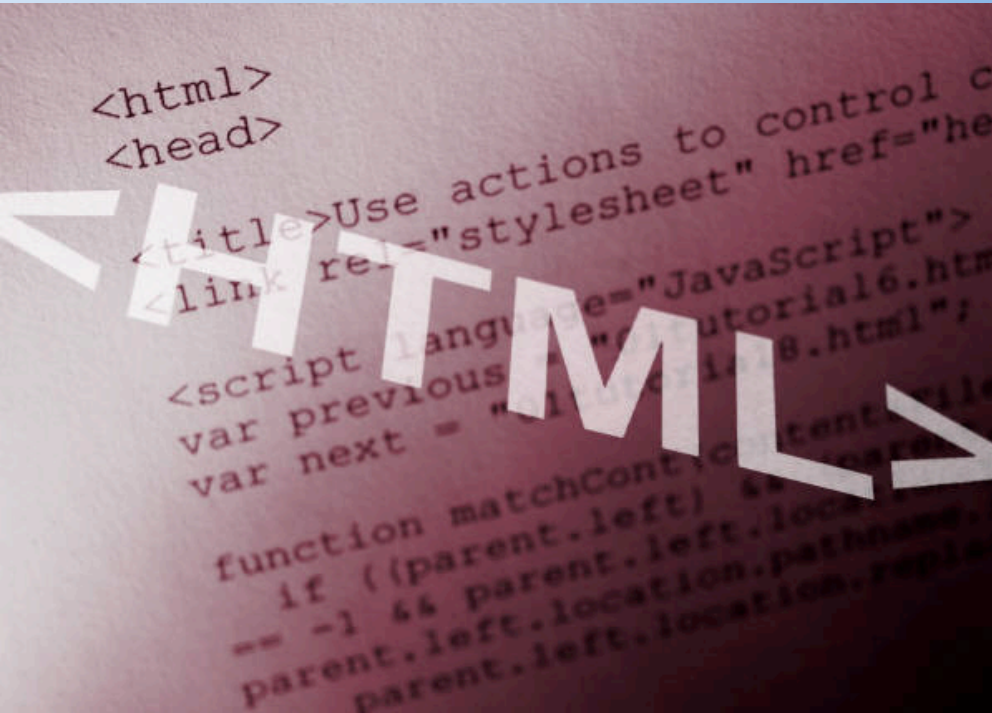


# MODULE

## CONCEPTION DE DOCUMENTS WEB



1. Langages pour l'IHM
2. HTML
3. **CSS**
4. Javascript
5. JQuery

# Les CSS

## Cascading Style Sheets

- **Rôle, usages**
- **Sélecteurs**
- **Syntaxes de regroupement**
- **Intégration dans les documents HTML et XHTML**
- **L'héritage**
- **Présentation de quelques propriétés**
- **Le positionnement**
- **Les types de média**
- **Le principe de cascade**
- **Bonne pratiques**

- **CSS= Cascading Style Sheets**  
= feuilles de style en cascade
  - Ensemble de règles
  - Qui sélectionnent les éléments HTML
  - Qui leur associent des **caractéristiques de style.**
- **4 versions** : CSS 1, CSS 2, CSS 3 et CSS 4, qui est en cours d'élaboration

- **Séparation** du contenu et de la présentation
- **Positionnement** et **style**
- Construction d'une **charte graphique**  
(et d'une cohérence globale)
- Factorisation des règles de style

- Séparation du contenu et de la présentation :
  - **Conserver la lisibilité des documents même avec des navigateurs ne supportant pas les CSS**
  - Permettre leur **consultation avec d'autres supports** (assistants personnels, synthèse vocale, navigateurs braille...)
  - Améliorer l'**accessibilité** des documents
  - **Améliorer l'impression** des documents

- Une **règle** se compose de :
  - Un **sélecteur**
  - Une **déclaration** entre **accolades** { }, composée de propriétés séparées par des ;
    - Une propriété est constituée de
      - Un **mot-clé**, suivi de ":"
      - La (les) **valeur**(s) associée(s)

Exemple (pour appliquer une taille de 18 pixels aux titres de niveau 1), la règle est :

```
h1 { font-size:18px; }
```

- Les commentaires dans les feuilles de style s'écrivent entre `/*` et `*/`

Par exemple :

```
/* commentaires */
```

*Commenter son code est toujours conseillé, notamment lorsque les feuilles de style deviennent longues / complexes*

**Sélecteur** = « lien » entre le document HTML et la feuille de style (ou les feuilles de style).

Permet au CSS d'identifier (sélectionner) le(s) élément(s) au(x)quel(s) appliquer une règle.

3 types simples de sélecteurs :

- les balises
- les classes
- les identificateurs



Toutes les balises HTML peuvent servir de sélecteurs à des règles CSS.

### Exemples

- Pour agir sur les paragraphes

```
p { font-size: 14px; }
```

- Pour agir sur les liens

```
a { font-family: tahoma, arial, verdana; }
```

**Les classes HTML = familles d'éléments** librement définies

```
<p id="p1" class="maclasse">...</p>  
<span id="s2" class="maclasse">...</span>
```

Sélection en CSS : nom de la classe préfixé d'un point

Exemple

```
.maclasse { color: green; }
```

### Les classes peuvent être **restreintes à un élément**

- Le point est précédé de l'élément auquel la classe peut être appliquée
- La classe ne s'applique qu'aux éléments correspondants

```
p.maclasse {color : green;}
```

- La règle s'appliquera à **tout paragraphe** de classe maclasse
- Mais PAS aux autres éléments même s'ils sont de la classe maclasse (div, h1, span, table ou autre)

Les identificateurs ne peuvent porter que sur un seul élément, **unique** et identifié (de fait) dans un document HTML

Le sélecteur d'un identifiant est écrit avec un dièse (#) comme préfixe

```
#monidentificateur { ... }
```

Exemple :

```
#grandtitre {  
    font-weight: bold;  
}
```

Côté HTML :

```
<h1 id="grandtitre">Mon titre</h1>
```

## Quelques remarques en guise de conclusion

### Utiliser des classes ou des identificateurs ?

- Lorsque c'est possible, lorsqu'un **élément est identifié de manière unique** dans un document, on privilégie l'utilisation d'un **identificateur** : le code est ainsi plus facile à lire et à maintenir.
- Les **classes** sont adaptées pour des **éléments génériques**, (éventuellement) appelés à être utilisés plusieurs fois dans un même documents : « types » de paragraphes, éléments redondants (mise en valeur de portions de texte) ...

## 4 façons d'incorporer les directives de style au XHTML.

- Données incorporées dans les balises
- Données incorporées dans l'en-tête du document
- Feuille externe attachée ou liée
- (Feuille de style importée)

# Intégration des CSS dans les documents

## Directives de style dans les balises

### Données incorporées dans les balises

Avec l'attribut style dans le corps du document.

```
<p style="color:red; font-weight:bold;">
```

Rouge gras

```
</p>
```



# Intégration des CSS dans les documents

## Directives de style dans l'en-tête

### Données incorporées dans l'en-tête du document

Avec l'élément `<style>` dans l'en-tête du document.

Les règles s'appliquent alors à l'ensemble du document.

```
<head>
```

```
...
```

```
<style type="text/css">
```

```
    h4 { color:green }
```

```
</style>
```

```
</head>
```

# Intégration des CSS dans les documents

## Feuille externe attachée ou liée

Avec l'élément **<link>** vers une feuille de style externe, placé **dans l'en-tête du document**

```
<head>
```

```
...
```

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

```
</head>
```

```
...
```

Les règles de style se trouvent dans un fichier monstyle.css auquel le document fait appel ; Il est possible et parfois utile de lier plusieurs feuilles de styles successivement.

**Remarque** : en appliquant un style « par défaut » à un élément englobant, dans le cas où ce style n'est pas redéfini, les éléments « enfants » de body, et les « enfants des enfants » hériteront de ce style

```
<div style="color:red; font-weight:bold;">
```

Rouge gras

```
<div>
```

Rouge gras aussi

```
</div>
```

```
</div>
```

La div englobée hérite des propriétés de style

Note : définir un style sur **body** = l'appliquer par défaut à tous les éléments de la page

**NB** : Certaines propriétés de style ne sont pas transmises de l'élément parent à l'élément enfant, c'est le cas de margin, padding (et d'autres propriétés de bloc). Il y a aussi certains bugs de navigateurs anciens

### Positionnement "dans le flux"

- Par défaut, le navigateur construit l'affichage au fur et à mesure que les descriptions des éléments lui parviennent : un bloc est placé en dessous du précédents, les éléments inline sont placés les uns à côté des autres.
- Ces éléments sont alors dits "**dans le flux**", sa position dépend alors de ses propres marges et des marges internes du conteneur et des éléments adjacents.

# Le positionnement

## Positionnement dans le flux

```
div {  
  width: 600px ;  
  padding-top: 20px ;  
  border: solid 1px black;  
}  
p {  
  margin-left: 20px ;  
  margin-bottom: 20px ;  
  width: 300px;  
  border: solid 1px black;  
}
```

```
<div>  
  <p>paragraphe</p>  
  <p>paragraphe</p>  
</div>
```

paragraphe

paragraphe

## Le positionnement

### Positionnement relatif


- Le **positionnement relatif** permet d'inscrire un contenu en flux normal, puis de le décaler horizontalement et/ou verticalement. Le contenu suivant n'est pas affecté par ce déplacement, qui peut donc entraîner des chevauchements.
- Les propriétés **top**, **right**, **left**, **bottom**, permettent de fixer la position relative.

# Le positionnement

## Positionnement relatif

```
<div>
<p id="premier">paragraphe</p>
<p id="second">paragraphe</p>
</div>
```

```
...
div {
    width : 600px ;
    padding-top : 20px ;
    border: solid 1px black;
}
p#premier {
    margin-left : 20px ;
    width: 300px;
    border: solid 1px black;
}
p#second {
    margin-left : 20px ;
    width: 300px;
    border: solid 1px black;
    position: relative;
    left: 4px;
    bottom: 22px;
}
```



paragraphe

paragraphe



# Le positionnement

## Positionnement relatif

Autre exemple : le décalage est relatif à la position normale de l'élément dans le bloc parent

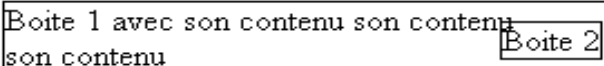
Un paragraphe avec un élément décalé du reste du texte.

```
.decale {  
  position: relative;  
  bottom: 5px;  
  border: solid 1px black;  
}  
...  
<p>Un paragraphe avec  
<span class="decale">un  
&eacute;l&eacute;ment  
d&eacute;cal&eacute; </span>  
du reste du texte.</p>  
...
```

## Positionnement absolu

- La position de l'élément est déterminée de manière **absolue** dans son conteneur parent **positionné** le plus proche, ou à défaut, dans la fenêtre du navigateur
- On utilise la propriété **position**, avec la valeur **absolute**, pour positionner un élément de manière absolue.
- Les propriétés **top**, **right**, **left**, **bottom**, permettent alors de fixer la position.

```
#boite1 {  
  position: relative;  
  width: 300px;  
  border: solid 1px black;  
}  
#boite2 {  
  position: absolute;  
  top: 10px;  
  right: 30px;  
  border: solid 1px black;  
}  
...  
<div id="boite1">  
  <p>Boite 1</p>  
  <div id="boite2">Boite 2</div>  
</div>
```



Boite 1 avec son contenu son contenu  
son contenu

Boite 2

## Le positionnement

### Positionnement fixé

Le **positionnement fixé** est très comparable au positionnement absolu, sauf que l'élément fixé **reste à sa place sur l'écran** même lorsque l'utilisateur fait défiler le contenu.

Un élément fixé est comme « ancré » à sa place.

On utilise la propriété **position**, avec la valeur **fixed**.

Les propriétés **top**, **right**, **left**, **bottom**, permettent alors de définir la position.

# Le positionnement

# Positionnement flottant

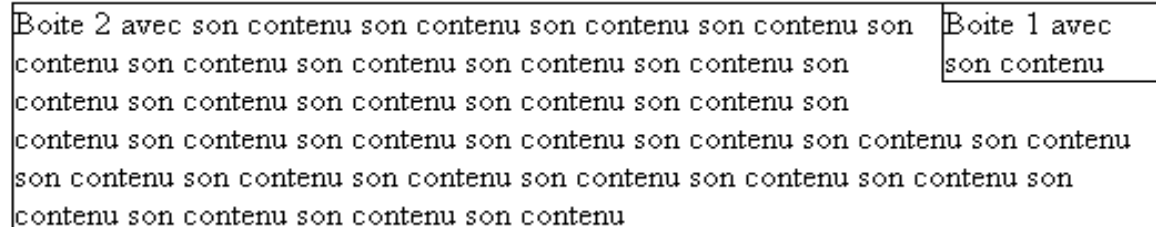
- On utilise la propriété **float**, avec la valeur **left** ou **right**, pour positionner un élément de manière flottante.
- L'élément prend **place à gauche (ou à droite) dans l'élément conteneur**.
- L'élément suivant occupera la **place laissée libre**.
- La boîte 1 se place de manière flottante à droite, la boîte 2 occupe l'espace restant.

```
#boite1 {
    float: right;
    width: 100px;
    border: solid 1px black;
}

#boite2 {
    border: solid 1px black;
}

...

<div id="boite1">Boite 1</div>
<div id="boite2">Boite 2</div>
```

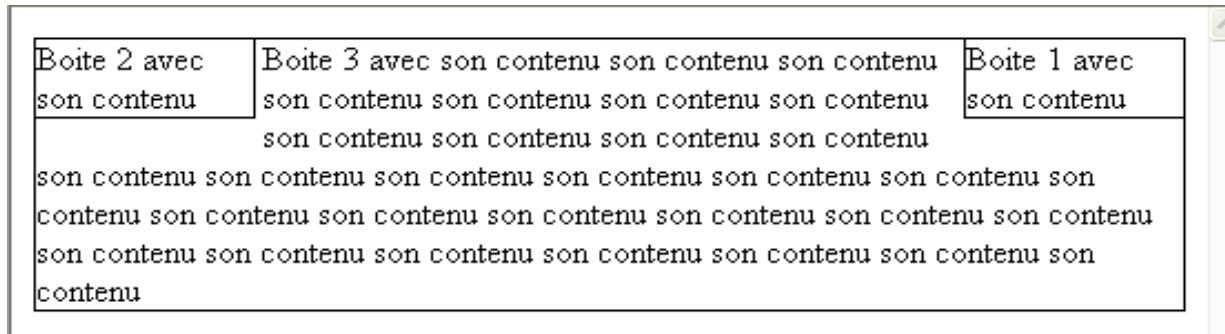


# Le positionnement

## Exemple avec 2 boîtes flottantes

```
<div id="boite1">Boite 1</div>
<div id="boite2">Boite 2</div>
<div id="boite3">Boite 3</div>
```

```
...
#boite1 {
    float: right;
    width: 100px;
    border: solid 1px black;
}
#boite2 {
    float: left;
    width: 100px;
    border: solid 1px black;
}
#boite3 {
    border: solid 1px black;
}
```





## Le positionnement

### Positionnement flottant

- La propriété **clear** permet d'interdire le « voisinage » d'un élément « flottant ».
- Elle accepte 3 valeurs
  - left** : interdit le « voisinage » d'un élément « flottant » à gauche
  - right** : interdit le « voisinage » d'un élément « flottant » à droite
  - both** : interdit le « voisinage » d'un élément « flottant » des deux cotés

## Le positionnement

### Positionnement flottant

```
#boite2 {  
    float: left;  
    width: 100px;  
    border: solid 1px black;  
}  
#boite3 {  
    clear: both;  
}  
<div id="boite2">Boite 2  
</div>  
<div id="boite3">Boite 3  
</div>
```

Boite 2 avec  
son contenu

Boite 3 avec son contenu son  
contenu son contenu son cor  
son contenu son contenu son  
contenu son contenu son cor  
son contenu son contenu son



- Par défaut, le dernier élément (dans l'ordre d'écriture du code) se place au dessus des éléments précédents.
- La propriété **z-index** permet de changer ce placement : dans un même élément conteneur, les éléments ayant un z-index supérieur sont placés au dessus des éléments ayant un z-index inférieur.
- Ceci permet de placer les éléments les uns au dessus des autres.

Les CSS permettent de gérer des **directives de style alternatives pour différents médias** (périphériques de consultation, d'impression, périphériques vocaux et braille...)

## Types de media

### Attribut media

- Lors du lien fait avec une feuille de style l'attribut **media** (balise **<link>**) permet de spécifier un ou plusieurs média(s) visé(s) par la feuille de style.
- Cet attribut est valable aussi lorsque l'on ouvre un **bloc de déclaration de style** dans l'en tête d'un document

```
<link href="default.css" rel="stylesheet" type="text/css" media="screen">  
<link href="default.css" rel="stylesheet" type="text/css" media="print">  
<style type="text/css" media="screen,print">  
...  
</style>
```

Par défaut, si l'attribut média n'est pas spécifié, les directives s'appliquent pour tous les médias (all)

### @media

- Indique un bloc de règles qui ne s'appliquent qu'aux médias spécifiés.
- Cette règle est utilisable dans les feuilles de style liées ou importées, ou dans les directives fixées dans l'en tête des documents XHTML.

```
@media print {  
  body {  
    background-color: #ffffff;  
    color: #000000;  
  }  
}
```

Il est possible d'indiquer une liste de médias en les séparant par des virgules

## Types de media @-règles

•all	Tous médias
•aural	Parole et synthétiseurs de sons
•braille	Interface tactiles braille
•embossed	Impression braille
•handheld	Petits dispositifs ou dispositifs tenus en main
•print	Impression
•projection	Projection
•screen	Ecrans d'ordinateurs
•tty	Teletypes, terminaux...
•tv	TV

## Quelques remarques en guise de conclusion

### Erreurs classiques

- L'élément `div` ne remplace pas tous les éléments : Il vaut mieux privilégier les éléments HTML ayant une valeur sémantique.
- Il n'est pas indiqué de créer des divisions « à outrance » dans les documents.
- Tous les éléments n'ont pas besoin de « `class` ». Multiplier les classes et par conséquent les attributs d'éléments HTML (pour les rattacher à une classe) rend le code beaucoup plus lourd. Il est préférable d'utiliser la sélection hiérarchique plutôt que de vouloir « typer » tous les éléments par des classes.

## Quelques remarques en guise de conclusion

### Règles d'écriture, code réutilisable

Ne pas négliger les commentaires pour , par exemple :

Donner en introduction des informations sur l'auteur, les  
évolution de la feuilles de style, la todo list...  
(voir <http://cssdoc.net/> par exemple)

Proposer un sommaire de la feuille de style (ses sections)

Normaliser les codes couleur utilisés

Diviser la feuilles de style en sections et sous-sections clairement  
visibles et identifiées