  
<!--  
du CSS ici  
-->  
</style>
```

- Ou on lie un fichier CSS avec :  

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

  
Commentaires : entre /\* et \*/

# Format d'une classe CSS

Le CSS est formé d'un ensemble de classes. Une classe s'écrit de cette façon :

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

# Format d'une classe CSS

selecteur peut être :

- un nom de balise : les propriétés s'appliquent à toutes ces balises
- un nom générique (commençant par un point) : les attributs s'appliquent aux balises utilisant `class="selecteur"` (sans le point)
- un mélange des deux, séparés par des virgules : les attributs s'appliquent suivant les deux points précédents

Les propriétés désignent les éléments modifiés (couleur, bordure, fond, marges...)

Les valeurs désignent par quelles valeurs sont remplacées les propriétés désignées.

# Exemple de CSS

## style.css

```
body { font-family : Arial ; }  
p { background-color : #FOCOCO ; border : thin solid black ; }  
.titre { color : yellow ; }
```

# Exemple de CSS

index.xhtml

```
<html>
  <head>
    <link rel="stylesheet" type="text/css" href="style.css"/>
    <title>Exemple CSS</title>
  </head>
  <body>
    <h1 class="titre">Titre en jaune</h1>
    <p>Un paragraphe avec bordure et couleur de fond</p>
  </body>
</html>
```



# Les couleurs

mot-clé : black, blue, brown, cyan, gray, green, pink, purple, red, ...

code hexadécimal : #999999, composantes RGB

fonction `rgb()` : `rgb(r,v,b)` avec  $0 \leq r, v, b \leq 255$

# Plan

- 1 Généralités sur les CSS
- 2 Les sélecteurs**
- 3 Les propriétés
- 4 le dimensionnement et le positionnement

# Les sélecteurs

## Un seul élément

```
p { color : yellow ; background-color : blue ; }
```

## Plusieurs éléments

```
h1, div, p { color : yellow ; background-color : blue ; }  
div { margin : 20px ; }
```

## Le sélecteur universel

```
* { background-color : blue ; }  
p { background-color : gray ; }
```

# Les classes

```
.rouge { color : red ; }
```

```
.rouge { color : red ; }  
div.rouge { color : yellow ; }
```

```
.rouge { color : red ; }  
div.rouge { background-color : blue ; }
```

## Plusieurs classes au même élément

```
*.jaune { color : yellow ; }  
div.jaune { color : green ; }  
.classe1 { color : red ; }  
.classe2 { font-style : italic ; }  
.classe3 { background-color : blue ; }
```

# Plusieurs classes au même élément

## Exemple

```
<h1 class = "jaune">Titre en jaune</h1>
<div class = "classe1">
Texte en rouge
</div>
<div class = "classe1 classe2">
Texte en rouge et en italique
</div>
<div class = "classe1 classe3">
Texte en rouge sur fond bleu
</div>
<div class = "jaune classe2 classe3">
Texte en vert et en italique sur fond bleu
</div>
```

## Sélecteur d'identifiant id

```
div { color : black ; }  
#bleu { color : white ; background-color : blue ; }
```

```
<div>  
Texte en noir  
</div>  
<div id = "bleu">  
Texte en blanc sur fond bleu  
</div>
```

Attention à la casse!!

# Les sélecteurs d'attributs

```
acronym [title] { color : red ; background-color : gray ; }  
* [title] { background-color : yellow ; }  
h2 [title] [id] { background-color : yellow ; }
```

# Les sélecteurs de valeur d'attribut

```
element [attribut1 = "valeur1"] [attribut2 = "valeur2"] ... {  
element [attribut ~= "valeur"] { definition du style ; }
```

# Les sélecteurs contextuels parent-descendant

```
element-parent element-descendant { definition du style ; }  
ul li ol li { background-color : red ; color : blue ; }  
element-parent > element-enfant { definition du style ; }
```

On peut combiner

# Les sélecteurs d'éléments adjacents

```
element1 + element2 { definition du style ; }
```

# Les pseudo-classes applicables aux liens

```
a:link  
a:visited
```

# Les pseudo-classes dynamiques

```
:focus  
:hover  
:active
```

## Pseudo-classes diverses

```
:first-child  
:lang (code) (ex : xml:lang = "code")
```

# Les pseudo-éléments

```
:first-letter  
:first-line  
:before { content : "avant" ; definition du style ; }  
:after { content : "apres" ; definition du style ; }
```

# La déclaration !important

## Gestion des conflits

```
* {color : black !important ; background-color : yellow ; }  
div {color : blue ; background-color : white ; }
```

# Plan

- 1 Généralités sur les CSS
- 2 Les sélecteurs
- 3 Les propriétés
- 4 le dimensionnement et le positionnement

# Les propriétés (1)

- `color` : valeur ; : couleur d'avant-plan
- `background-color` : valeur ; : couleur de fond
- `background-image` : `url(URL)` ; : image de fond

## Les propriétés (2)

- `border-style : style{1,4} ;` :  
style peut prendre les valeurs suivantes :
  - ▶ `none` : pas de bordure
  - ▶ `hidden` : idem sauf cellule de tableau
  - ▶ `dotted` : pointillés courts
  - ▶ `dashed` : tirets longs
  - ▶ `solid` : pleine continue
  - ▶ `double` : 2 traits parallèles continus
  - ▶ `groove` : bordure en creux
  - ▶ `ridge` : bordure en relief
  - ▶ `inset` : bordure en creux dont chaque côté n'a qu'une seule couleur
  - ▶ `outset` : bordure en relief dont chaque côté n'a qu'une seule couleur

## Les propriétés (3)

On peut spécifier 1, 2 ou 4 valeurs

- 1 : 4 côtés
- 2 : la première s'applique aux côtés haut et bas, la deuxième s'applique aux côtés droit et gauche
- 3 : la première s'applique aux côtés haut, la deuxième s'applique aux côtés droit et gauche, la troisième s'applique aux côtés bas
- 4 : haut, droit, bas, gauche

## Les propriétés (4)

- `border-width : width{1,4} ;` :  
width peut prendre les valeurs suivantes :

- ▶ `thin` : fin
- ▶ `medium` : moyen
- ▶ `thick` : épais
- ▶ valeur numérique

On peut spécifier 1, 2 ou 4 valeurs

- ▶ 1 : 4 côtés
  - ▶ 2 : la première s'applique aux côtés haut et bas, la deuxième s'applique aux côtés droit et gauche
  - ▶ 3 : la première s'applique aux côtés haut, la deuxième s'applique aux côtés droit et gauche, la troisième s'applique aux côtés bas
  - ▶ 4 : haut, droit, bas, gauche
- `border-color : couleur{1,4} ;` :

# Les propriétés (5)

- `border : width style couleur ;` :  
`h1 { border : 5px double blue ; }`  
est équivalent à  
`h1 { border-width : 5px ; border-style : double ; border-c`

## Les propriétés (6)

- `margin` : valeur {1,4} ; : marge (top, right, bottom, left)
- `padding` : valeur {1,4} ; : marge

## Les propriétés (7)

- `outline-style` : valeur {1,4} ; : style de contour
- `outline-width` : valeur {1,4} ; : largeur de contour
- `outline-color` : valeur {1,4} ; : couleur de contour
- `outline` : `width color style` ; : contour



## Les propriétés (9)

- `text-transform : none | uppercase | lowercase | capitalize`  
casse
- `text-decoration : none | underline | overline | line-through`  
mise en évidence
- `line-height : normal | valeur | pourcent ;` : interligne
- `font : style variant weight [size/line-height family] ;`

## Les propriétés (10)

- `text-align` : `left` | `center` | `right` | `justify` ; : alignement horizontal
- `letter-spacing` : `normal` | valeur ; : espacement entre les caractères
- `word-spacing` : `normal` | valeur ; : espacement entre les mots

# Plan

- 1 Généralités sur les CSS
- 2 Les sélecteurs
- 3 Les propriétés
- 4 le dimensionnement et le positionnement

# Créer des cadres

```
<!DOCTYPE xhtml PUBLIC  
    "-//W3C//DTD XHTML 1.0 Frameset//EN"  
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
```

# Les éléments `frameset` et `frame`

L'élément `<frameset>` permet la division de la page en différents cadres dont les dimensions sont définies par les attributs `rows` et `cols` :

`rows` : permet de diviser la page en cadres horizontaux et contient une suite de dimensions séparées par des virgules (pixels, pourcentages ou proportions)

`cols` : idem pour un découpage vertical

L'élément `<frameset>` contient les éléments `<frame/>` et `<noframes>`

# Les éléments `frame` et `noframe`

## L'élément `frame`

possède l'attribut `src` pour spécifier l'URL décrivant le contenu du cadre

## L'élément `noframes`

décrit un contenu alternatif au cas où le navigateur client ne gère pas les cadres

## Exemple

```
<frameset rows="100,400,200">
  <frame src="cadre1.xhtml"/>
  <frame src="cadre2.xhtml"/>
  <frame src="cadre3.xhtml"/>
</frameset>
<body>
  <p>Votre navigateur ne supporte pas les cadres.</p>
</body>
</frameset>
```

```
rows="100,400,200"
```

```
rows="100,*,200"
```

```
rows="15%,75%,10%"
```

```
rows="2,5,1"
```

# Communication entre cadres

## Exemple

```
<frameset rows="80,*">
  <frame src="cadre4.xhtml" id="haut"/>
  <frameset rows="16%,*">
    <frame src="cadre5.xhtml" id="gauche"/>
    <frame src="cadre6.xhtml" id="centre"/>
  </frameset>
</frameset>
```

# Communication entre cadres

## Exemple

cadre4.xhtml

```
<!DOCTYPE xhtml PUBLIC
    "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional
.
.
.
<body>
  <ul>
    <li><a href="choix1.xhtml" title="Choix 1" target="centr
    <li><a href="choix2.xhtml" title="Choix 2" target="centr
    <li><a href="choix3.xhtml" title="Choix 3" target="centr
  </ul>
</body>
```

# Créer une mise en page : le dimensionnement et le positionnement

## Le dimensionnement des éléments

- `width` : `<longueur>` | `pourcent` | `auto` | `inherit` ;
- `height` : `<longueur>` | `pourcent` | `auto` | `inherit` ;
- - `overflow` : `visible` | `hidden` | `scroll` | `auto` | `inherit` ;
    - ▶ `visible` : le contenu débordant est affiché
    - ▶ `hidden` : le contenu débordant est caché donc invisible
    - ▶ `scroll` : ascenseurs droit et bas systématiques même sans débordement
    - ▶ `auto` : ascenseurs en cas de débordement
- `min-height` : `<longueur>` | `<pourcent>` | `inherit` ;
- `max-height` : `<longueur>` | `<pourcent>` | `none` | `inherit` ;
- `min-width` : `<longueur>` | `<pourcent>` | `inherit` ;
- `max-width` : `<longueur>` | `<pourcent>` | `none` | `inherit` ;

# Le rendu des éléments



`display : none | inline | block | list-item | table | inline-block`

- ▶ `none` : pas d'affichage
- ▶ `inline` : sur une ligne
- ▶ `block` : bloc (comme `<h1>`, `<p>`, `<div>`, ...)
- ▶ `list-item` : liste (comme `<li>`)

## Exemple

```
li { display : inline ; border : solid 1px black ; }
```

```
<ul>  
  <li>Point 1</li>  
  <li>Point 2</li>  
  <li>Point 3</li>  
</ul>
```

# Le positionnement des éléments

## Le flottement

```
float : left | right | none | inherit
```

## Empêcher le flottement

pour les éléments de bloc

```
clear : none | left | right | both | inherit
```

- `none` : flottement autorisé
- `left` : flottement gauche interdit
- `right` : flottement droit interdit
- `both` : flottements gauche et droit interdits

# Le positionnement relatif

```
position : relative
```

avec

```
left : <longueur> | <pourcent> | auto | inherit
```

```
top : <longueur> | <pourcent> | auto | inherit
```

```
right : <longueur> | <pourcent> | auto | inherit
```

```
bottom : <longueur> | <pourcent> | auto | inherit
```

où <longueur> est positive ou négative

# Le positionnement absolu

```
position : absolu
```

## Exemple

```
div.cadre1 { position : absolute ;  
             border : thin solid black ;  
             left : 0px ;  
             top : 0px ;  
             width : 100% ;  
             height : 20% ; }  
  
div.cadre2 { position : absolute ;  
             border : thin solid black ;  
             left : 0px ;  
             top : 20% ;  
             width : 20% ;  
             height : 100% ; }  
  
div.cadre3 { position : absolute ;  
             border : thin solid black ;  
             left : 20% ;  
             top : 20% ;  
             width : 80% ;  
             height : 80% ; }
```

# Le positionnement fixe

```
position : fixed
```

cas particulier du positionnement absolu

le conteneur n'est pas l'élément parent mais la fenêtre du navigateur

# Visibilité et ordre d'empilement

`visibility` : `visible` | `hidden` | `collapse` | `inherit`  
`collapse` : idem `hidden` pour les cellules de tableaux

`z-index` : `auto` | `<N>` | `inherit`