Module:
Développement Web

Mohamed CHERRADI

UAE / ENSAH

Formateur Web

m.cherradi@uae.ac.ma

Chapitre 3
CSS & Bootstrap

Plan

Introduction : Style de page web

Les différentes manières d'intégration de css

...

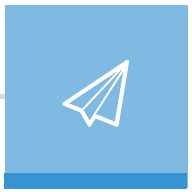
Les sélecteurs CSS

Les différentes méthodes de sélection des éléments HTML

...

Les propriétés CSS

Mise en forme du texte
Les unités de mesure
Propriétés des boîtes, d'affichage, de positionnement, …
Les grilles en CSS, Responsive design (Media Queries)
…



Bootstrap

Les formulaires, les boutons, les tableaux
Les images,
SweetAlert, ChartJS, D3JS, …
…

Introduction

Style des pages web

Introduction

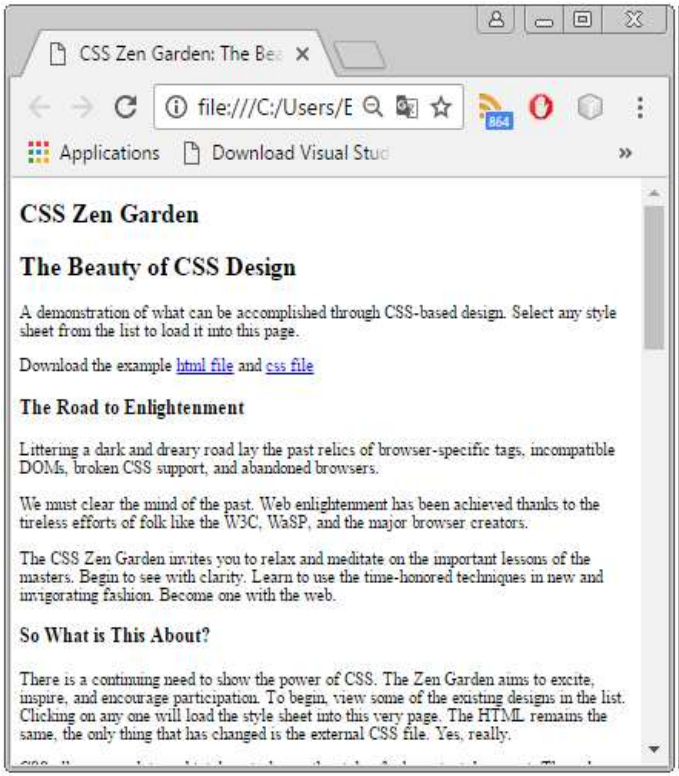
- **C**ascading **S**tyle **S**heets (**CSS**) est un langage de feuille de style utilisé pour décrire la présentation d'un document écrit en HTML ou en XML.
- HTML définit uniquement la structure du contenu, c'est le langage CSS, qui détermine l'apparence de ce contenu.
- CSS décrit la façon dont les éléments doivent être affichés à l'écran. C'est grâce à ce langage qu'on peut :
  - Choisir la couleur et la taille de texte (text)
  - Sélectionner la police de caractère (font)
  - Définir les bordures (border), le fond
  - L'emplacement des éléments du document
  - Etc
- Ce langage est officiellement spécifié (ou défini) par le W3C. Il existe Trois versions: CSS1 (1996), CSS2 et CSS3.

Introduction

- L’un des objectifs principaux des CSS3, est la séparation de la présentation du contenu de la page web (la structure)
- Le code CSS consiste en un ensemble de règles qui définissent le formatage des balises HTML
- CSS fait appel à des sélecteurs qui permettent d'appliquer des styles aux différents éléments HTML.
- Un sélecteur CSS est la partie de la règle CSS qui désigne les éléments d'un document ciblés par cette règle
- Si on veut ajouter un commentaire dans le code CSS on doit le mettre entre les symboles /* et */
- Exemple:
  - **/* Ceci est un commentaire */**

Exemple

- Source : http://www.csszengarden.com/



HTML sans CSS

HTML + CSS

Principe de fonctionnement

- Les navigateurs web appliquent des règles CSS à un document afin que ceux-ci soient affichés correctement.
- Une règle CSS est une liste de propriétés associées à un sélecteur. Une liste de règles de CSS est contenue dans une feuille de style qui définit la mise en forme d'une page.
- Exemple:

- Quand plusieurs règles sont mises en œuvre, celle qui est spécifique a la priorité. D'où, ce langage est nommé langage de feuille de style en **cascade**.
- La notion de « **cascade** » fait référence aux règles de priorité qui existent entre les différents sélecteur

Principe de fonctionnement

- La cascade est un processus hiérarchique:
  - C - Specificité des sélecteurs (Specificity): Les sélecteurs **plus spécifiques** l'emportent sur les sélecteurs **moins spécifiques**. Par exemple, un style défini pour un ID (#exemple) aura une spécificité plus élevée qu'un style défini pour une classe (.exemple)
  - A - L'importance des déclarations (!important): Si une déclaration est marquée avec la propriété !important, elle aura la priorité sur d'autres déclarations, même si elles sont plus spécifiques. Cependant, l'usage excessif de !important est généralement **découragé**, car il peut rendre le code difficile à maintenir.
  - S - Ordre de la feuille de style (Source Order): Les styles déclarés en dernier dans les feuilles de style l'emportent sur les styles précédents, tout simplement parce qu'ils sont les derniers à être lus par le navigateur.
- En résumé, la cascade en CSS est un mécanisme qui détermine quelles règles de style seront appliquées à un élément en fonction de la spécificité des sélecteurs, de l'importance des déclarations et de l'ordre des règles dans la feuille de style. Cela permet une gestion efficace des styles et une adaptation précise à différents éléments de la page.

Syntaxe CSS

- CSS est un langage déclaratif dont la syntaxe est plutôt simple et directe.
- La syntaxe générale pour définir une règle CSS est donnée comme suit :

```
Sélecteur {
        propriété1: valeurX;
        propriété2: valeurY
}
```

  - *Propriété* : est un identifiant de style permettant de définir certains fonctionnalité, comme la couleur, style d'écriture, la taille, etc …
  - *Valeur* : décrit comment la fonctionnalité doit être utilisée par le navigateur.
- Exemple : « color : red; ». Dans cet exemple on associe à la propriété « color » la couleur rouge « red ».
- Cette opération d'association s'appelle « *déclaration CSS* ». Les déclarations CSS sont placées dans des *blocs* de déclarations. Enfin, les *blocs* sont associés à des *sélecteurs* afin de créer des *règles CSS*.
- **Exemple** : p { color: red; }, cette règle permet d'appliquer la couleur rouge sur les paragraphes entourés de la balise <p>.

Liaison HTML - CSS

- Il est possible d'écrire le code CSS directement dans les balises <style>, mais il est plus adapté de créer des fichiers .css séparés de façon à réutiliser les informations de mise en forme sur d'autres pages.
- Il existe trois différents endroits où il est possible de mettre du code CSS:
  - **Méthode 1**: directement dans l'en-tête <head> du fichier HTML. Cela consiste à insérer le code CSS directement dans une balise <style> à l'intérieur de l'en-tête <head>. Un exemple :

```
<head>
    <style> p { color: red; } </style>
</head>
```

  - **Méthode 2**: directement dans les balises du fichier HTML via un attribut style (méthode la moins recommandée). Un exemple :

```
<body>
   <p style="color: blue;">Bonjour et bienvenue sur ce site !</p>
</body>
```

Liaison HTML - CSS

- **Méthode 3**: dans un fichier séparé portant l'extension .css. *Il s'agit de la méthode la plus recommandée.* Les avantages de cette méthode sont :
  - *Eviter la redondance* : le style défini dans un fichier de façon séparé peut être appliqué sur plusieurs document html, par contre, pour les deux méthodes précédentes le style concerne que le document sur lequel le style est appliqué.
  - *Eviter de tout mélanger dans un même fichier*
- La liaison d'un document html avec un fichier .css se fait grâce à l'attribut href de l'élément <link>. Exemple : application de la couleur rouge sur les paragraphes d'un document(nommé test.html). A noter que le fichier du style (nommé style.css) et le document se trouvent dans le même dossier.

```html
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8" />
<link rel="stylesheet" type="text/css" href="style.css">
<title>Style CSS </title>
</head>
<body>
  <h1>Pour tester CSS</h1>
  <p>Bonjour et bienvenue sur ce site !</p>
</body>
</html>
```

test.html

```css
p{
    color: red;
}
```

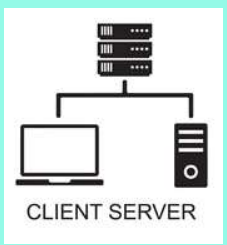
style.css

Les **Classes** et les **ID**

- **Les classes** servent à donner plusieurs styles à **la même balise**
- Exemple: on veut que tous les paragraphes de classes « introduction » soient écrites en bleu.
  - Code HTML : Bonjour !
  - Code CSS : p.introduction {color: blue;} ou .introduction {color: blue;}

- Les ID fonctionnent exactement de la même manière que les classes sauf qu'un ID doit obligatoirement être **unique** dans une page HTML
- Exemple: on veut que tous les paragraphes de classes « introduction » soient écrites en bleu.
  - Code HTML :
  - Code CSS : #logo{ /* votre style css */}

Les sélecteurs CSS

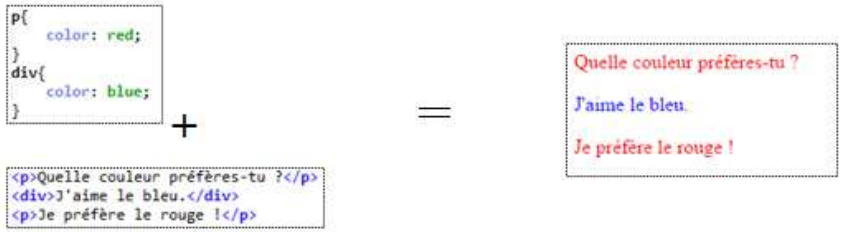


Présentation

- Le rôle d'un sélecteur est de localiser un élément HTML afin de lui attribuer des règles de style.
- Il existe une multitude de méthodes permettant de cibler et de localiser une zone. On distingue 3 types de sélecteurs :
  - Les sélecteurs simples :
    - En fonction du nom d'un élément,
    - En fonction de la valeur d'un attribut d'un élément, en fonction d'une pseudo-classe,
  - Les sélecteurs combinés :
    - En fonction de la hiérarchie d'un élément,
    - En fonction de la position d'un élément.
  - Les sélecteurs multiples:

Les sélecteurs de type (type selectors)

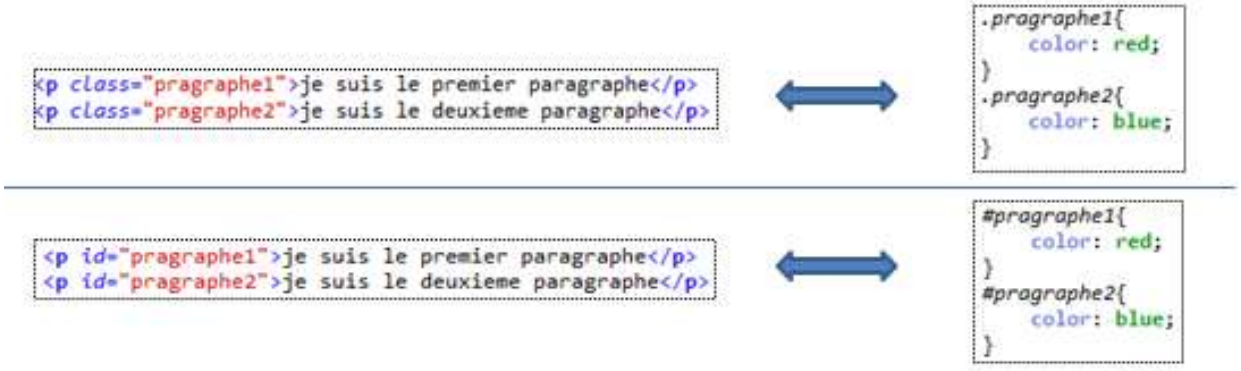
- Aussi appelés sélecteurs d'élément, ces sélecteurs correspondant aux éléments HTML (exemple <p>).
- C'est la méthode la plus simple pour cibler tous les éléments d'un type donné. Prenons un exemple :



- Son inconvénient est que 'par exemple' tous les paragraphes possèdent la même présentation ici, ils seront donc tous écrits en rouge.
- La question qui se pose : est comment faire pour que certains paragraphes seulement soient écrits d'une manière différente.

Les sélecteurs d'identifiant et de classe

- Pour résoudre le problème liée à l'utilisation de nom de l'élément html, on peut utiliser deux attributs spéciaux *qui fonctionnent sur toutes les balises* :
  - L'attribut « class » : ceci permet de définir un sélecteur de classe. Ce dernier est composé d'un point (.), suivi d'un nom de classe.
  - L'attribut « id » : ceci permet de définir un sélecteur d'identifiant. Ce dernier commence par un dièse (#), suivi par le nom de l'identifiant attribué à un élément.
- Exemple :



Les sélecteurs d'attributs et leur valeur

- Les sélecteurs d'attribut permettent de sélectionner les éléments HTML en fonction de noms de leurs attributs et des valeurs de ceux-ci.
- Pour utiliser ces sélecteurs, on écrira des crochets "[ ]" dans lesquels on place le nom de l'attribut et éventuellement une condition sur la valeur de l'attribut. Les sélecteurs d'attributs peuvent être classés en **deux catégories** :
  - Les sélecteurs d'attribut avec ou sans valeur :
    - **[attr]**: sélectionne tous les éléments avec l'attribut attr, quelque soit sa valeur.
    - **[attr=val]**: sélectionne tous les éléments avec l'attribut attr, mais seulement si la valeur est égale à val.
    - **[attr~=val]**: sélectionne tous les éléments dont la valeur de l'attribut attr est une liste de mots séparés par des espaces, dont l'un est exactement "val".

Les sélecteurs d'attributs et leur valeur

- Les sélecteurs d'attribut utilisant un filtre sur les fragments de chaînes:
  - **[attr^=val]**: sélectionne tous les éléments dont la valeur de l'attribut attr commence par val.
  - **[attr*=val]**: sélectionne tous les éléments dont la valeur de l'attribut attr contient la chaîne val.
  - **[attr$=val]** : sélectionne tous les éléments dont la valeur de l'attribut attr finit avec val.
  - **[attr|=val]**: sélectionne tous les éléments dont l'attribut attr vaut val ou commence par val suivi de –
    - Exemple : <a href=" " hreflang="en-US">

Exemple :

```
a[title]
{
/* Sélectionne tous les liens <a> qui possèdent un attribut title.*/
}

a[title="Cliquez ici"]
{
 /* dans ce cas l'attribut doit en plus avoir exactement pour valeur « Cliquez ici ».*/
}
```

```
Les modules de deuxieme semestre :
<ul>
<li module-obligatoire="Développement">Programmation Orientée Objet en C++ </li>
<li module-obligatoire="Mathématiques et Informatique">Algorithmique avancée et complexité </li>
<li module-obligatoire>Modélisation avec UML </li>
<li module-obligatoire="Mathématiques et Informatique">Recherche Opérationnelle</li>
<li module-obligatoire="Développement">Développement d'applications Web</li>
<li module-optionnel>Techniques et économie de l'entreprise </li>
</ul>
```

+

```
[module-obligatoire] {
  color: green
}
[module-obligatoire=Développement ]{
  color: goldenrod;
}
[module-obligatoire~=Informatique] {
  color: red;
}
```

=

Les modules de deuxième semestre :

- Programmation Orientée Objet en C++
- Algorithmique avancée et complexité
- Modélisation avec UML
- Recherche Opérationnelle
- Développement d'applications Web
- Techniques et économie de l'entreprise

Sélecteur universel

- Le sélecteur universel, représenté par *, est le plus large. Il permet de sélectionner tous les éléments d'une page. Il est rarement utile d'appliquer une même mise en forme sur toute une page. Ce sélecteur est donc généralement utilisé avec d'autres sélecteurs (voir *les mécanismes de combinaison* ci-après).
- Exemple :

```html
<h1>Pour tester CSS</h1>
<p class="paragraphe1"> Je suis le premier paragraphe</p>
<p class="paragraphe2"> Je suis le deuxieme paragraphe</p>
<p class="paragraphe3"> Je suis le troisième paragraphe</p>
```

+

```css
*{
    color:red;
}
.paragraphe1{
    color:blue;
}
.paragraphe2{
    font-weight: bold
}
p{
    text-decoration: underline;
}
```

=

Sélecteur et pseudo-classes

- Une pseudo-classe est utilisée pour définir un état spécial d'un élément. Le style s'applique en fonction de l'état de l'élément.
- Par exemple, il peut être utilisé pour appliquer de style quand un utilisateur passe la souris sur l'élément.
- Ce sont des mots-clés précédés par deux points (**:**) et qui sont ajoutés aux sélecteurs. ***La syntaxe est*** :

```
selector:pseudo-class {
    property:value;
}
```

Sélecteur et pseudo-classes

- Voici une liste non-exhaustive :
  - *:hover* : sélectionne l'élément avec la souris en dessus.
  - *:active* : sélectionne l'élément actif
  - *:visited* : sélectionne tous les liens visités
  - *:first-child* : sélectionne l'élément qui est le premier enfant de son parent. Il existe aussi, :last-child, :only-child, :nth-child(n) . Ce dernier peut être utilisé de plusieurs façon en changeant a et b dans nth-child(an+b).
  - *:not(selector)* : Sélectionne tous les éléments sauf celui entre ().

Exemples

- Exemple 1: le code suivant permet de changer la couleur d'un élément lorsqu'on passe la souris dessus.

```
div {
    background-color: green;
    color: white;
    padding: 25px;
    text-align: center;
}
div:hover {
    background-color: blue;
}
```

Couleur initiale

Souris dessus

- Exemple 2:

```
p:first-child {
    color: blue;
}
```

+

```
<p>paragraphe 1 </p>
<p>paragraphe 2 </p>
```

=

paragraphe 1
paragraphe 2

- Exemple 3:

```
p:nth-child(2n+1){
    color: blue;
}
```

+

```
<p>paragraphe 1 </p>
<p>paragraphe 2 </p>
<p>paragraphe 3 </p>
<p>paragraphe 4 </p>
```

=

paragraphe 1
paragraphe 2
paragraphe 3
paragraphe 4

Les pseudo-éléments

- Les pseudo-éléments ressemblent beaucoup aux pseudo-classes : ce sont des mots-clés précédés par **deux** deux-points (**::**) et qui sont ajoutés aux sélecteurs. Ils permettent de représenter des parties.
- On peut voir cela comme un « emplacement » par rapport à l'élément sélectionné : *::after*, *::before*, *::first-letter*, *::first-line*, *::selection* ...

```
p.intro::first-letter {
    color: red;
    font-size:200%;
}
```

```
<p class="intro"> Cette introduction ...</p>
```

Cette introduction ...

```
a {
  color: red;
  font-weight: bold;
  text-decoration: none;
}
a:hover{
  color: black;
}

[href*=ensah]::after {
  content: '-->';
}
```

```
<a href="http://www.ensah.ma">cliquer </a>
```

cliquer -->

```
p::selection {
    color: red;
    background: yellow;
}
```

Les combinateurs

- CSS donne la possibilité de combiner les sélecteurs pour obtenir un résultat précis. Selon les relations entre les éléments, CSS permet de combiner les sélections. Ces relations sont exprimées sous la forme « combinateurs ».
- Exemple : h3+p{ }, le style concerne la première balise <p> située après un titre <h3>.
- Dans le tableau qui suit, A et B représentent n'importe quel sélecteur :

	Combinateur	**Élément(s) sélectionné(s)**
	A, B	Tout élément correspondant à A et à B
B inclus dans **A**	A B	Tout élément correspondant à B et qui est un descendant d'un élément correspondant à A (c'est-à-dire que l'élément correspondant à B sera un fils (voire un fils d'un fils, voire un fils d'un fils d'un fils...) d'un élément correspondant à A.
	A > B	Tout élément correspondant à B et qui est un fils direct d'un élément correspondant à A
B voisin de **A**	A + B	Tout élément correspondant à B et qui est le prochain voisin d'un élément correspondant à A. Il cible **un élément frère et adjacent (qui suit immédiatement)** d'un élément HTML
	A ~ B	Tout élément correspondant à B et qui est un voisin d'un élément correspondant à A (c'est-à-dire un des fils du même parent)

Exemple

```
<ul>
    <li> Liste item1</li>
    <li> Liste item2
        <ol>
            <li> Liste item2-1 </li>
            <li> Liste item2-2 </li>
        </ol>
    </li>
    <li> Liste item3</li>
</ul>
```

```
ul li {
    color: red;
}
```

Toutes les listes auront une couleur rouge y compris les li de (ol)

```
ul > li {
    color: red;
}
```

Seuls les listes Item 1, 2 et 3 auront une couleur rouge, car ils sont les enfants de ul, alors que les items 2-1 et 2-2 sont ses petits-enfants.

```
<h1>Titre 1 </h1>
<p>Paragraphe 1</p>
<p>Paragraphe 2</p>
<p>Paragraphe 3</p>
```

```
h1+p {
    font-size: 1.5em;
    font-family: arial;
}
```

Seul le paragraphe 1 aura le style spécifié.

```
h1~p {
    font-size: 1.5em;
    font-family: arial;
}
```

Tous les paragraphes auront le même style spécifié.

```
ul,h1{
font-weight: bold;
}
```

Tous les éléments de la liste et tous les titres h1 auront le style spécifié

Les propriétés CSS

Propriétés de mise en forme du texte

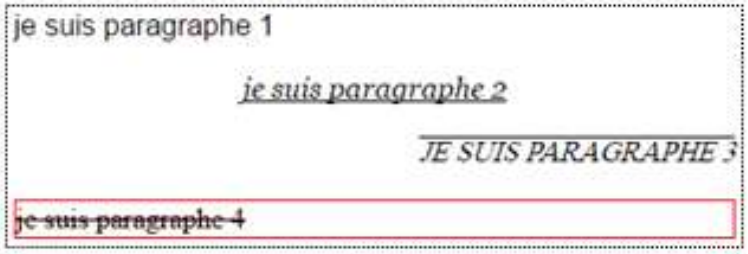
- CSS fournit de nombreuses propriétés pour la mise en forme du texte dont voici quelques-unes:

Propriété	**Description**	**Valeurs (exemples)**
font-family	définit une liste de polices dans lesquelles le texte peut apparaître.	Arial, Courier New, Georgia, Impact, Times, Verdana….
font-size	ajuste la taille du texte.	2px, 1.5em, 2rem, 20pt, 20%
font-weight	définit l'épaisseur des caractères.	normal (par défaut), bold.
font-style	détermine le style du texte.	normal, italic, oblique.
line-height	définit la hauteur de la ligne. Peut être utilisée pour définir l'interligne.	2px, 1.5em, 2rem, 20pt, 20%
text-transform	modifie la casse du texte (MAJUSCULES, minuscules ou en Capitales).	
text-align	contrôle l'alignement du texte.	left, right, center, ou justifiy
text-decoration	permet de faire apparaitre une ligne en dessous, au dessus, ou ~~à travers de texte~~.	**none**, **underline** (au dessous), **overline** (au dessus), **line-through** (barrée)
text-shadow	fait apparaître une ou plusieurs ombres derrière le texte.	5px 5px 2px blue, horizontale, verticale, fondu, couleur

Exemple

```css
p.paragraphe1 {
    font-family: Arial,Serif,Georgia,Times;
    font-size: 100%;
    font-style: normal;
    text-align: justify;
}
p.paragraphe2 {
    font-family: Georgia,Arial,Serif,times;
    font-style: italic;
    text-align: center;
    text-decoration: underline;

}
p.paragraphe3 {
    font-style: oblique;
    line-height: 20px;
    text-align: right;
    text-decoration: overline;
    text-transform: uppercase;
}
p.paragraphe4 {
    text-shadow: 0px 0px #ff0000;
    text-decoration: line-through;
    line-height:20px;
    border: 1px solid red;
}
```



- Remarque : On peut combiner toutes ces propriétés dans la propriété font.

  Il faut respecter l'ordre des valeurs prises par la propriété font

- Syntaxe : font: *font-style* **font-weight** font-size font-family;
- Exemple: p { font: italic bold Georgia, serif; }

Les unités de mesure

Pixel (X)	Centimeter [cm]
0.01 pixel (X)	0.0002645833 cm
0.1 pixel (X)	0.0026458333 cm
1 pixel (X)	0.0264583333 cm
2 pixel (X)	0.0529166667 cm
3 pixel (X)	0.079375 cm
5 pixel (X)	0.1322916667 cm
10 pixel (X)	0.2645833333 cm
20 pixel (X)	0.5291666667 cm
50 pixel (X)	1.3229166667 cm
100 pixel (X)	2.6458333333 cm

- Il existe quatre unités pour définir la taille des éléments en HTML qui sont : px, em, rem, %.
  - *px* : le pixel est une unité de mesure absolue ce qui veut dire que une valeur d'1px correspond à 1px de l'écran. L'inconvénient de px est que la taille reste fixe et ne s'adapte pas au changement de l'appareil.
  - *em* / *%*: ce sont des unités relatives à la taille du parent, em multiplie la taille tandis que % la diminue. Pour em par exemple, si dans body on définit une font-size de **16px** alors un h1 de **2em** aura une taille de **32px**. L'inconvénient est que par exemple pour un élément li si sa taille est 2em s'il y un autre li imbriqué dedans ce dernier aura alors une taille de police 4em.
  - *rem* : le rem est une unité qui agit comme le em mais qui résout le précédent problème de l'héritage. Il ne se base pas sur l'élément parent pour obtenir sa taille mais sur l'élément racine (rem = root em).

- La taille relative: permet de déterminer une taille relative à la taille prise par défaut par les navigateurs (généralement 16px). Cette taille est déterminée généralement en em (**1em =16px**) ou en pourcentage.

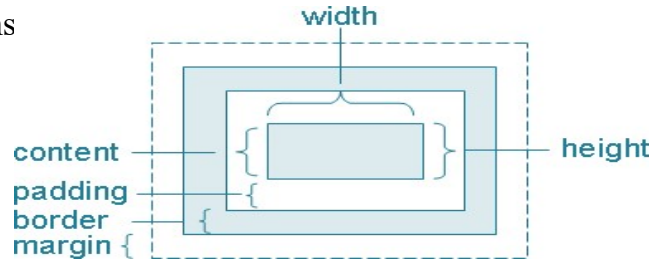
Propriétés de couleur et de fond

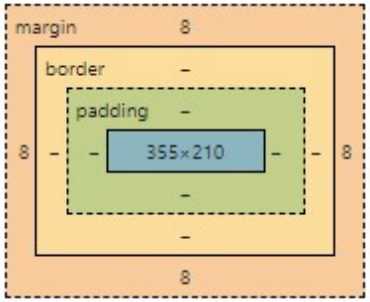
Propriété	Description	Valeurs (exemples)
color	Couleur du texte	(red, green, blue), #CF1A20, rgb(0,128,0), hsl(120, 100%, 25%)
background-color	Couleur de fond	Identique à color
background-image	Image de fond	url('image.png')
background-attachment	Fond fixe	fixed, scroll
background-repeat	Répétition du fond	Repeat,no-repeat, repeat-x, repeat-y
background-position	Position du fond	(x y), top, center, bottom, left, right
opacity	Transparence	0.5

- La valeur d'une couleur peur être désignée de deux manières différentes: par le nom de la couleur ou par la valeur hexadécimale qui correspond à cette couleur. Cette valeur hexadécimal doit commencer par le caractère # suivi de 6 chiffres en notation hexadécimale.
- La couleur en Hexadécimal: Les deux premiers chiffres correspondent à « une quantité » de la couleur Rouge, les deux suivants représentent le Vert et les deux derniers le Bleu. Il s'agit du système RVB (RGB en anglais). Exemple: color: #0000FF ou bien rgb(0,0,255)

Propriétés des boîtes

- Chaque élément d'un document est matérialisé par une boîte qui peut être ajustée grâce à des propriétés CSS spécifiques. Ces propriétés peuvent être représentées ains

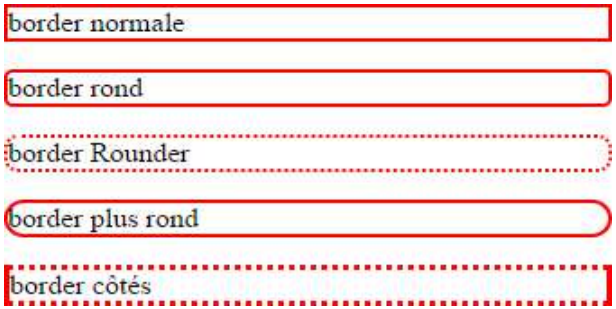




Propriété	Description	Valeurs (exemples)
width, height	Largeur, Hauteur,	150px, 80%...
margin	Super-propriété de margin. Combine : margin-top, margin-right, margin-bottom, margin-left.	23px 5px 23px 5px *(haut, droite, bas, gauche)*
padding	Super-propriété de marge intérieure. Combine : padding-top, padding-right,padding-bottom, padding-left.	23px (haut, droite, bas, gauche)
border	Super-propriété de bordure. Combine : border-width, border-color, border-style, border-top, border-right, border-bottom, border-left.	3px solid black
border-radius	Bordure arrondie.	15px 50px 30px 5px;
box-shadow	Ombre de boîte (horizontale, verticale, fondu, couleur)	6px 6px 0px black

Exemple

```
p.normal {
    border: 2px solid red;
}
p.round1 {
    border: 2px solid red;
    border-radius: 5px;
}
p.round2 {
    border: 2px dotted red;
    border-radius: 8px;
}
p.round3 {
    border: 2px solid red;
    border-radius: 12px;
}
p.côtés {
    border-top-style: dotted;
    border-right-style: solid;
    border-bottom-style: dotted;
    border-left-style: solid;
    border-color: red;
}
```




```
div {
    width: 300px;
    height: 100px;
    padding: 15px;
    background-color: yellow;
    box-shadow: 10px 10px;
    margin-left: 50%;
}
```


```
.bottStyle {
    border-radius: 5px 20px;
    background: #73AD21;
    padding: 20px;
    width: 70px;
    height: 5px;
    float: left;
    list-style-type: none;
}
```

Propriétés d'affichage

- Par défaut, les éléments se succèdent dans l'ordre où ils sont déclarés dans le code HTML tout en respectant ce qu'on appelle le flux d'un document.
- Il existe plusieurs propriétés permettant de contrôler l'affichage de n'importe quel élément sur la page html.

Propriété	Description	Valeurs (exemples)
display	Permet de déterminer le comportement d'un élément « boite ». Pour un élément son contenu est par défaut affiché sur la ligne «inline». Pour un type d'élément on peut avoir un affichage par bloque «exemple <p>»	block, inline, inline-block, none, table, table-cell,...
visibility	Permet de rendre un élément visible ou non visible (hidden). Cette dernière cache l'élément en gardant l'espace occupé à la différence de display:none permettant de disparaitre l'élément sans laisser de trace.	visible, hidden
overflow	Comportement en cas de dépassement	auto, scroll, visible, hidden
clip	Affichage d'une partie de l'élément	rect (0px, 60px, 30px, 0px)

Exemple

```
<p class="none">
1. l'Ecole Nationale des Sciences Appliquées d'Al Hoceima (ENSAH)
 est un établissement <span>public d'enseignement supérieur</span>
 relevant de l'université Mohammed Premier
</p>
<p class="inline">
2. l'Ecole Nationale des Sciences Appliquées d'Al Hoceima (ENSAH)
 est un établissement <span>public d'enseignement supérieur</span>
 relevant de l'université Mohammed Premier
</p>
<p class="block">
3. l'Ecole Nationale des Sciences Appliquées d'Al Hoceima (ENSAH)
 est un établissement <span>public d'enseignement supérieur</span>
 relevant de l'université Mohammed Premier
</p>
<p class="inline-block">
4. l'Ecole Nationale des Sciences Appliquées d'Al Hoceima (ENSAH)
 est un établissement <span>public d'enseignement supérieur</span>
 relevant de l'université Mohammed Premier
</p>
```

+

```
span {
  width: 10em;
  background: yellow;
}

.none span         { display: none;         }
.inline span       { display: inline;       }
.block span        { display: block;        }
.inline-block span { display: inline-block; }
```

=

1. l'Ecole Nationale des Sciences Appliquées d'Al Hoceima (ENSAH) est un établissement relevant de l'université Mohammed Premier

2. l'Ecole Nationale des Sciences Appliquées d'Al Hoceima (ENSAH) est un établissement public d'enseignement supérieur relevant de l'université Mohammed Premier

3. l'Ecole Nationale des Sciences Appliquées d'Al Hoceima (ENSAH) est un établissement public d'enseignement supérieur relevant de l'université Mohammed Premier

4. l'Ecole Nationale des Sciences Appliquées d'Al Hoceima (ENSAH) est un établissement public d'enseignement supérieur relevant de l'université Mohammed Premier

Propriétés de positionnement

Propriété	Description	Valeurs (exemples)
float	Indique qu'un élément doit être retiré de son flux par défaut(none) et doit être placé sur le côté droit ou sur le côté gauche de son conteneur.	left, right, none
clear	Indique qu'un élément doit venir se placer en dessous des éléments flottants qui le précèdent.	left, right, both, none
position	Permet d'activer le positionnement. Elle utilisée avec les propriétés: top, bottom, left, right. Ces derniers comportent différemment selon la valeur de position. - static: aucun effet des 4 propriétés(par défaut). - relative: par rapport à sa position normale dans le document. -fixed : par rapport au Viewport (constant au défilement) - absolute: par rapport à l'élément parent. - sticky: relative et fixed au même temps.	relative, absolute, static, fixed,sticky
top, bottom feft, right	Position par rapport au haut, au bas, à la gauche et/ou à la droite.	20px
z-index	Ordre d'affichage en cas de superposition (mise en derrière)	-1, 1

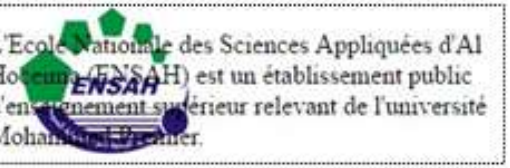
Propriétés de positionnement

```
<img src="logo.jpg">
<p>
L'Ecole Nationale des Sciences Appliquées
d'Al Hoceima <abbr> (ENSAH) </abbr> est
 un établissement public
d'enseignement supérieur relevant
de l'université Mohammed Premier.
</p>
```

+

```
img{
    position: absolute;
    top:-5px;
    left:20px;
    z-index:-1; /* Image derrière le texte*/
}
```

=




```
ul {
    list-style-type: none;
    margin: 0;
    padding: 0;
    overflow: hidden;
    background-color: #333;
}
li {
    float: left;
}
li a {
    display: inline-block;
    color: white;
    text-align: center;
    padding: 14px 16px;
    text-decoration: none;
}
li a:hover {
    background-color: #111;
}
```

+

```
<ul>
  <li><a href="#home">Home</a></li>
  <li><a href="#news">News</a></li>
  <li><a href="#contact">Contact</a></li>
  <li><a href="#about">About</a></li>
</ul>
```

=

Propriétés des listes et tableaux

Propriété	Description	Valeurs (exemples)
list-style-type	Type de liste	disc, circle, square, decimal, lower-roman, upper-roman, lower-alpha, upper-alpha, none
list-style-position	Position en retrait	outside, inside
list-style-image	Puce personnalisée	url('puce.png')
list-style	Super-propriété de liste. Combine list-style-type, list-style-position, list-style-image.	list-style: square inside;

Propriété	Description	Valeurs (exemples)
border-collapse	Permet de fusionner les bordures	collapse, separate
empty-cells	Contrôle l'affichage des cellules vides	hide, show
caption-side	Position du titre du tableau	bottom, top

Exemple

```css
ul.a {
    list-style-type: circle;
}

ul.b {
    list-style-type: square;
}

ol.c {
    list-style-type: upper-roman;
}

ol.d {
    list-style-type: lower-alpha;
}
```

- Génie Informatique :
- Génie Civil :
- Génie de l'Environnement

- Génie Informatique :
- Génie Civil :
- Génie de l'Environnement

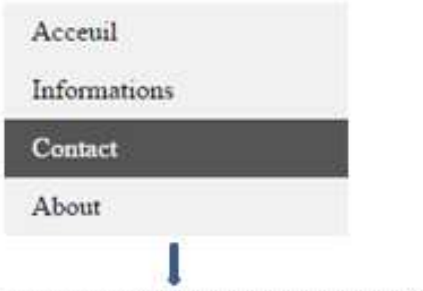
I. Génie Informatique :
II. Génie Civil :
III. Génie de l'Environnement

a. Génie Informatique :
b. Génie Civil :
c. Génie de l'Environnement

```css
table {
    border-collapse: collapse;
    width: 60%;
    margin-left: 20%;
}
th, td {
    padding: 8px;
    text-align: left;
    border-bottom: 1px solid #ddd;
}
th {
    background-color: #4CAF50;
    color: white;
}
tr:nth-child(even){background-color: #f2f2f2}
```

Prénom	Nom	Profession
Peter	Griffin	Professeur
Lois	Griffin	Avocate
Joe	Swanson	Journaliste
Cleveland	Brown	Médecin

Exemple d'application




```
ul {
    list-style-type: none;
    margin: 0;
    padding: 0;
    width: 200px;
    background-color: #f1f1f1;
}
li a {
    display: block;
    color: #000;
    padding: 8px 16px;
    text-decoration: none;
}
li a:hover {
    background-color: #555;
    color: white;
}
```


```
<h2>Dropdown Menu</h2>
<div class="dropdown">
  <button class="dropbtn">ENSAH</button>
  <div class="dropdown-content">
    <a href="#">Présentation</a>
    <a href="#">Les études</a>
    <a href="#">Conditions d'accès</a>
  </div>
</div>
```

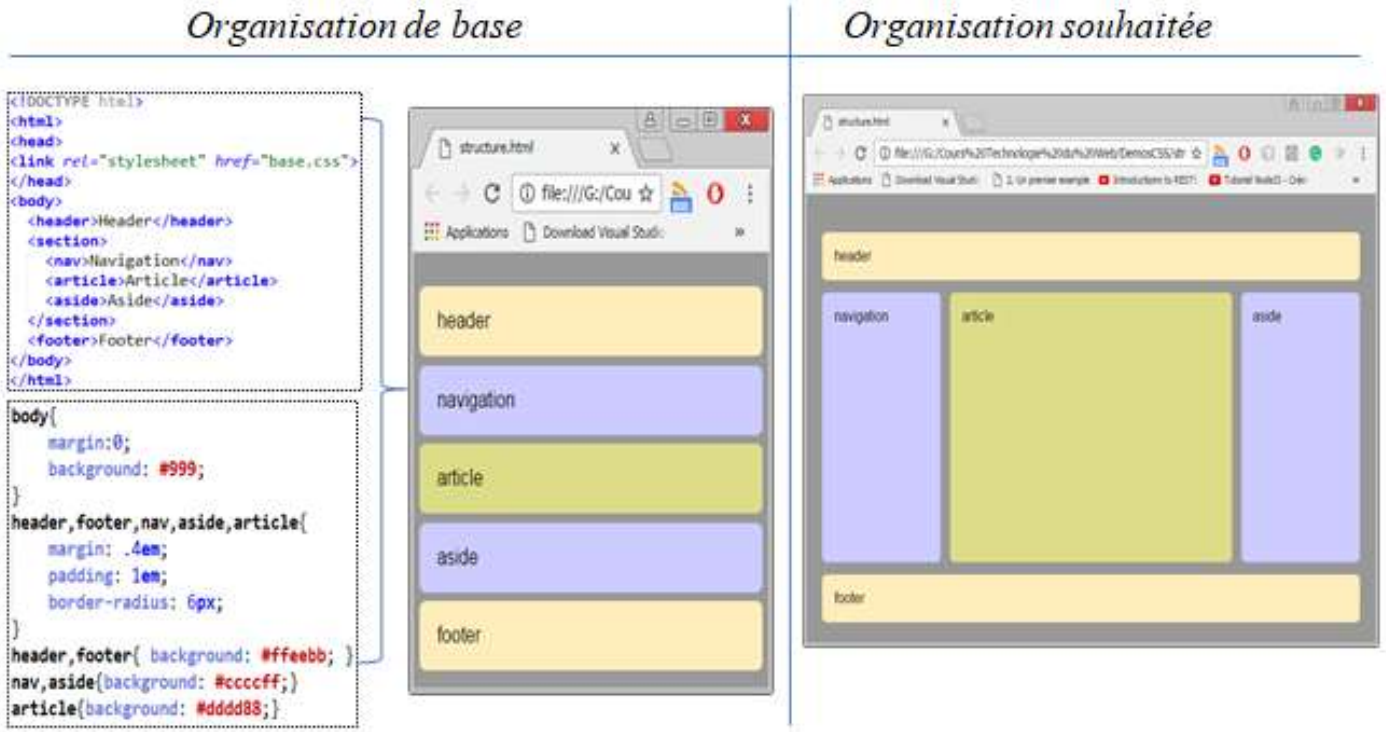
```
.dropbtn {
    background-color: #4CAF50;
    color: white;
    padding: 12px;
    font-size: 12px;
    border: none;
    cursor: pointer;
}
.dropdown {
    position: relative;
    display: inline-block;
}
.dropdown-content {
    display: none;
    position: absolute;
    background-color: #f9f9f9;
    min-width: 160px;
    box-shadow: 0px 8px 16px 0px rgba(0,0,0,0.2);
}
.dropdown-content a {
    color: black;
    padding: 12px 16px;
    text-decoration: none;
    display: block;
}
.dropdown-content a:hover {background-color: #f1f1f1}
.dropdown:hover .dropdown-content { display: block; }
.dropdown:hover .dropbtn { background-color: #3e8e41; }
```

Les grilles en CSS

- L'organisation spatiale des pages web est l'une des premières préoccupations lorsque l'on crée un site web.
- Cela consiste à définir la grille de la page web. Autrement dit, définir l'emplacement de chaque éléments de la page(en-tête, menus, section, article, footer,…)
- Pour changer le comportement naturel d'affichage, il existe plusieurs solutions à savoir :
  - *Grille avec float:* permet de faire retirer l'élément du flux normal et de le placer soit à droite (float: right) ou à gauche (float: left).
  - *Grille avec inline-block:* pouvoir aligner des blocs dimensionnés sans sortir du flux.
  - Grille avec *table-cell:* organiser des éléments sous forme de cellule d'un tableau.
  - *Grille avec CSS3 columns:* les multi-colonnes, introduites en CSS3, offrent la possibilité de distribuer du contenu sur plusieurs colonnes.
  - *Grille avec CSS3 Flexbox:* un nouveau mode de positionnement, introduit via la propriété display, permettant de créer un contexte général d'affichage sur un parent et d'en faire hériter ses enfants.

Les grilles en CSS

- Pour montrer le principe de fonctionnement on va utiliserl'exemple suivant :



Les grilles en CSS

- Dans cette section on s'intéresse plus à la technique d'organisation à base de Flexbox présentant des avantages majeurs par rapport à d'autres techniques, dont voici quelques inconvénients :

float
-Difficile à maîtriser l'espace
-Nécessite d'autres propriétés à savoir: clear, margin, width, height, etc…

```
nav,article,aside{
float: left;
}
```

inline-bock
- Difficile à maîtriser l'espace
- Nécessite d'utiliser d'autres propriétés à savoir: width, height, etc

```
nav,article,aside{
    display: inline-block;
}
```

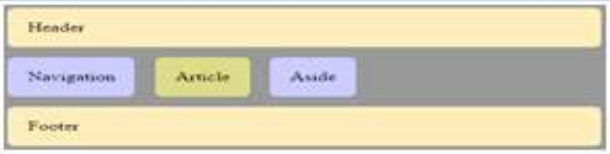
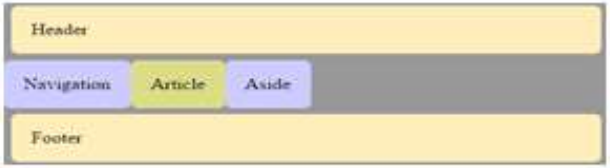


table-cell
-Difficile à maîtriser l'espace
- Pas de passage à la ligne
- Nécessite d'utiliser d'autres propriétés à savoir: width, height, etc

```
nav,article,aside{
    display: table-cell;
}
```



columns
- Difficile à maîtriser l'espace
-Les contenus passent d'une colonne à l'autre ce qui rend difficile de contrôler ce comportement

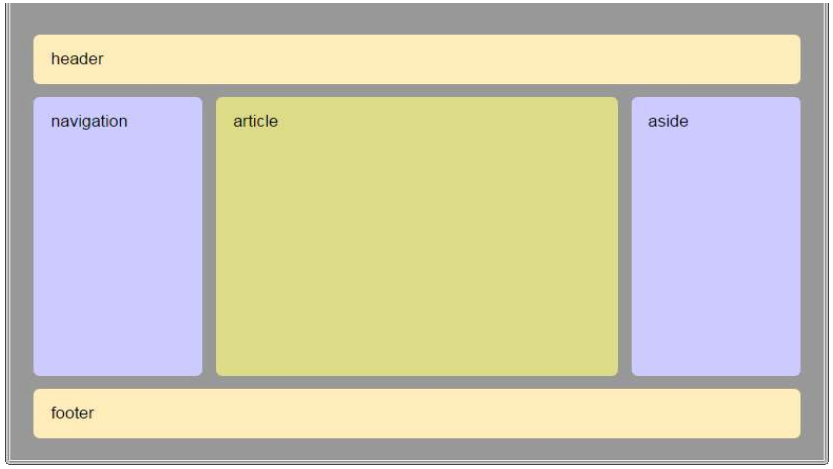
```
section{
    columns:3;
}
```

CSS Flexbox

- **Flexbox** (pour flexible box) est un nouveau mode de mise en page ajouté dans CSS3 prévoyant une disposition des éléments d'une page de telle sorte que ces éléments possèdent un comportement prévisible lorsqu'ils doivent s'accommoder de différentes tailles d'écrans/appareils.
- Il est utilisé pour placer les éléments d'une page dans n'importe quelle direction avec une flexibilité de telle sorte qu'ils peuvent avoir des dimensions pour s'adapter à la place disponible.
- L'aspect principal d'une mise en page *flex* est la capacité à modifier la largeur et/ou la hauteur des éléments pour remplir au mieux l'espace disponible sur n'importe quel appareil.
- Le principe de la mise en page avec Flexbox est simple : on utilise un conteneur et on place les éléments dedans.
- Le conteneur ***flex*** élargit les éléments pour remplir l'espace libre ou les rétrécir pour éviter les débordements.

Flexbox : mise en page

- Pour montrer comment utiliser flexbox pour organiser le contenu on va utiliser l'exemple précédent. L'objectif est d'avoir la mise en page suivante :



- Pour ce faire, il faut activer la propriété flex dans le conteneur des éléments de document.
- Dans notre cas, on va considérer deux conteneurs: le premier c'est <body> qui est le conteneur principale. Le deuxième c'est <section> qui sert de conteneur des éléments <nav> <aricle> et <aside>, le but étant d'avoir une flexibilité au niveau de ces éléments.

Flexbox : mise en page

- Pour ce faire, il faut d'abord, activer flex (*display:flex*) au niveau de conteneur body pour donner un contexte flex à tout ses éléments directs. Pour placer ces derniers de façon verticale, il faut utiliser la propriété *flex-direction* en lui associant la valeur

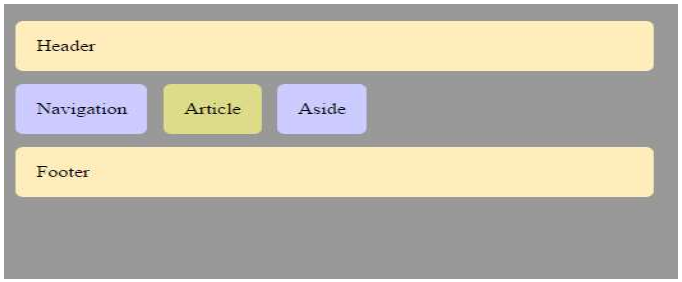
```
body {
  display: flex; /* crée un contexte flex pour ses enfants */
  flex-direction: column; /* affichage vertical */
  min-height: 100vh; /* toute la hauteur du viewport */
  padding: 1em;
}
```

- La propriété *min-height* :100vh, permet de spécifier que la flexibilité doit être en niveau de tout le Viewport (la surface de la fenêtre du navigateur).
- Les éléments de ***section*** vont être placés horizontalement. C'est pourquoi il faut activer *flex* aussi au niveau de ce conteneur :

```
section {
  display: flex; /* crée un contexte flex pour ses enfants */
}
```

Flexbox : mise en page

- Ceci permet d'avoir la mise en page suivante :



- S'on veut par exemple que la hauteur des éléments *header* et *footer* (height: 5em;) et la largeur des éléments *nav* et *aside* soient « fixe » (exemple width:10em).
- Pour bien maîtriser l'espace restant, c'est-à-dire pour que chaque élément puisse prendre l'espace qui reste, il faut lui rajouter la propriété *flex* avec valeur égale à 1. En faite, la valeur de *flex* permet de partager l'espace restant avec des pourcentages différents.

```
section{
    display: flex;
    flex:1; /*occupe la hauteur restante*/
}
article{
    flex:1; /* occupe la largeur restante*/
}
header,footer{
    height: 5em;
}
nav,aside{
    width: 10em;
}
```

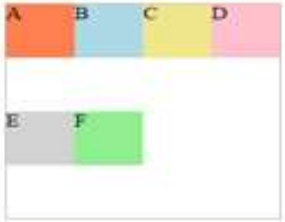
Flexbox : propriétés

Propriété	Description	Valeur
flex-direction	Permet d'agencer les éléments dans le sens voulu	*row, column, row-reverse, column-reverse*
flex-wrap	Permet de faire retour à la ligne	*Nowrap, wrap, wrap-reverse*
justify-content	Permet d'aligner les éléments sur l'axe X	*flex-start, flex-end, center, space-between, space-around, …*
align-items	Permet d'aligner les éléments sur l'axe Y	
order	Permet de modifier l'ordre des éléments	*1, 2,3…..*

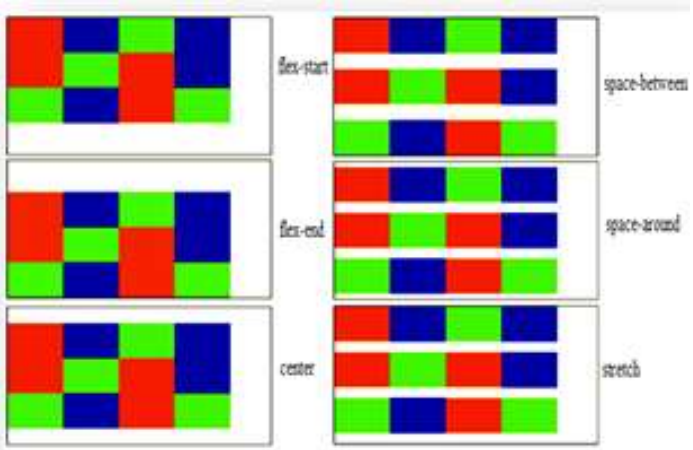
- Exemple :

```
display: flex;
flex-direction: row-reverse;
```


```
#main {
    width: 200px;
    height: 200px;
    display: flex;
    flex-wrap: wrap;
}
div {
    width: 50px;
    height: 50px;
}
```




```
div#DivA {order: 2;}
div#DivB {order: 4;}
div#DivC {order: 3;}
div#DivD {order: 1;}
```



Responsive Web Design

- Divers appareils (Ordinateurs, Smartphones, tablette, SmartTV, ..) peuvent être utilisés pour accéder à un site ou application Web.
- Le problème est que l'affichage ne s'adapte pas implicitement à la taille d'écran qui est différente d'un appareil à l'autre.
- Pour remédier à ce problème, deux solutions sont possible :
  - *Application native* : des applications dédiés pour chaque type d'appareil.
  - *Application responsive* : une même et seule application dont sa conception s'adapte pour chaque type d'appareil.
- La différence entre les deux solutions est que la première est évidement plus coûteuse au niveau conception, développement et maintenance.

Responsive Web Design

- Le but de « responsive Web design» est de *créer des solutions web aux interfaces* flexibles et élastiques s'adaptant automatiquement à la taille et à la résolution de l'écran de l'internaute.

- Pour déterminer comment les éléments du site doivent s'afficher, on se base généralement sur la largeur de l'écran.
- Cette technique peut être réalisé grâce aux *media queries*, un ensemble de règles ajoutés à CSS3 pour permettre le contrôle du style d'affichage en fonction des caractéristiques de l'écran.

Les medias queries

- Les *media queries* permettent d'adapter la présentation du contenu à une large gamme d'appareils sans changer le contenu lui-même.
- C'est l'un des nouveautés de CSS3, il ne s'agit pas de nouvelles propriétés mais de *règles* que l'on peut appliquer pour limiter la portée des déclarations CSS.
- Un exemple de règle: *Si la résolution de l'écran du visiteur est inférieure à la taille spécifié, alors appliquer les propriétés CSS correspondantes*.
- Les *media queries* sont donc des règles qui indiquent quand on doit appliquer des propriétés CSS.
- Cela va permettre de changer l'apparence du site dans certaines conditions: on peut par exemple augmenter la taille du texte, changer la couleur de fond, positionner différemment le menu dans certaines résolutions, etc.

Les medias queries

- Il y a deux façons de les utiliser :
  - en chargeant une feuille de style .css différente en fonction de la règle (ex : Si la résolution est inférieure à 800px de large, charge le fichier «small.css»). Dans ce cas, on ajoute à l'élément <link> un attribut media, dans lequel on va écrire la règle qui doit s'appliquer pour que le fichier soit chargé. Un exemple : <link rel="stylesheet" media=« (max-width: 800px)" href="small.css" />
  - en écrivant la règle directement dans le fichier .css habituel (ex : « Si la résolution est inférieure à 800px de large, appliquer les propriétés CSS ci-dessous »). Pour ce faire voici la syntaxe :

```
@media ( /*les règles sur la taille de l'ecran*/ ) {
   /* les propriétés CSS ici */
   }
```

- Exemple :

```
@media(max-width: 800px){
    p{
        color:red;
    }
}
```

Les medias queries : règles

- Il existe de nombreuses règles permettant de construire des media queries :
  - **Règles précisant le type d'écran** : Il possible de cibler un type de l'écran pour y appliquer une règle. Le type d'écran est précisé en utilisant les mots- clés suivants précédé de @media : screen : écran « classique » ; handheld : périphérique mobile ; print : impression ; tv : télévision ; projection : projecteur ; all : tous les types d'écran.
  - **Autres règles :**
    - color : gestion de la couleur (en bits/pixel).
    - height : hauteur de la zone d'affichage (fenêtre).
    - width : largeur de la zone d'affichage (fenêtre).
    - device-height : hauteur du périphérique.
    - device-width : largeur du périphérique.
    - orientation : orientation du périphérique (portrait ou paysage).
- Les règles peuvent être combinées à l'aide des mots suivants: only : « uniquement » ; and : « et » ; not : « non ».
- Exemple : @media tv and (min-width: 1280px)

Media queries : Exemples

- **Exemple 1**: Dans cet exemple, le texte des paragraphes sera écrit en bleu tant que la largeur de l'écran de navigateur dépasse les 1024px. Dans le cas contraire, les paragraphes seront écrits en style plus gros et en rouge.

```
/* Paragraphes en bleu par défaut */
p
{
    color: blue;
}
/* Nouvelles règles si la fenêtre fait au plus 1024px de large */
@media screen and (max-width: 1024px)
{
    p
    {
        color: red;
        background-color: black;
        font-size: 1.2em;
    }
}
```

Media queries : Exemples

- **Exemple 2**: on va se servir de l'exemple utilisé dans la section consacrée à flexbox. L'objectif c'est d'adapter la mise en page créée pour les petits appareils comme Smartphone ou tablette.

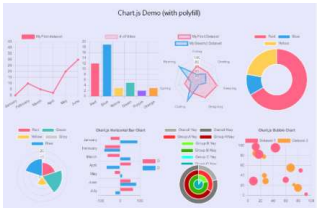
- Le code CSS correspondant à cette adaptation est donné comme suite:

```css
@media (max-width: 640px) {
  section {
    flex-direction: column; /* affichage vertical */
  }
  nav,aside {
    width: auto; /* pour écraser la valeur 10em */
  }
  nav,aside,article {
    flex-basis: auto; /* pour écraser la valeur 0, due au flex: 1 */
  }
}
```

Bootstrap

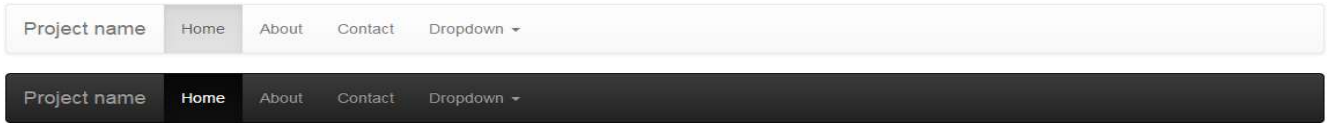
Présentation

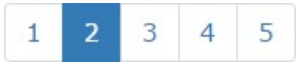
- Bootstrap est une bibliothèque gratuit et open-source contenant un ensemble d'éléments de feuilles de style prédéfinis utilisée coté front- end pour faciliter la conception de sites Web et d'applications Web.
- Il s'agit d'un ensemble des définitions de style de base pour tous les éléments HTML. Ils permettent de fournir une apparence uniforme et moderne pour le formatage du texte, des tables et des éléments de formulaire.
- En plus des éléments HTML réguliers, Bootstrap contient également de nombres éléments graphiques couramment utilisés au format standardisé : *boutons, libellés, icônes, miniatures, barres de progression*, ainsi que des extensions JavaScript optionnelles.
- Bootstrap adopte la conception des applications web adaptatives(responsive). En effet, les pages web développés à base de bootsrap s'adaptent dynamiquement au format des supports sur lesquels ils sont consultés (PC, tablette, smartphone).



Exemples de style Bootstrap

#	First Name	Last Name	Username
1	Mark	Otto	@mdo
2	Jacob	Thornton	@fat
3	Larry	the Bird	@twitter





Mise en œuvre

- L'ensemble définitions de style sont implémentés en tant que classes CSS (Exemple : btn, dropdown, input-group, nav…), qui doivent être appliquées à certains éléments HTML d'une page.
- Par exemple, pour changer le style par défaut d'un bouton par celui de Bootstrap, il faut donner à l'attribut classe de cet élément la valeur btn correspondante à un style prédéfini dans Bootstrap spécial pour les boutons.

```
<button type="button"> Bouton par defaut </button>
<button type="button" class="btn"> Bouton style bootsrap </button>
```

- Les différentes sélecteurs de type classes sont définies dans un fichier *.css* qui est « *bootstrap.css* » ou « *bootstrap.min.css* » *(*une version minimalisée en réduisant la taille de fichier).
- Pour pouvoir utiliser le style Bootstrap, il faut lier le document html avec au moins le fichier « bootstrap.min.css ». Deux manières:
  - En téléchargeant le fichier en question : `<link rel="stylesheet" type="text/css" href="css/bootstrap.min.css">`
  - En faisant liaison avec BootstrapCDN.
- Il existe des centaines de style prédéfinie dans Bootstrap: http://getbootstrap.com/components/

Style Bootstrap pour les boutons

- Bootstrap propose différentes styles (au nombre de sept) de boutons:

- Pour obtenir les styles de bouton ci-dessus, voici les classes correspondantes: .btn-primary, .btn-secondary, .btn-success, .btn- danger, .btn-warning, .btn-info, btn-light, .btn-dark, btn-link.
- Un exemple d'utilisation :
  - <button type="button" class="btn btn-primary">Ajouter</button>
  - <button type="button" class="btn btn-success">Modifier</button>
  - <button type="button" class="btn btn-danger">Supprimer</button>

Style Bootstrap pour les tableaux

- Pour la mise en place en style des tableaux, BS propose plusieurs style (.table, .table-bordered, .table-hover, .table-condensed). La classe .table représente le style de base.
- Exemple : la classe .table-striped ajoute des rayures zébrées à une table:

```
<table class="table table-striped">
  <thead>
    <tr>
      <th>Nom</th> <th>Prénom</th> <th>Profession</th>
    </tr>
  </thead>
  <tbody>
    <tr> <td>John</td> <td>Doe</td> <td>Professeur</td> </tr>
    <tr> <td>Mary</td> <td>Moe</td> <td>Avocate</td> </tr>
    <tr> <td>July</td> <td>Dooley</td> <td>Journaliste</td> </tr>
  </tbody>
</table>
```

Prénom	Nom	Profession
John	Doe	Professeur
Mary	Moe	Avocate
July	Dooley	Journaliste

Style Bootstrap pour les images

- La classe .rounded ajoute des coins arrondis à une image
- La classe .rounded-circle permet de mettre l'image en cercle
- La classe .img-thumbnail forme l'image sous forme de vignette:

Rounded Corners:

Circle:

Thumbnail:

Style Bootstrap pour les formulaires

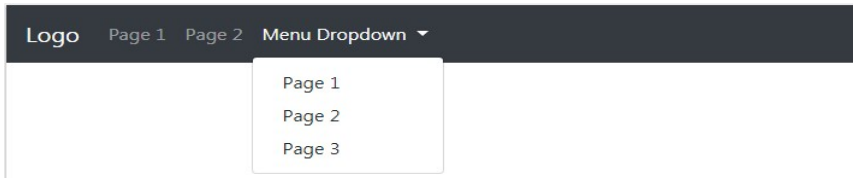
- Bootstrap propose trois types de mises en forme pour les formulaires :
  - Forme verticale (par défaut)
  - Forme horizontale(.form-horizontal)
  - Formulaire en ligne (.form-inline): les éléments seront affichés sur une même ligne.

```
<form class="form-inline" action="">
  <label for="email">Email:</label>
  <input type="email" class="form-control" id="email" placeholder="Enter email" name="email">
  <label for="pwd">Password:</label>
  <input type="password" class="form-control" id="pwd" placeholder="Enter password" name="pswd">
  <div class="form-check">
    <label class="form-check-label">
      <input class="form-check-input" type="checkbox" name="remember"> Remember me
    </label>
  </div>
  <button type="submit" class="btn btn-primary">Submit</button>
</form>
```

Style Bootstrap pour le menu de navigation

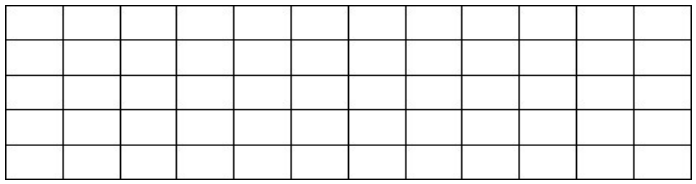
- Grâce à BS on peut créer des menues de navigation avec des styles différents.
- Le style de base est .nav. Pour un menu positionné en entête de page, il y a le style .navbar.
- Exemple :

```
<nav class="navbar navbar-expand-sm bg-dark navbar-dark">
  <!-- Brand -->
  <a class="navbar-brand" href="#">Logo</a>
  <!-- Links -->
  <ul class="navbar-nav">
    <li class="nav-item"> <a class="nav-link" href="#">Page 1 </a> </li>
    <li class="nav-item"> <a class="nav-link" href="#">Page 2 </a> </li>
    <!-- Dropdown -->
    <li class="nav-item dropdown">
      <a class="nav-link dropdown-toggle" href="#" id="navbardrop" data-toggle="dropdown">
        Menu Dropdown
      </a>
      <div class="dropdown-menu">
        <a class="dropdown-item" href="#">Page 1 </a>
        <a class="dropdown-item" href="#">Page 2</a>
        <a class="dropdown-item" href="#">Page 3</a>
      </div>
    </li>
  </ul>
</nav>
```



La mise en page Bootstrap

- L'organisation des éléments en Bootstrap se base sur une grille qui comporte 12 cellules comme suit .



- On peut alors décider d'organiser du contenu en utilisant pour chaque élément une ou plusieurs cellules, comme à la figure suivante :

- Sur une ligne donnée qu'on peut définir grâce à .row, BS propose cinq (Version 4) batteries de 12 classes pour définir le nombre de colonnes utilisées pour chaque élément:

La mise en page Bootstrap

`col-xs-1` ou `col-sm-1` ou `col-md-1` ou `col-lg-1`

`col-xs-2` ou `col-sm-2` ou `col-md-2` ou `col-lg-2`

`...`

`col-xs-12` ou `col-sm-12` ou `col-md-12` ou `col-lg-12`

- Les 5 sortes de classes pour les colonnes correspondent aux 5 types d'appareils sur lesquels peut s'adapter une application créée à base de Bootstrap :

	Petit écran (Phone en portrait)	Écran réduit (Phone en paysage)	Écran moyen (tablette)	Grand écran (Laptop, Desktop)	Large écran (Desktop, Tv)
Classe	col-	col-sm-*	col-md-*	col-lg-*	col-xl
Valeur de référence	<576px	>=576px	>=768px	>=992px	>=1200px

* Le nom des classes est intuitif : sm pour small, md pour medium et lg pour large.

La mise en page Bootstrap

- **Exemple** : le code suivant montre un exemple d'une affichage qui s'adapte au type d'appareil.

```
<header class="row"> Header </header>
<section class="row">
    <nav class="col-sm-2"> Navigation </nav>
    <article class="col-sm-8"> Article </article>
    <aside class="col-sm-2"> Aside </aside>
</section>
<footer class="row"> Footer </footer>
```

- S'il s'agit d'un Smartphone, les éléments vont utilisés l'affichage par défaut (Affichage de petit écran). Sinon on aura une affichage en 3 lignes, la ligne de milieu est divisée en 3 colonnes.

Divers

- Pingendo est un outil gratuit pour maquetter rapidement des sites Web basés sur le framework Bootstrap. Pour le télécharger il faut aller sur le site www.pingendo.com/
- Plusieurs styles Bootstrap sont disponible sur les sites : https://bootsnipp.com, https://bootswatch.com
- Pour ajouter des icones à des éléments, la bibliothèque « awesome » contient plus de 2000 icones.

Merci pour votre attention !!!