

Chapitre 3 : Le style des pages Web (CSS & Bootstrap)

Plan

- I. Introduction
- II. Sélecteurs
- III. Propriétés CSS
- IV. Les grilles en CSS
- V. Le responsive Web design
- VI. Bootstrap

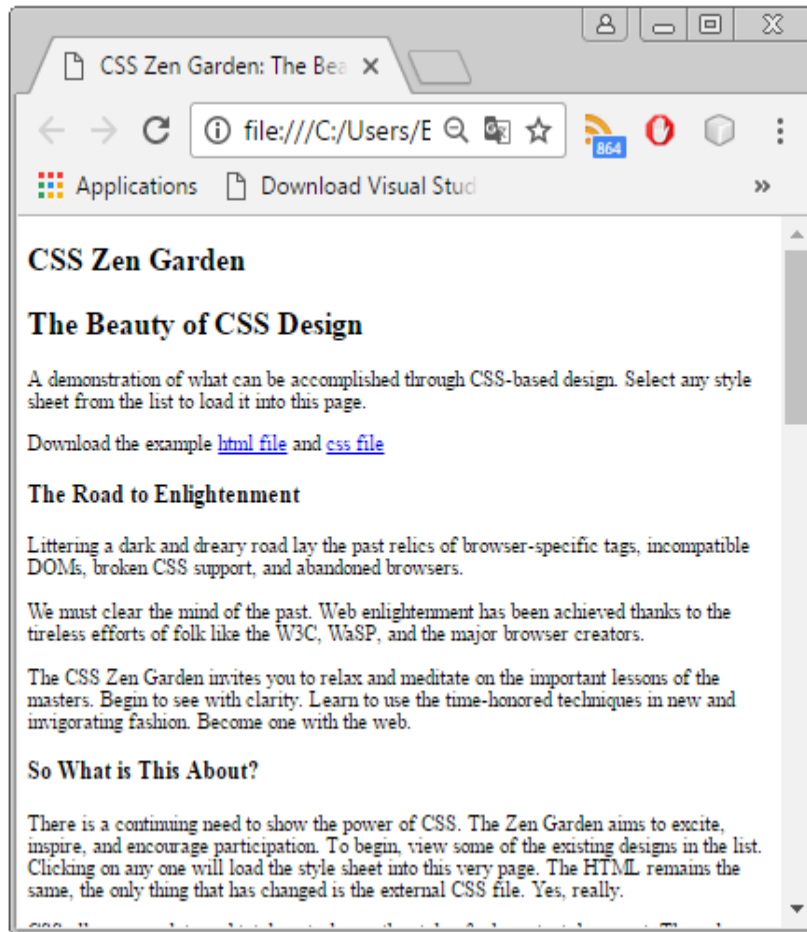
I. Introduction

1. CSS

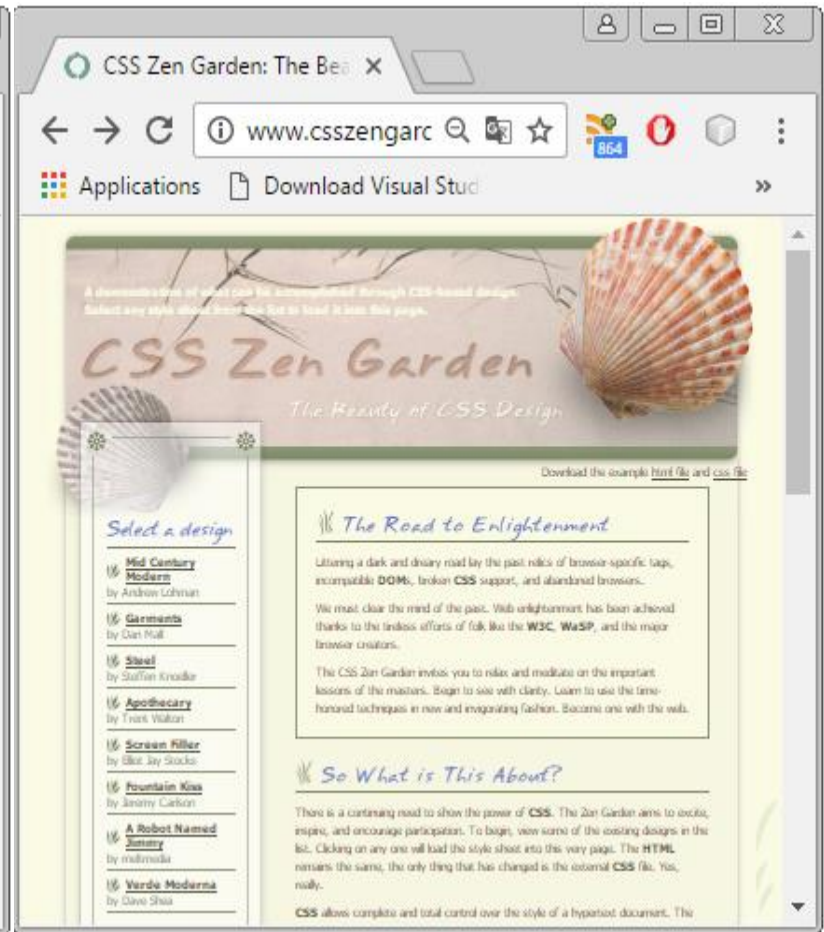
- ▶ **Cascading Style Sheets (CSS)** est un langage de feuille de style utilisé pour définir l'apparence visuelle et la mise en page des pages web.
- ▶ HTML définit uniquement la structure du contenu, c'est le langage CSS, qui **détermine l'apparence de ce contenu**.
- ▶ CSS **décrit la façon dont les 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 spécifié (ou défini) par le W3C **utilise des sélecteurs pour appliquer des règles de styles aux éléments HTML**.
- ▶ CSS3 est une évolution de CSS, avec des fonctionnalités avancées ajoutées à la spécification de base de CSS pour permettre aux développeurs de créer des designs plus créatifs et sophistiqués pour les pages web.

.... la suite

► Exemple (source : <http://www.csszengarden.com/>)



HTML sans CSS



HTML + CSS

2. Principe de fonctionnement

- Pour personnaliser l'apparence d'une page web, CSS utilise **des sélecteurs pour identifier les éléments HTML** auxquels des règles de style doivent être appliquées.
- Chaque **règle de style** est composée d'une **liste de propriétés**, qui forment une déclaration CSS. Ces déclarations sont ensuite regroupées dans des blocs de déclarations.
- La syntaxe générale pour définir une règle CSS est la suivante :

```
le sélecteur {
    propriété1: valeurX;
    propriété2: valeurY
}
```

- ▶ *Propriété* : permettant de définir une fonctionnalité donnée, comme la couleur, style d'écriture, la taille, etc ...
 - ▶ *Valeur* : décrit comment la fonctionnalité doit être utilisée par le navigateur
- ▶ Exemple :
- ```
>p { color: red; }
```

Sélecteur      {    color:    red;    }

propriété      valeur

### 3. Notion de cascade

- ▶ 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 sélecteurs**.
- ▶ Un ordre de cascade : une même règle appliquée aux éléments enfants annule celle de l'élément père.

```
<body> <-- élément grand-père
 <p> </p>
 <-- élément père

 <-- deux éléments enfants

</body>
```

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

*Dans ce cas c'est la couleur rouge qui sera appliquée.*

## 4. 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>
```

- ▶ **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.

```
<!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

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

style.css





# II. Les sélecteurs

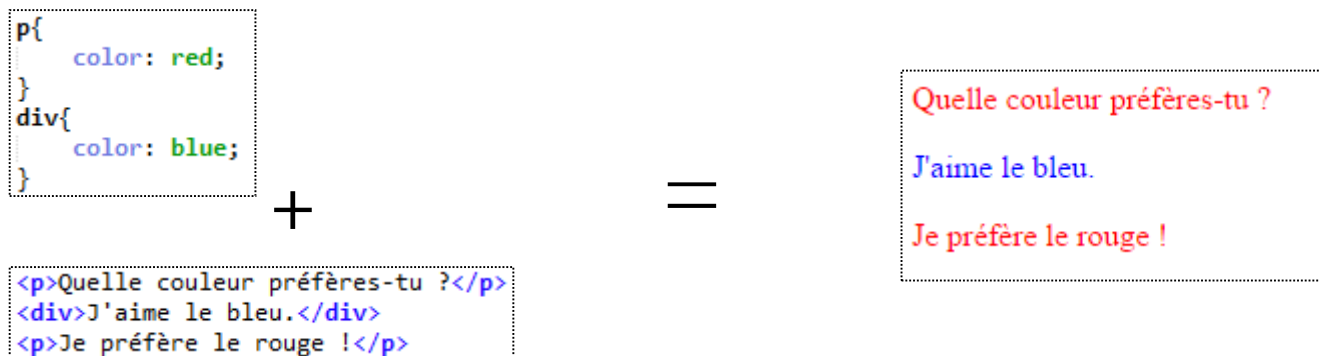
## 1. Présentation

---

- ▶ Le rôle d'un sélecteur en CSS est de **localiser un ou plusieurs éléments HTML** pour leur appliquer des règles de style. Il existe une variété de méthodes pour cibler et localiser les éléments HTML sur une page web, et on peut les regrouper en deux types :
  - ▶ 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*,
    - ▶ de façon globale.
  - ▶ Les sélecteurs combinés :
    - ▶ en fonction de la hiérarchie d'un élément,
    - ▶ de la position d'un élément.

## 2. 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 est comment faire pour que certains paragraphes seulement soient écrits d'une manière différente.

### 3. Les sélecteurs de classe et d'identifiant

- 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 :

```
<p class="pragraphe1">je suis le premier paragraphe</p>
<p class="pragraphe2">je suis le deuxieme paragraphe</p>
```



```
.pragraphe1{
 color: red;
}
.pragraphe2{
 color: blue;
}
```

```
<p id="pragraphe1">je suis le premier paragraphe</p>
<p id="pragraphe2">je suis le deuxieme paragraphe</p>
```



```
#pragraphe1{
 color: red;
}
#pragraphe2{
 color: blue;
}
```

## 4. Les sélecteurs d'attribut et leurs valeurs

- ▶ 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'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 finit avec 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 l'attribut attr vaut val ou commence par val suivi de -. Exemple : `<a href=" " hreflang="en-US">`

.... la suite

## ► Exemples :

```
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 deuxième semestre :

```

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

```

+

```
[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

## 5. Sélecteur pseudo-classes

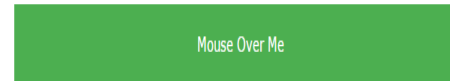
---

- ▶ 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;  
}`
- ▶ Voici une liste non-exhaustive :
  - **:*hover*** : cibler un élément lorsqu'il est survolé par la souris.
  - **:*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 `<div>` 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>` = paragraphe 1  
`<p>paragraphe 2 </p>` = paragraphe 2

- Exemple 3 :

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

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

## 6. 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 cibler des parties de l'élément concerné.
- ▶ 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>`

= C Cette introduction ...

```
a {
 color: red;
 font-weight: bold;
 text-decoration: none;
}
[href*=ensah]::after {
 content: '-->';
}
```

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

= cliquer -->

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

Le style s'applique sur le texte qu'on est en train de le sélectionner.



## 7. Les combineurs

- ▶ 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 « combineurs ».
- ▶ **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 <b>qui est un descendant</b> 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 <b>qui est un fils direct</b> d'un élément correspondant à A
B voisin de A	A + B	Tout élément correspondant à B et qui est <b>le prochain voisin</b> d'un élément correspondant à A (c'est-à-dire le prochain fils du même parent)
	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)

## ► Exemples :

<pre>&lt;ul&gt;   &lt;li&gt; Liste item1&lt;/li&gt;   &lt;li&gt; Liste item2&lt;/li&gt;   &lt;ol&gt;     &lt;li&gt; Liste item2-1 &lt;/li&gt;     &lt;li&gt; Liste item2-2 &lt;/li&gt;   &lt;/ol&gt;   &lt;li&gt; Liste item3&lt;/li&gt; &lt;/ul&gt;</pre>	<pre>ul li {   color: red; }</pre> <p>Toutes les items listes auront une couleur rouge y compris les li de (ol)</p> <hr/> <pre>ul &gt; li {   color: red; }</pre> <p>Seuls les listes Item 1, 2 et 3 auront une couleur rouge, car ils sont les enfants direct de ul, alors que les items 2-1 et 2-2 sont ses petits-enfants.</p>
<pre>&lt;h1&gt;Titre 1 &lt;/h1&gt; &lt;p&gt;Paragraphe 1&lt;/p&gt; &lt;p&gt;Paragraphe 2&lt;/p&gt; &lt;p&gt;Paragraphe 3&lt;/p&gt;</pre>	<pre>h1+p {   font-size: 1.5em;   font-family: arial; } h1~p {   font-size: 1.5em;   font-family: arial; }</pre> <p>Seul le paragraphe 1 aura le style spécifié.</p> <p>Tous les paragraphes auront le même style spécifié.</p>
<pre>ul,h1{   font-weight: bold; }</pre>	<p>Tous les éléments de la liste et tous les titres h1 auront le style spécifié</p>

## 8. Le 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.
- ▶ Exemple :

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

+

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

=

**Pour tester CSS**

Je suis le premier paragraphe

**Je suis le deuxième paragraphe**

Je suis le deuxième paragraphe

# III. Les propriétés CSS

## 1. Présentation

---

- ▶ Les propriétés CSS sont des règles utilisées pour définir l'apparence, le positionnement et le formatage des éléments HTML sur une page web.
- ▶ Les propriétés CSS peuvent être appliquées à différents éléments HTML pour contrôler des aspects tels que la couleur, la taille, la police, la disposition, les marges, les bordures, les arrière-plans, les effets visuels et bien plus encore.
- ▶ Dans la suite de cette section, nous allons présenter certaines des propriétés CSS les plus couramment utilisées, classées par catégorie :
  - ▶ Propriétés de mise en forme du texte
  - ▶ Propriétés de couleur et de fond
  - ▶ Propriétés des boîtes
  - ▶ Propriétés d'affichage
  - ▶ Propriétés de positionnement
  - ▶ Propriétés des listes et tableaux

## 2. 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-style	détermine le <b>style du texte</b> .	normal, italic, oblique.
font-weight	définit <b>l'épaisseur des caractères</b> .	normal, bold.
font-size	ajuste <b>la taille du texte</b> .	2px, 1.5em, 2rem, 20pt, 20%
line-height	définit <b>la hauteur</b> de la ligne. Peut être utilisée pour <b>définir l'interligne</b> .	2px, 1.5em, 2rem, 20pt, 20%
font-family	définit une <b>liste de polices</b> dans lesquelles le texte peut apparaître.	Arial, Courier New, Georgia, Impact, Times, Verdana....
font	Super propriétés : font: [style] [variant] [weight] [size] [/line-height] [family];	italic small-caps bold 16px/1.5 "Arial", sans-serif;
text-transform	modifie <b>la casse du texte</b> (MAJUSCULES, minuscules ou en Capitales).	capitalize, uppercase, lowercase, ...
text-align	contrôle l'alignement du texte.	left, right, center, ou justify
text-decoration	permet de faire apparaître <b>une ligne en dessous</b> , au dessus, ou <b>à travers de texte</b> .	overline, line-through, underline
text-shadow	fait apparaître une ou plusieurs <b>ombres</b> derrière le texte.	5px 5px 2px blue, horizontale, verticale, fondu, couleur

.... la suite

## ► Exemple :

```
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;
}
```

je suis paragraphe 1

*je suis paragraphe 2*

*JE SUIS PARAGRAPHE 3*

je suis paragraphe 4

### 3. Les unités de mesures

---

- ▶ Il existe plusieurs unités de mesures pour définir la taille des éléments en HTML, à savoir : 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 et 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).

## 4. Propriétés de couleur et de fond

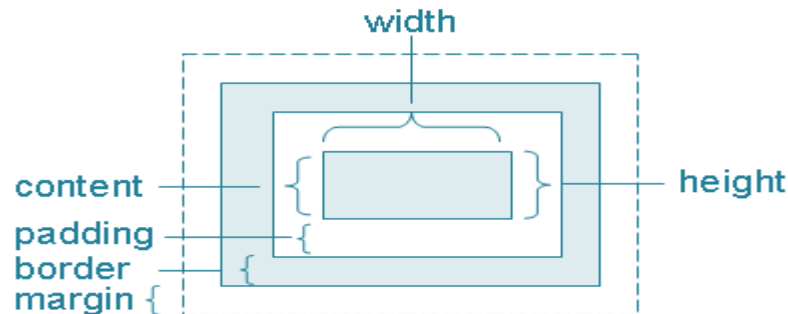
---

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



## 5. 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 ainsi :



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 5px 23px 5px (haut, droite, bas, gauche)
border	Super-propriété de bordure. Combine : border-width, border-style, border-color, 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

## ► Exemples :

```
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;
}
```

border normale

border rond

border Rounder

border plus rond

border côtés

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

Je suis un box-shadow

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

[Accueil](#)

[Présentation](#)

[Contact](#)

## 6. 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	La propriété CSS "display" définit comment un élément doit être affiché sur une page Web et comment il doit interagir avec les autres éléments. Si par exemple : " <b>display:block</b> ": cela signifie que <b>l'élément occupe tout l'espace horizontal</b> disponible <b>et crée une nouvelle ligne après lui</b> .	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 <b>gardant l'espace occupé</b> à la différence de display:none.	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 public d'enseignement supérieur
relevant de l'université Mohammed Premier
</p>
<p class="inline">
2. L'Ecole Nationale des Sciences Appliquées d'Al Hoceima (ENSAH)
est un établissement public d'enseignement supérieur
relevant de l'université Mohammed Premier
</p>
<p class="block">
3. L'Ecole Nationale des Sciences Appliquées d'Al Hoceima (ENSAH)
est un établissement public d'enseignement supérieur
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 public d'enseignement supérieur
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

## 7. Propriétés de positionnement classiques

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. <b>Elle utilisée avec les propriétés: <i>top, bottom, left, right</i></b> . Ces derniers comportent différemment selon la valeur de position. - <i>static</i> : aucun effet des 4 propriétés(par défaut). - <i>relative</i> : par rapport à <b>sa position normale</b> dans le document. - <i>absolute</i> : par rapport à l'élément parent qui a une propriété de position différente de static sinon à body. - <i>fixed</i> : par rapport au Viewport ( <b>constant au défilement</b> ) - <i>sticky</i> : relative et fixed au même temps mais reste toujours visible sur le Viewport.	relative, absolute, static, fixed, sticky
top, bottom left, 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

## ► Exemples :

```

<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*/
}
```

=

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

```
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;
}
```

+

```

 Home
 News
 Contact
 About

```

=

Home    News    Contact    About

## 8. 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

## ► Exemples :

```
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;
}
```

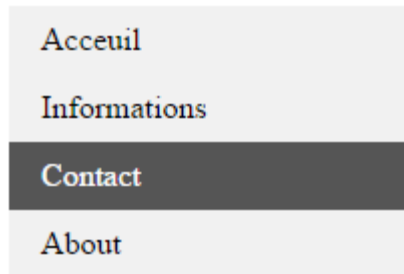
- Génie Informatique ;
  - Génie Civil ;
  - Génie de l'Environnement
- 
- Génie Informatique ;
  - Génie Civil ;
  - Génie de l'Environnement
- 
- I. Génie Informatique ;
  - II. Génie Civil ;
  - III. Génie de l'Environnement
- 
- a. Génie Informatique ;
  - b. Génie Civil ;
  - c. Génie de l'Environnement

```
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}
```

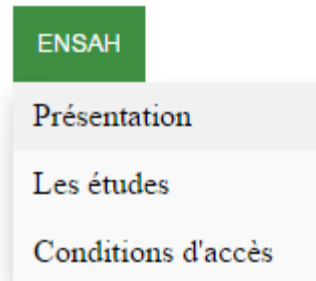
Prénom	Nom	Profession
Peter	Griffin	Professeur
Lois	Griffin	Avocate
Joe	Swanson	Journaliste
Cleveland	Brown	Médecin



## 9. Exemples d'applications



```
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">
 Présentation
 Les études
 Conditions d'accès
 </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; }
```

# IV. Les grilles en CSS

## 1. Présentation

- ▶ 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,...)
- ▶ 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.
- ▶ 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.

.... la suite

- Dans la suite, on s'intéressera plus à la technique d'organisation à base de Flexbox. On va utiliser l'exemple ci-après afin d'illustrer le mode d'utilisation de cette technologie.

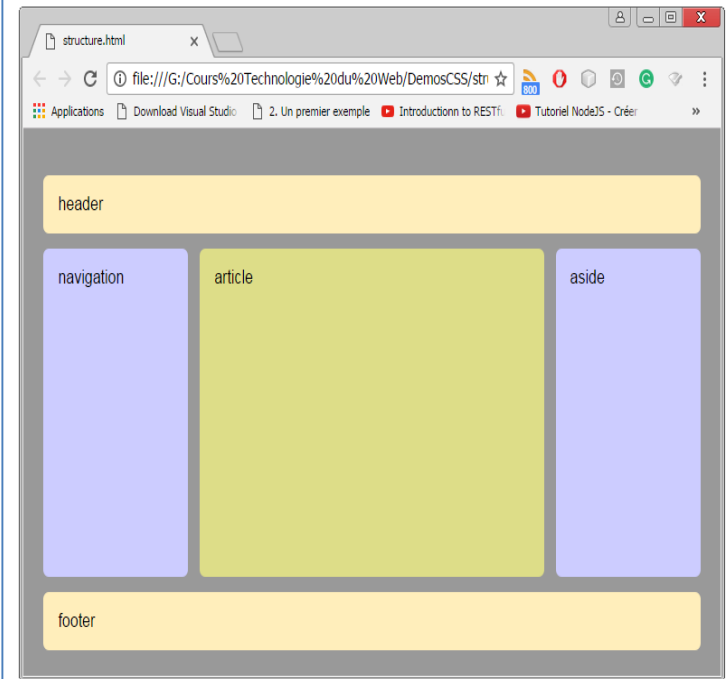
### *Organisation de base*

```
<!DOCTYPE html>
<html>
<head>
<link rel="stylesheet" href="base.css">
</head>
<body>
<header>Header</header>
<section>
<nav>Navigation</nav>
<article>Article</article>
<aside>Aside</aside>
</section>
<footer>Footer</footer>
</body>
</html>
```

```
body{
margin:0;
background: #999;
}
header,footer,nav,aside,article{
margin: .4em;
padding: 1em;
border-radius: 6px;
}
header,footer{ background: #ffeebb; }
nav,aside{background: #ccccff;}
article{background: #ddd888;}
```



### *Organisation souhaitée*



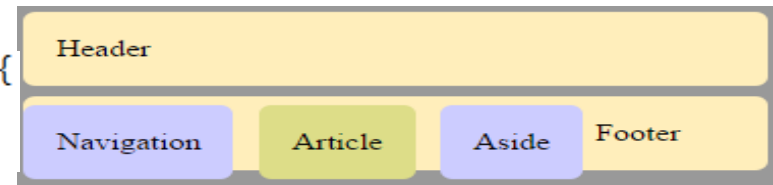
.... la suite

- Flexbox présent 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, overflow, etc...

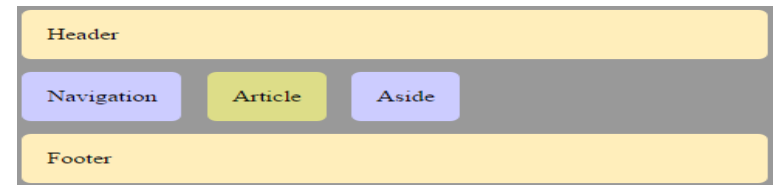
```
nav,article,aside{
float: left;
}
```



### inline-block

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

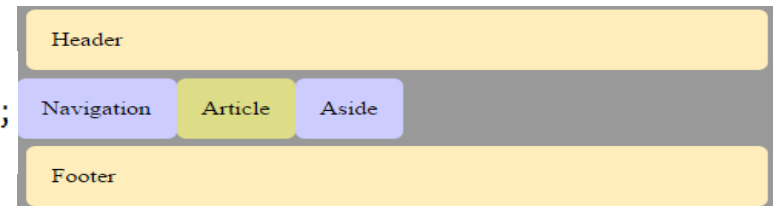
```
nav,article,aside{
display: inline-block;
}
```



### 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;
}
```



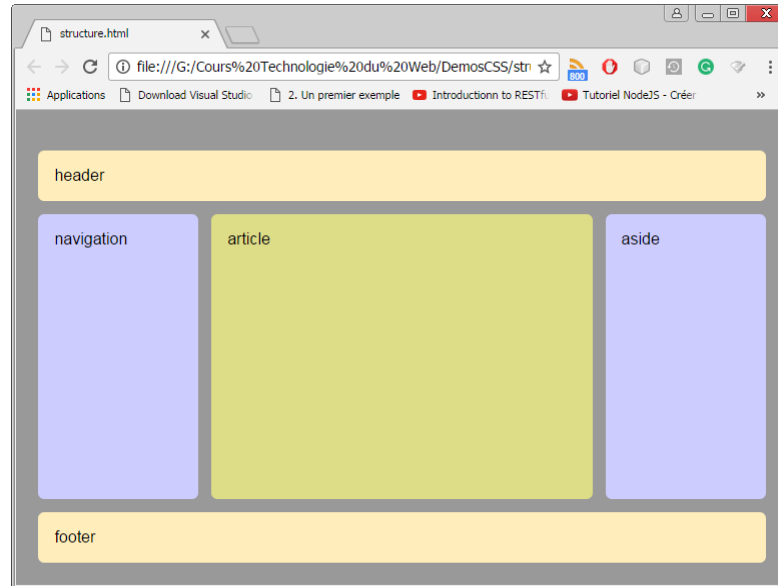
## 2. CSS3 Flexbox

---

- ▶ **Flexbox** (pour flexible box) est un nouveau mode de positionnement des éléments au sein d'une page Web.
- ▶ Il permet de créer des mises en page **flexibles et adaptables** à différents dispositifs et tailles d'écran.
- ▶ Il offre aux éléments une grande flexibilité en leur permettant de s'adapter à l'espace disponible en termes de dimensions.
- ▶ 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.**

### 3. Flexbox: mise en ouvre

- ▶ On va se servir de l'exemple précédent pour montrer comment utiliser *flexbox* pour réaliser la mise en page suivante :



- ▶ Pour rendre les éléments du document flexibles, la première étape consiste à **activer la propriété flex sur leur conteneur**.
- ▶ Dans notre cas, nous avons deux conteneurs à considérer : le premier est `<body>`, qui agit en tant que conteneur principal, et le deuxième est `<section>`, servant de conteneur pour les éléments `<nav>`, `<article>` et `<aside>`. L'objectif est de rendre ces éléments flexibles."

.... la suite

- Pour ce faire, il faut d'abord, **activer** flex (*display:flex*) **au niveau de conteneur body** pour donner un contexte flexible à tout ses éléments directs. Par défaut, les éléments flex sont alignés en ligne horizontale, de gauche à droite.
- Pour un alignement vertical, il faut utiliser la propriété *flex-direction* en lui associant la valeur **column**.

```
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 */
}
```

.... la suite

- ▶ 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 restant, il faut lui rajouter la propriété *flex* avec valeur égale à 1. En fait, la propriété CSS "flex" permet de partager l'espace restant avec des proportions différentes, exprimées en pourcentages ou en unités de mesure.

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



## 4. Flexbox: propriétés

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

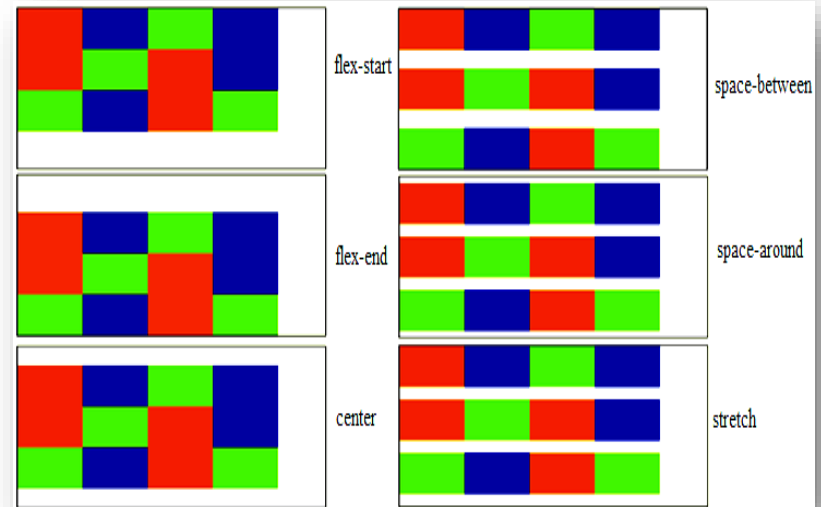
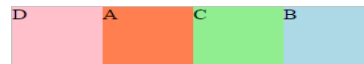
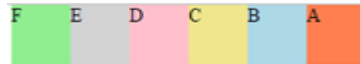
► Examples :

```
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;}

```



# V. Responsive Web design

## 1. Présentation

---

- ▶ 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 évidemment plus coûteuse au niveau conception, développement et maintenance.

- 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.

## 2. Les media 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 tant, 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.

### 3. Media queries: mise en application

---

► Il y a deux façons de les utiliser :

- En chargeant une feuille de style .css différente en fonction de la règle (par exemple : si la résolution est inférieure à 800px de large, charge le fichier «small.css»), 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'écran*/) {
 /* les propriétés CSS ici */
}
```

Exemple :

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

## 4. Media queries: règles

---

- ▶ Il existe de nombreuses règles permettant de construire des media queries :
  - ▶ **Règles précisant le type d'écran** : Il est 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 précédés par @media : **screen** : écran « classique » ; **handheld** : périphérique mobile ; **print** : impression ; **tv** : télévision ; **projection** : projecteur ; **all** : tous les types d'écran.
  - ▶ **Autres règles** :
    - ▶ **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/landscape).
    - ▶ **color** : gestion de la couleur (en bits/pixel).
- ▶ Les règles peuvent être combinées à l'aide des mots suivants : **and** : « et » ; **not** : « non », **only** : « uniquement » ; . Exemple : @media tv and (min-width: 1280px)

## 5. 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;
 }
}
```

.... la suite

- **Exemple 2** : on va utiliser l'exemple présenté dans la section dédiée à Flexbox. L'objectif est d'adapter la mise en page créée pour les petits appareils tels que les smartphones ou les tablettes.



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

```
@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 */
 }
}
```



# VI. Bootstrap

## 1. Présentation

---

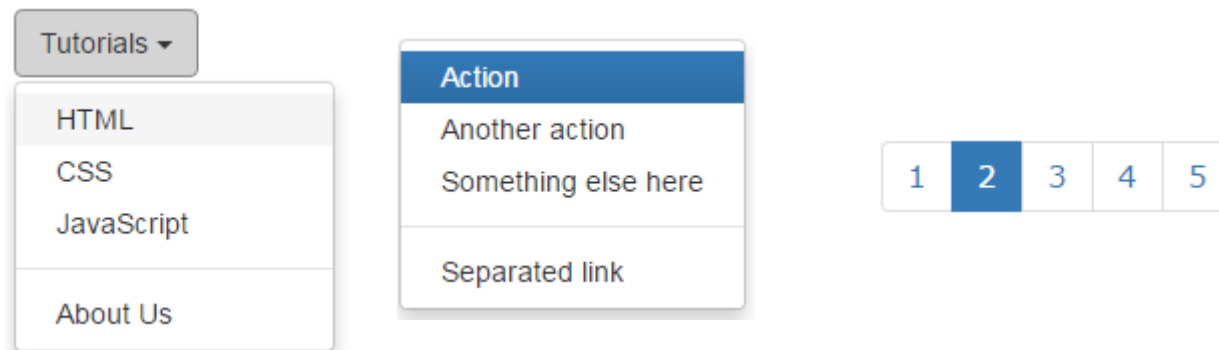
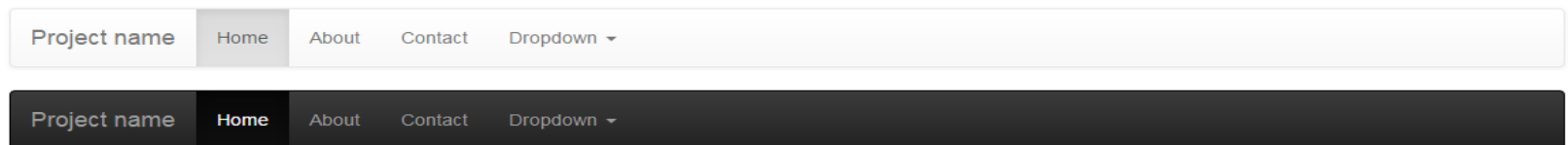
- ▶ **Bootstrap** est une bibliothèque gratuit et open-source **contenant un ensemble d'éléments de feuilles de style prédéfinis** utilisée coté front-end pour faciliter la conception de sites Web et d'applications Web.
- ▶ Il s'agit d'un ensemble des définitions de style de base pour tous les éléments 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 de applications web adaptatives(responsive)**. En effet, les pages web développés à base de bootstrap s'adaptent dynamiquement au format des supports sur lesquels ils sont consultés (PC, tablette, smartphone).

.... la suite

## ► Des exemples de style BS



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

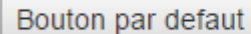


## 2. Mise en ouvre

- ▶ L'ensemble de 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 défaut </button>
```

```
<button type="button" class="btn"> Bouton style bootstrap </button>
```



- ▶ Les différents 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](https://getbootstrap.com/).
- ▶ Il existe des centaines de style prédéfinis dans Bootstrap (<http://getbootstrap.com/components/> ).

### 3. style BS pour boutons

---

- ▶ Bootstrap propose 9 styles 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>`



## 4. style BS pour 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

## 5. style BS pour 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:



## 6. style BS pour 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>
```

Email:  Password:  ☐ Remember me

## 7. style BS pour le menu de navigation

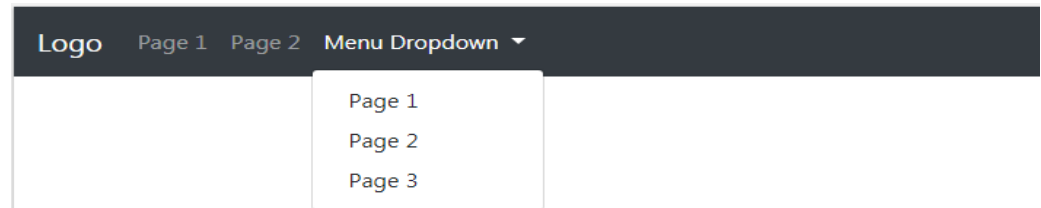
- ▶ Grâce à BS on peut créer des menus de navigation avec des styles différents.
- ▶ Le style de base est `.nav`. Pour un menu positionné en entête de page, il y a le style `.navbar`.
- ▶ Un exemple :

```
<nav class="navbar navbar-expand-sm bg-dark navbar-dark">
 <!-- Brand -->
 Logo
 <!-- Links -->
 <ul class="navbar-nav">
 <li class="nav-item"> Page 1
 <li class="nav-item"> Page 2
 <!-- Dropdown -->
 <li class="nav-item dropdown">

 Menu Dropdown

 <div class="dropdown-menu">
 Page 1
 Page 2
 Page 3
 </div>

</nav>
```





## 8. 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 une ou plusieurs cellules pour chaque élément, comme illustré dans la figure suivante :

		Élément 1									

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

.... la suite

`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.

- **Exemple** : le code suivant présent 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.



- ▶ 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/](http://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.