

# COURS HTML/CSS

- HTML 5
- CSS
- Framework CSS : Bootstrap 5

- Spécification : <https://html.spec.whatwg.org/>

```
<!DOCTYPE html>
<html lang="fr">
  <head>
    <meta charset="UTF-8">
    <title></title>
    <meta name="description" content="">
  </head>
  <body>
</body>
</html>
```

- Attribut « lang » peut être assigné à un élément particulier
  - `<p lang="en" > English </p>`
- La balise head : la balise contient plusieurs types d'éléments:
  - L'encodage avec la balise meta ou charset.  
`<meta charset="UTF-8">`
  - Le titre de la page.  
`<title > MaPage /title>`
  - Les liens avec la balise link.
    - Favicon `<link rel="icon" type="image/gif" href="/favicon.gif" />`
    - Feuille de style CSS `<link rel="stylesheet" type="text/css" href="style.css">`
  - Et d'autres indications par des métas.  
`<meta name="description" content="">`

- ❑ `<header>` : contient une introduction pour une partie de la page ou la page entière.
- ❑ `<footer>` : contient des informations qui se placent à la fin d'une section. On peut le placer en bas d'une section ou de la page, mais aussi n'importe où dans la section. Par exemple cela contient un lien sur l'index, qui l'on peut placer sous le titre.
- ❑ `<nav>` Ce conteneur est destiné à enclore un groupe de liens.
- ❑ `<main>` : désigne le contenu principal de la page. On conseille d'utiliser une seule balise main dans une page mais on peut avoir plusieurs à condition d'en afficher qu'une seule à la fois (en utilisant l'attribut hidden)
- ❑ `<article>` : désigne un contenu typique que l'on peut retrouver sur différentes pages, ou même différents sites. Cela peut être un post de forum, un article de presse et cela s'adresse à des outils qui pourront les extraire plus facilement (en les séparant du contenu inutile tel que menus de navigation).
- ❑ `<aside>` : représente un contenu annexe au contenu proprement dit et peut définir une barre latérale.

## □ Découpage plus précis

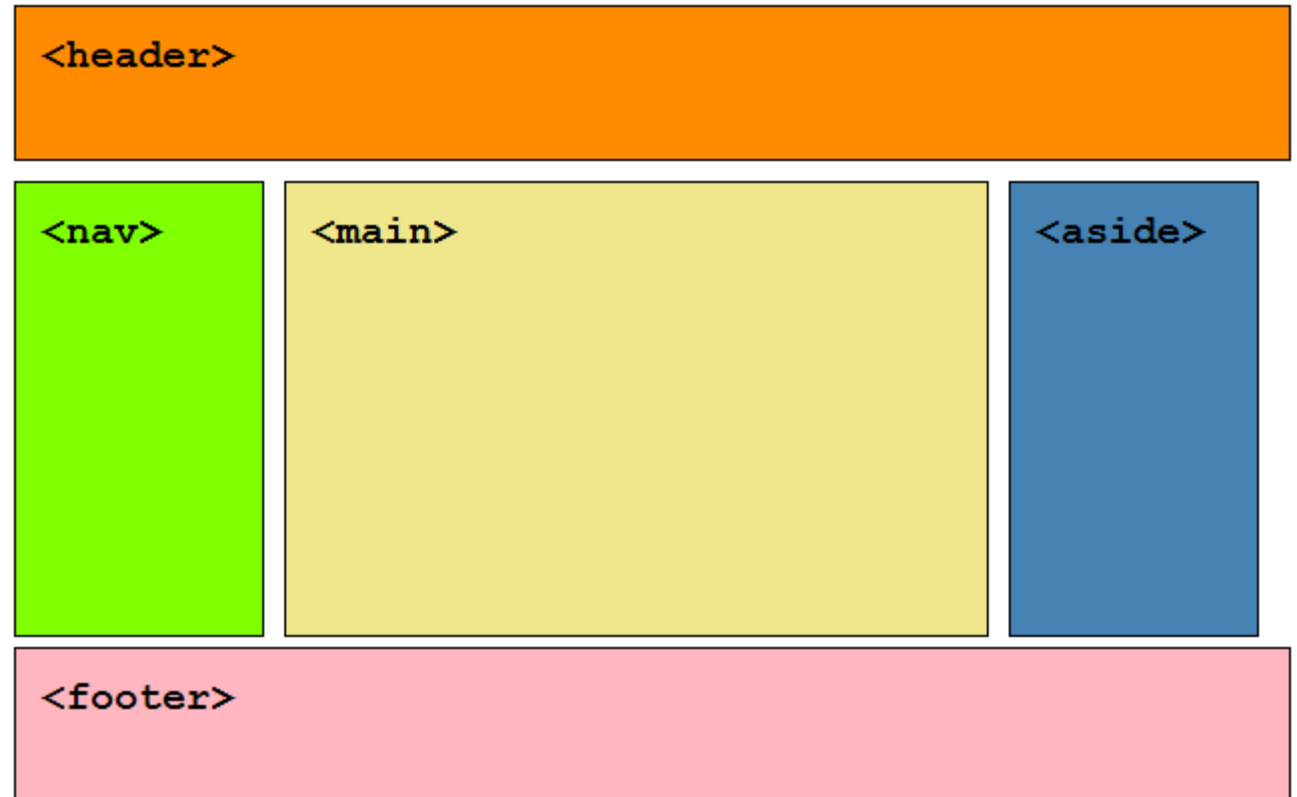
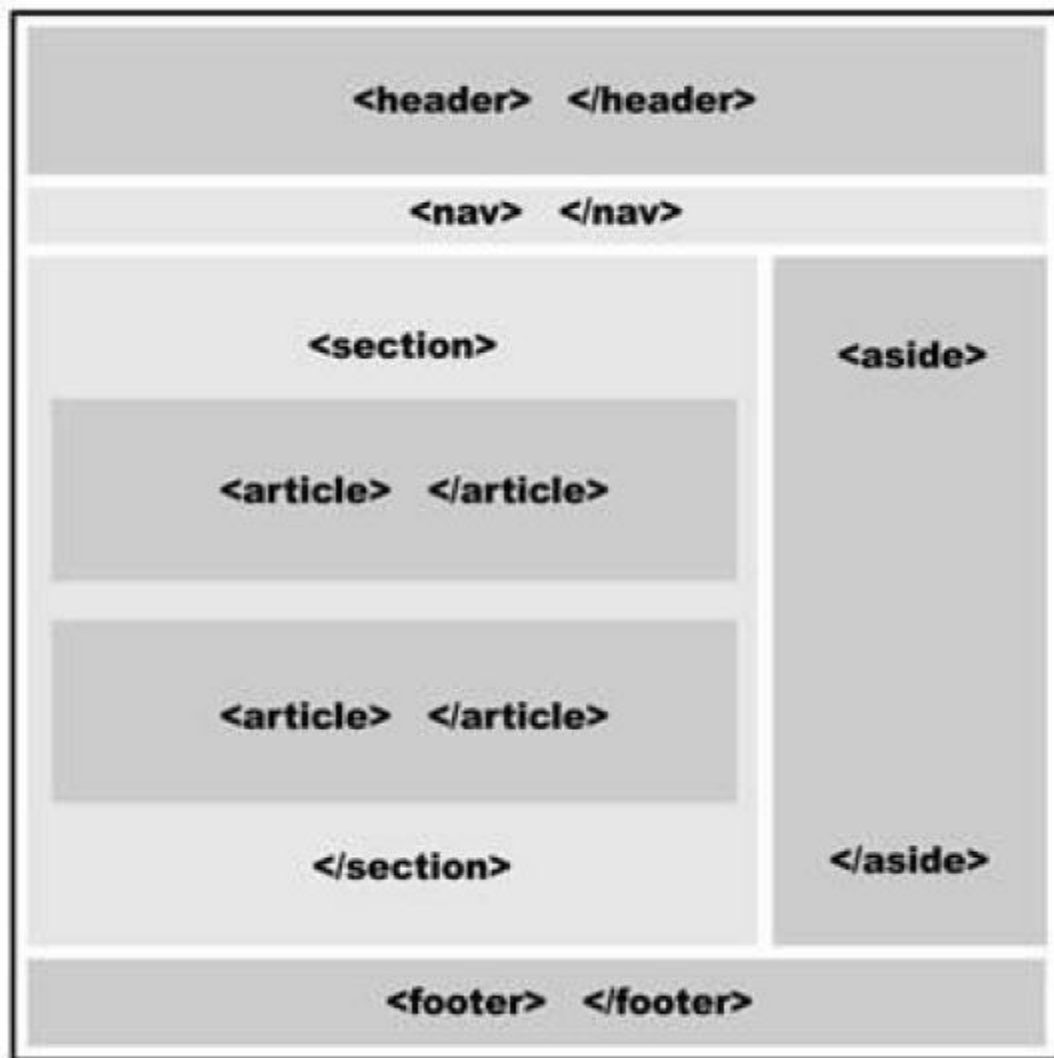
- `<section>` : les sections délimitent les parties du contenu. Il appartient alors au webmaster de leur associer une feuille de style ou de les utiliser dynamiquement dans des scripts. Très basiquement, on peut encadrer une section par une bordure, ou la séparer de ce qui précède par un espace.

## □ Autres balises sémantiques

- `<address>` : contient des information de contact, par exemple le nom de l'auteur.
- `<mark>` : permet de marquer une partie d'un texte, le mettre en relief, comme les anciennes balises `<strong>` mais en plus général.
- `<time datetime = "14-03-20213 "> Mardi 15 Mars</time>`
- `<data value="1234"> c'est une donnée qui a la valeur 1234</data>`

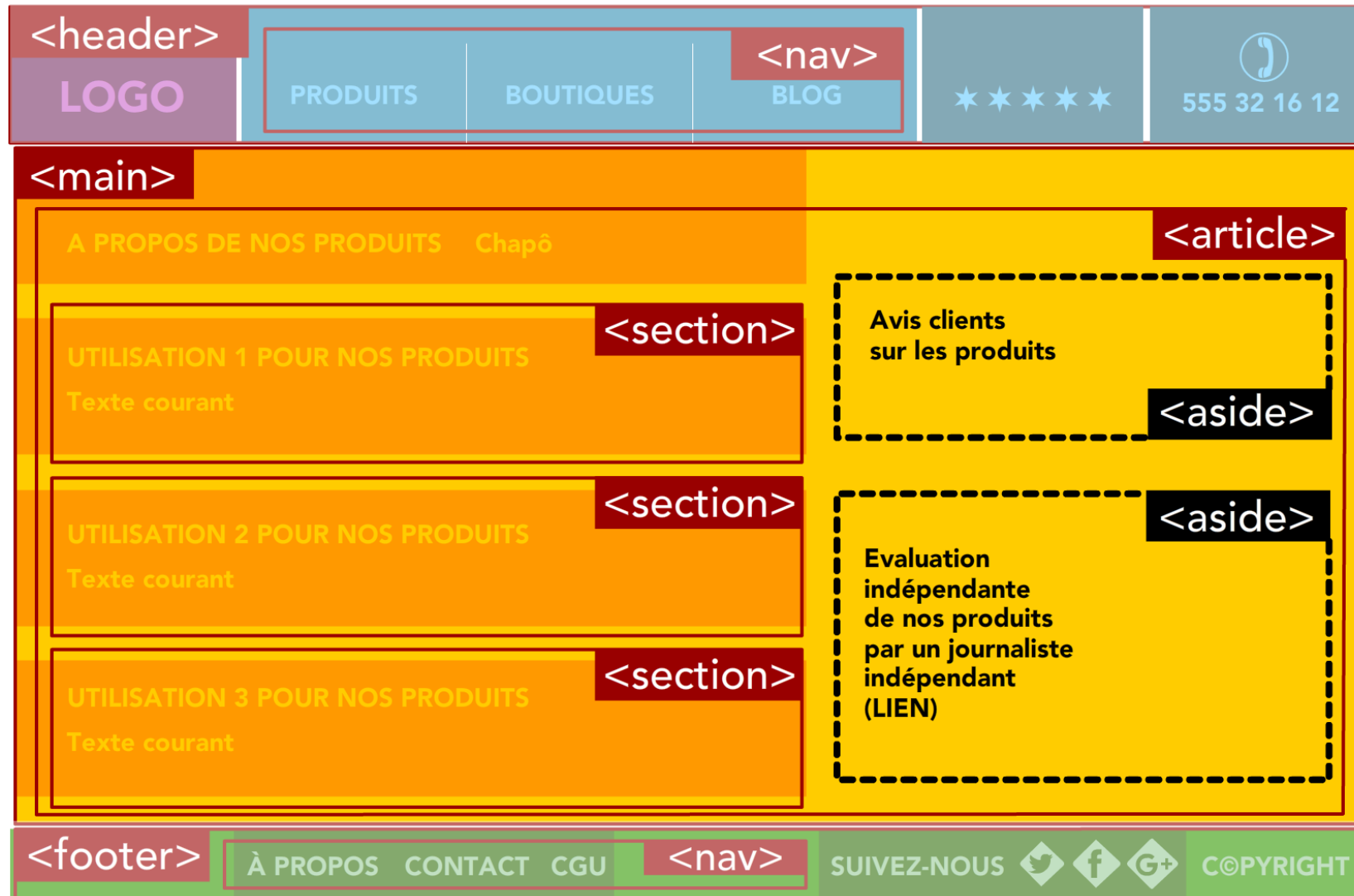
# Exemples d'utilisation des balises sémantiques

7



# Exemples d'utilisation des balises sémantiques

8





## □ Les balises du formulaire :

- balise d'ouverture `<FORM>` et fermeture `</FORM>` :
- La spécification de la méthode employée pour l'envoi d'informations au serveur + l'adresse de destination des informations (`<form method="get" action="test.php">.. </form>`)

### **Les Balises :**

**La balise `<input>`**

**La balise `<select>`**

**La balise `<option>`**

**La balise `<textarea>`**

### **Les champs :**

**Champ de texte**

**Zone de texte**

**Boutons radios**

**Cases à cocher**

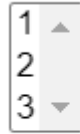
**Liste déroulante**

**Liste ouverte**

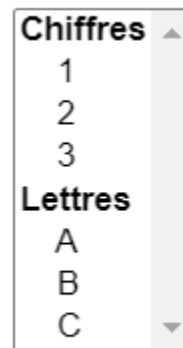
# L'élément select d'un formulaire

10

```
<select name="select" size="3">  
  <option> 1 </option>  
  <option> 2 </option>  
  <option> 3 </option>  
</select>
```



```
<select name="select" size="8">  
  <optgroup label="Chiffres">  
    <option> 1 </option>  
    <option> 2 </option>  
    <option> 3 </option>  
  </optgroup>  
  <optgroup label="Lettres">  
    <option> A </option>  
    <option> B </option>  
    <option> C </option>  
  </optgroup>  
</select>
```

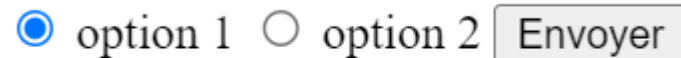


- Input type=radio

- ▣ `<input type="radio" name="x" value="" checked >`
- ▣ `<input type="radio" name="x" value="" >`

- Exemple

```
<form name="form1" method="post" action="script.php">  
  <label> <input type="radio" name="x" value="un" checked > option 1 </label>  
  <label> <input type="radio" name="x" value="deux" > option 2 </label>  
  <input type="submit" value="Envoyer">  
</form>
```



- côté client

- ▣ `var element = document.form1.x;`

`<Input type = "?" />`

- Type="text"
  - ▣ Attributs : minlength, maxlength, size, ...
- Type = "number"
  - ▣ Attributs : min, max, ...
- Type="date"
  - ▣ Attributs : min, max, ...
- Autres : password, email, url, image, hidden, file, etc.

```
<table>
  <caption>Libellé</caption>
  <thead>
    <tr>
      <th>Titre 1
      <th>Titre 2
  <tfoot> Bas de page
  <tbody>
    <tr>
      <th>Titre de ligne
      <td>Contenu de cellule
    ...
  </table>
```

14

CSS

## Objectif : simplifier la présentation/design des pages web

### Les avantages de CSS :

- La **structure et la présentation** sont gérées séparément
- Présenter de façon homogène,
- Positionner rigoureusement les éléments,
- Le code HTML est **allégé** et gagne en lisibilité.

## □ Balise <style>

### ▣ attributs :

- type="..." : type de contenu Internet
- media="..." : définit le média de destination ( screen, print, projection, braille, speech, all)
- title="..." : titre de la feuille de styles

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

```
<!--
```

```
.....
```

```
-->
```

```
</STYLE>
```



## CSS - les propriétés :

- ▣ Propriétés de mise en forme des polices
- ▣ Propriétés de mise en forme de texte
- ▣ Propriétés des couleurs de texte et de fonds de pages
- ▣ Propriétés de mise en forme des paragraphes
- ▣ Propriétés des bordures de la boîte BORDER
- ▣ Propriétés de marge externe MARGIN
- ▣ Propriétés de marge interne PADDING
- ▣ Propriétés de LIST-STYLE

Les détails se trouvent dans  
: <https://www.w3.org/TR/CSS/#properties>

- Voici un exemple de règle CSS permettant d'afficher les entêtes principaux (H1) en bleu :

```
H1 { color : blue }
```

- Une règle CSS est composée de 2 parties : un sélecteur (ici H1) et une déclaration (color:blue).
- Une déclaration à elle-même deux parties : une propriété (color) et une valeur (blue).

```
H1, P { color : red }
```

```
P { margin-left : 1cm ; text-style : italic }
```

- Le code CSS peut être placé à 3 positions différentes dans la page.
  - ▣ Dans l'élément HTML lui-même : attribut style
  - ▣ Dans la page HTML : balise style
  - ▣ Dans un fichier indépendant

# Placer une feuille de style directement dans la balise HTML

20

```
<HTML>
  <HEAD>
    <TITLE>
      CCSinline
    </TITLE>
  </HEAD>
  <BODY>
    <p style="font-style: italic; size: 1.5em;">
      CSS directement dans la balise HTML </p>
  </BODY>
</HTML>
```



*CSS directement dans la balise HTML*

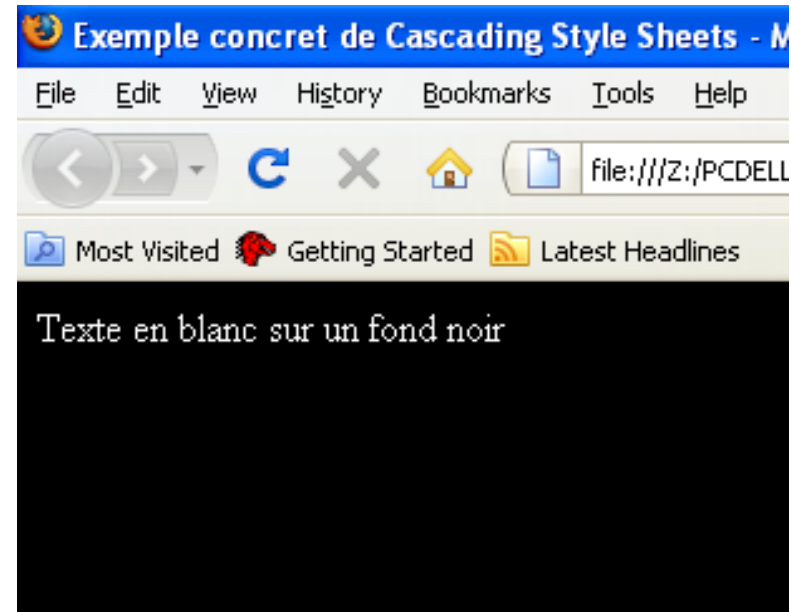
**Cette méthode est à éviter** (cache visibilité du code HTML)

# Placer une feuille de style dans l'entête de la page HTML

21

- Dans le code dans la page HTML, entre les deux balises <head> et </head>.

```
<HTML>
<HEAD>
  <TITLE>Exemple concret de Cascading Style Sheets</TITLE>
  <STYLE TYPE="text/css">
    <!--
      BODY {
        color: white;
        background: black;
      }
    -->
  </STYLE>
</HEAD>
<BODY>
  Texte en blanc sur un fond noir
</BODY>
</HTML>
```



## Placer une feuille de style dans un fichier séparé

22

- Syntaxe: `<link rel="stylesheet" type="text/css" href="style.css" />`
- La méthode "**<link href=...**" permet aussi de mettre en place des feuilles de styles destinées aux différents medias (imprimante, navigateurs de PDA, *etc.*).
  - ▣ screen, projection, print, all...
- Comment ça marche : Il s'agit de placer la feuille de style dans un fichier séparé, et à y faire référence dans l'entête du document.

```
...<head>
<title><title>
<link rel="stylesheet" type="text/css" href="style.css" media="screen, projection"
/>
</head>
<body> ...
```

Fichier CSS (style.css) :

```
H1, H2, { color : red; font : bold} /* Les titres principaux sont en rouge */
H3, H4, H5 { color : green} /* Les titres secondaires sont en vert */
```

Méthode à utiliser dans le projet

# Placer une feuille de style par importation

**La règle @import** : @import est une propriété CSS2 qui doit être suivie de l'URL d'un fichier qui contiendra des styles à appliquer en plus de la feuille de style en cours.

Syntaxe:

```
<style type="text/css">  
@import url(styles/affichage.css) media;  
</style>
```

*Cela peut être utile pour importer des feuilles de style dans d'autres feuilles de style.*

- permettent de choisir entre plusieurs types de présentations pour un même élément HTML.
  - Exemple : on peut avoir les paragraphes d'entête, des remarques et des paragraphes normaux.
- l'attribut class, qui peut être appliqué à tous les éléments HTML, permet de préciser des classes d'éléments. On peut alors préciser les styles qui doivent s'appliquer à chaque classe.

```
.rouge { color: red }  
P.entete { font-style: italic }
```



```
<HTML>
  <HEAD>
    <TITLE> Les classes </TITLE>
    <STYLE type="text/css">
      <!--
      .rouge { color: red }
      P.entete { color: green; font-style: italic }
      //-->
    </STYLE>
  </HEAD>
  <BODY>
    <H2 class=rouge> Un titre rouge </H2>
    <P class=entete> Un titre italic en vert </P>
  </BODY>
</HTML>
```

**Un titre rouge**

*Un titre italic en vert*

Exemple :

```
#menu { background-color:silver;  
width:100px; float:left; }  
#contenu { margin-left:110px; }
```

- ***un id s'applique à un objet unique : il ne peut pas y avoir deux mêmes id dans une page***
- ***une classe peut caractériser plusieurs objets (identiques ou non)***

```
<h1 id="haut">Exemple des sélecteurs "class" et  
"id"</h1>  
  
<div id="menu">  
<ul>  
    <li>item 1</li>  
    <li>item 2</li>  
    <li>item 3</li>  
</ul>  
</div>  
  
<div id="contenu">  
    <p>  
    text.  
    </p>  
  
    <p class="haut">  
    <a href="#haut">Haut de page</a>  
    </p>  
</div>
```

- Il est possible de préciser les propriétés qui doivent s'appliquer à un élément dans le cas où celui est utilisé à l'intérieur d'un autre élément (et pas ailleurs).
  - ▣ H2 EM {color : green}
- Dans le code HTML, on peut utiliser l'élément EM dans un entête H2 ou pas :
  - ▣ `<H2>Un titre <EM>mis en valeur</EM></H2>`  
`<P>Du texte <EM>mis en valeur</EM>`

```
<HTML>
<HEAD>
  <TITLE> Contextes </TITLE>
  <STYLE type="text/css">
    <!--
    H2 EM {color : green}
    //-->
  </STYLE>
</HEAD>
<BODY>
<H2>Un titre <EM>mis en valeur</EM></H2>
<P>Du texte <EM>mis en valeur</EM>
  </P>
</BODY>
</HTML>
```

Un titre ***mis en valeur***

Du texte *mis en valeur*

## □ Les pseudo-classes d'ancre. :link et :visited

- CSS permet de représenter différemment les liens qui n'ont pas été visités de ceux qui l'ont déjà été au travers des pseudo-classes ':link' et ':visited'.

- Ex. `A:link { color : fushia; }` `A:visited { color : olive; }` `A:active { color : green; }`
- La pseudo-classe :link s'applique aux liens qui n'ont pas été visités ;
- La pseudo-classe :visited s'applique lorsque le lien a été visité par l'utilisateur.
- La pseudo-classe :active permet de cibler un élément lorsque celui-ci est activé par l'utilisateur.

## exemple

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML
4.01 Transitional//EN">
<html>
  <head>
    <link rel="stylesheet" href="P.css"
type="text/css">
    <title>CSS Pseudo Classes</title>
  </head>
  <body>
    <td>
      <a href="http://www.utc.fr">site UTC</a>
    </td>
  </body>
</html>
```

```
a:link
{
background-color: YellowGreen;
color: white;
font-family: Arial;
font-size: .8em;
}
a:visited
{
background-color: red;
color: white;
font-family: Arial;
font-size: .8em;
}
a:hover
{
background-color: cyan;
color: blue;
font-family: Arial;
font-size: .8em;
}
```

- Les pseudo-éléments créent des abstractions dans l'arbre en plus des éléments déjà spécifiés par le langage du document.
- Ainsi, certains langages n'offrent pas de mécanismes de correspondance avec la première lettre ou la première ligne du contenu d'un élément. Les pseudo-éléments de CSS permettent aux auteurs d'y accéder.

- P:first-letter { font-size: 200%;}
- P:first-line { font-variant:small-caps; }

```
<HTML>
  <HEAD>
    <TITLE>Lettrine et première ligne</TITLE>
    <STYLE type="text/css">
      <!--
        P                { color: red; font-size: 12pt }
        P:first-letter { color: green; font-size: 200%; font-weight: 900 }
        P:first-line   { color: blue }
      //-->
    </STYLE>
  </HEAD>
  <BODY>
    <P>
      Voici paragraphe dont la première lettre est verte et grande,
      dont le texte de la première ligne est bleu et les autres lignes
      sont rouges.
    </P>
  </BODY>
</HTML>
```



# Positionner des éléments grâce aux CSS

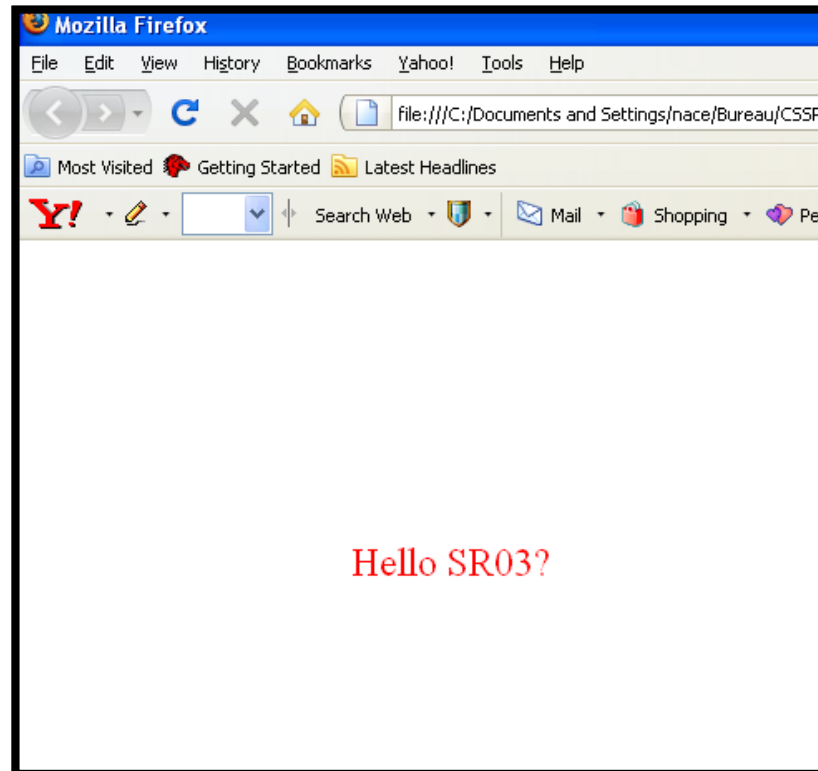
33

- Il est possible grâce aux feuilles de style de positionner au pixel près du texte ou des images grâce aux balises `<SPAN>` (de type inline) et `<DIV>` (de type block).
  
- **Positionnement relatif et absolu**
  - Le positionnement absolu {position: absolute} se détermine par rapport au coin supérieur gauche de la fenêtre du navigateur. Les coordonnées d'un point s'expriment alors de haut en bas (top) et de gauche à droite (left).
  
  - La position relative se fait par rapport à d'autres éléments, c'est-à-dire que les éléments contenus dans la balises *DIV* ou *SPAN* seront positionnés à la suite des éléments HTML après lesquels ils se trouvent.

# Positionner des éléments grâce aux CSS

34

```
<HTML>
<HEAD>
<STYLE>
<!--
.pos{position: absolute; top: 180px; left: 200px; color: red; font-size: x-large}
-->
</STYLE>
</HEAD>
<BODY>
<DIV class=pos>
Hello SR03?
</DIV>
</BODY>
</HTML>
```



- Les propriétés des CSS peuvent être définies plusieurs fois. C'est toujours la dernière définition qui compte. Cela permet d'importer plusieurs feuilles de styles, et de redéfinir certains styles dans le document.

```
<body>
  <div>
    <p>
      bla bla bla
    </p>
  </div>
</body>
```

- Supposons qu'on dispose d'une première feuille de style, que nous appellerons style1.css qui contienne les propriétés suivantes :
  - H1 { color : red; font-size : 48pt }
  - H2 { color : blue; font-size : 12pt }
- Nous utilisons aussi une autre feuille de style, nommée style2.css et contenant les propriétés suivantes :
  - H2 {color : green; }
  - H3 {color : pink; font-size : 12pt }

- Dans une page donnée, nous incluons dans l'entête l'appel de ces deux feuilles, ainsi que la définition d'autres propriétés.

```
<HEAD>
<TITLE>...</TITLE>
<LINK rel=STYLESHEET href="style1.css" type="text/css">
<LINK rel=STYLESHEET href="style2.css" type="text/css">
<STYLE type="text/css">
<!--
H1 { color : fushia; } H2 { font-size : 16pt; } H3 { font-size : 14pt; } -->
</STYLE>
</HEAD>
```

- Déterminons les valeurs utilisées pour ce document :
  - H1 : fushia, 48 points
  - H2 : vert, 16 points
  - H3 : rose, 14 points

- Pour déterminer la valeur d'une propriété, on dispose donc de la notion de cascade. Dans les cas où la propriété n'a pas été définie, deux possibilités se présentent :
  - ▣ Ou bien la propriété est dite "héritée". Dans ce cas, c'est la valeur de l'élément "parent", c'est à dire de l'élément dans lequel est utilisé l'élément courant.
  - ▣ Dans l'autre cas, on prend la valeur par default.

**A retenir : la dernière règle lue est prioritaire!!**

- CSS3 est modulaire. Le degré d'avancement de CSS3 varie selon les modules et le degré de priorité qui leur a été donné par le groupe de travail CSS
  - ▣ Il offre de nouvelles fonctionnalités pour améliorer le design des sites web, en particulier:
    - Les arrières plans et les bordures
    - Les effets de texte
    - La mise en page multi-colonnes...
    - Disposition des boîtes flexibles CSS
- CSS4 est en cours...

This is an example of a box with rounded borders

This is an example of a box with gradient border

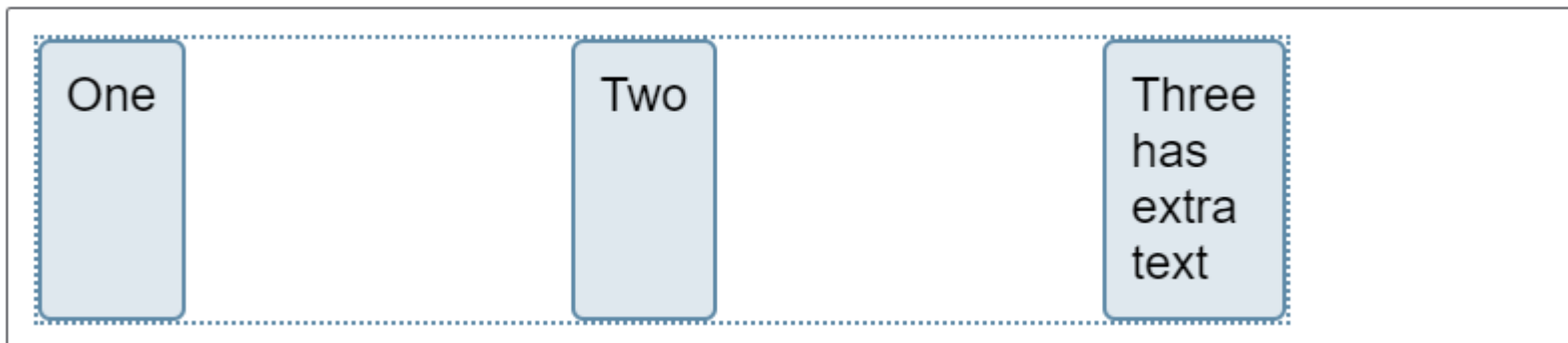
This is an example of a box with a drop shadow

This is an example of text with a shadow applied

Le module des boîtes flexibles, aussi appelé « flexbox », a été conçu comme un modèle de disposition unidimensionnel et comme une méthode permettant de distribuer l'espace entre des objets d'une interface ainsi que de les aligner.

```
<div class="box">
  <div>One</div>
  <div>Two</div>
  <div>Three
    <br>has
    <br>extra
    <br>text
  </div>
</div>
```

```
.box {
  display: flex;
  justify-content: space-between;
}
```







- Il existe plusieurs
  - ▣ Bootstrap => plusieurs versions 3, 4, 5
  - ▣ W3.CSS
  - ▣ Tailwind
  - ▣ Materialize
  - ▣ Etc.

- ❑ Est un framework qui permet de créer rapidement et facilement des pages web
- ❑ Permet de créer des éléments graphiques avancés à l'aide de html et des modèles css : typographie, formulaire, boutons, etc.
- ❑ Fournit également une partie JS via des plugins qui sont optionnels.
- ❑ Facilite la création de sites web responsives

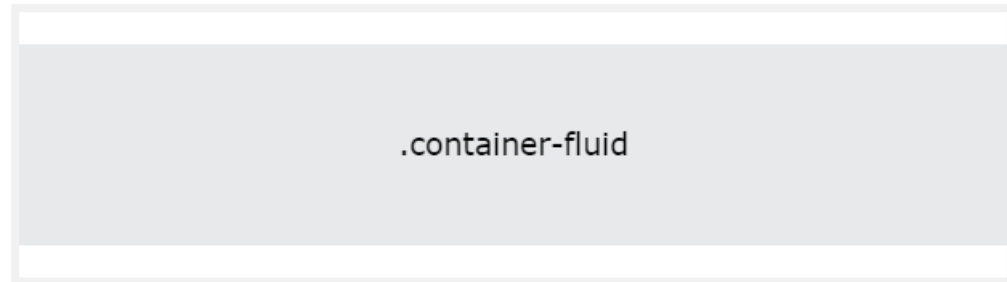
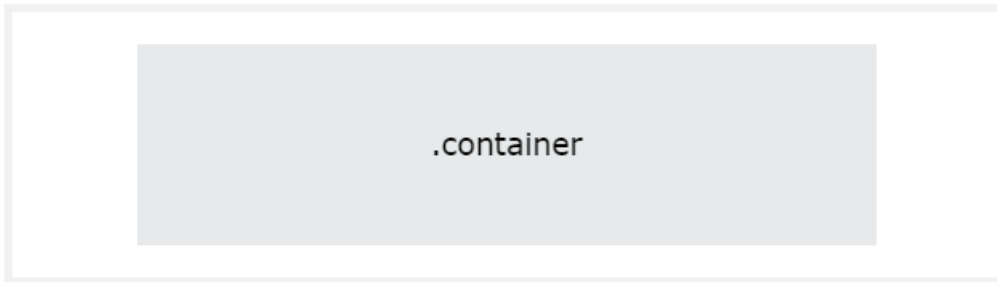
```
<!-- Latest compiled and minified CSS -->
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/css/bootstrap.min.css" rel="stylesheet">

<!-- Latest compiled JavaScript -->
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/js/bootstrap.bundle.min.js"></script>
```

- CDN permet de récupérer le css et js de bootstrap sans les charger sur votre serveur
  - ▣ Les fichiers seront téléchargés sur la machine cliente selon sa localisation et stockés dans le cache=> optimisation des performances

important de le les utiliser (perf ++) : au lieu d'importer le fichier CSS (décharge site web)

- ❑ `class=container`
  - ▣ Pour créer un conteneur responsive de largeur fixe
- ❑ `Class=.container-fluid`
  - ▣ Pour créer un conteneur pleine largeur, qui s'étendra toujours sur toute la largeur de l'écran (la largeur est toujours de 100 %) :



- ❑ Exemple : marges, bordure et couleur de l'arrière plan

```
<div class="container p-5 my-5 border"></div>
```

```
<div class="container p-5 my-5 bg-dark text-white"></div>
```

```
<div class="container p-5 my-5 bg-primary text-white"></div>
```

```
<div class="container-fluid mt-3">  
  <h1>Three equal width columns</h1>  
  <p>Note: Try to add a new div with class="col" inside the row class - this will  
create four equal-width columns.</p>  
  <div class="row">  
    <div class="col p-3 bg-primary text-white">.col</div>  
    <div class="col p-3 bg-dark text-white">.col</div>  
    <div class="col p-3 bg-primary text-white">.col</div>  
  </div>  
</div>
```



- Ce système est construit avec flexbox et permet jusqu'à 12 colonnes sur la page
- Classe de grilles : il existe 6 classes qui sont adaptées à chaque taille d'écran sinon avec la classe « col » on laisse bootstrap gérer d'une manière automatique.

- En-tête bootstrap : h1, h2, ....h6:
  - ▣ `<p class=h1> .....</p>`
- En-têtes d’affichage : sont utilisés pour se démarquer davantage que les en-têtes normaux
  - ▣ class .display-1 to .display-6
- Utiliser l’élément mark pour mettre du texte en surbrillance
- Etc.

## □ Table basique

```
<table class="table">
  <thead>
    <tr>
      <th>Firstname</th>
      <th>Lastname</th>
      <th>Email</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>John</td>
      <td>Doe</td>
      <td>john@example.com</td>
    </tr>
    <tr>
      <td>Mary</td>
      <td>Moe</td>
      <td>mary@example.com</td>
    </tr>
    <tr>
      <td>July</td>
      <td>Dooley</td>
      <td>july@example.com</td>
    </tr>
  </tbody>
</table>
```

Firstname	Lastname	Email
John	Doe	john@example.com
Mary	Moe	mary@example.com
July	Dooley	july@example.com



Logo Link Link Link

```
<nav class="navbar navbar-expand-sm navbar-dark bg-dark">
  <div class="container-fluid">
    <a class="navbar-brand" href="javascript:void(0)">Logo</a>
    <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#mynavbar">
      <span class="navbar-toggler-icon"></span>
    </button>
    <div class="collapse navbar-collapse" id="mynavbar">
      <ul class="navbar-nav me-auto">
        <li class="nav-item">
          <a class="nav-link" href="javascript:void(0)">Link</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="javascript:void(0)">Link</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="javascript:void(0)">Link</a>
        </li>
      </ul>
      <form class="d-flex">
        <input class="form-control me-2" type="text" placeholder="Search">
        <button class="btn btn-primary" type="button">Search</button>
      </form>
    </div>
  </div>
</nav>
```

```
<div class="container mt-3">
  <h2> form</h2>
  <form action="/action_page.php">
    <div class="mb-3 mt-3">
      <label for="email">Email:</label>
      <input type="email" class="form-control" id="email" placeholder="Enter email" name="email">
    </div>
    <div class="mb-3">
      <label for="pwd">Password:</label>
      <input type="password" class="form-control" id="pwd" placeholder="Enter password" name="pswd">
    </div>
    <div class="form-check mb-3">
      <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>
</div>
```

## form

Email:

Password:

☐ Remember me

Username:



Please fill out this field.

Password:



Please fill out this field.

☐ I agree on blabla.

Check this checkbox to continue.

```
<form action="/action_page.php" class="was-validated">
  <div class="mb-3 mt-3">
    <label for="uname" class="form-label">Username:</label>
    <input type="text" class="form-control" id="uname" placeholder="Enter username" name="uname" required>
    <div class="valid-feedback">Valid.</div>
    <div class="invalid-feedback">Please fill out this field.</div>
  </div>
  <div class="mb-3">
    <label for="pwd" class="form-label">Password:</label>
    <input type="password" class="form-control" id="pwd" placeholder="Enter password" name="pswd" required>
    <div class="valid-feedback">Valid.</div>
    <div class="invalid-feedback">Please fill out this field.</div>
  </div>
  <div class="form-check mb-3">
    <input class="form-check-input" type="checkbox" id="myCheck" name="remember" required>
    <label class="form-check-label" for="myCheck">I agree on blabla.</label>
    <div class="valid-feedback">Valid.</div>
    <div class="invalid-feedback">Check this checkbox to continue.</div>
  </div>
  <button type="submit" class="btn btn-primary">Submit</button>
</form>
```

- <https://www.w3schools.com/>
- <https://developer.mozilla.org/fr/>

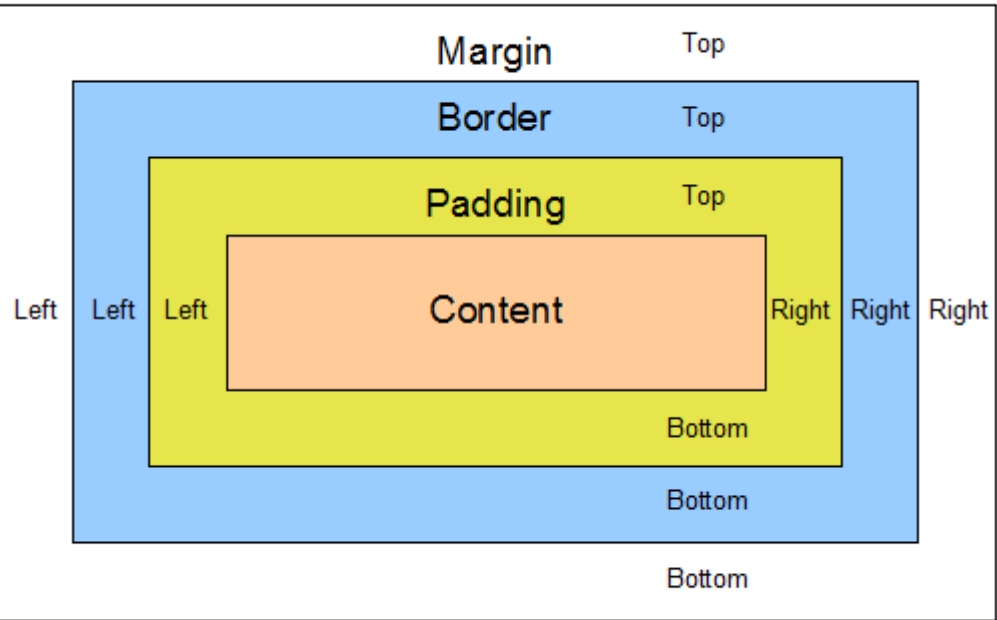
Model

View : HTML/CSS, JS

Controller : serveur web (PHP, nodeJS, etc)

- traite les interactions + sélectionne la vue
- applique les changements sur le modèle + récupère le modèle

avaJava.com Web Tutorials - Cascading Style Sheets



How are margins, borders, padding, and content related?

# Squelette de la page HTML / CSS

## Devtools (F12)

56

Dimensions: Réactivité ▾ 321 x 285 100% ▾ Aucune limitation ▾

Home About Services Contact

## Welcome to Our Website

This is the main content area of your website. You can add various sections and articles here.

© 2024 Your Website. All rights reserved.

Éléments Sources Enregistreur Informations sur les performances Console >> ⚙️ ⋮ ✕

<head> ⋮ </head>

<body> == \$0

> <header> ⋮ </header>

> <main> ⋮ </main>

> <footer> ⋮ </footer>

</body>

</html>

html body

Styles Calculés Mise en page Écouteurs d'événements >>

Filtrer :hov .cls + 🖨️ 📏

element.style { }

body { styles.css:8

font-family: Arial, sans-serif;

line-height: 1.6;

Console Nouveautés ✕ Problèmes Conditions du réseau ✕



```

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Template HTML/CSS</title>
  <link rel="stylesheet" href="styles.css">
</head>
<body>
  <header>
    <nav>
      <ul>
        <li><a href="#">Home</a></li>
        <li><a href="#">About</a></li>
        <li><a href="#">Services</a></li>
        <li><a href="#">Contact</a></li>
      </ul>
    </nav>
  </header>

  <main>
    <section>
      <h1>Welcome to Our Website</h1>
      <p>This is the main content area of your website. You can add various
sections and articles here.</p>
    </section>
  </main>

  <footer>
    <p>&copy; 2024 Your Website. All rights reserved.</p>
  </footer>
</body>
</html>

```

## HTML : index.html

```

/* Reset some default browser styles
*/
body, h1, h2, h3, p, ul {
  margin: 0;
  padding: 0;
}

/* Basic styling */
body {
  font-family: Arial, sans-serif;
  line-height: 1.6;
}

header {
  background-color: #333;
  color: #fff;
  padding: 10px 0;
}

nav ul {
  list-style: none;
  text-align: center;
}

nav ul li {
  display: inline;
  margin-right: 20px;
}

nav ul li a {
  color: #fff;
  text-decoration: none;
}

main {
  padding: 20px;
}

```

## CSS : styles.css

```

footer {
  background-color: #333;
  color: #fff;
  text-align: center;
  padding: 10px 0;
}

```

## Website Header

[Home](#) [About](#) [Services](#) [Contact](#)

### Contact Us

Name:

Email:

Message:

Send

```
<form action="#" method="post">
  <label for="name">Name:</label>
  <input type="text" id="name" name="name" required>

  <label for="email">Email:</label>
  <input type="email" id="email" name="email" required>

  <label for="message">Message:</label>
  <textarea id="message" name="message" rows="4"
required>
  </textarea>

  <button type="submit">Send</button>
</form>
```

```
form {
  max-width: 400px;
  margin: 0 auto;
}
form label {
  display: block;
  margin-bottom: 5px;
}
form input[type="text"],
form input[type="email"],
form textarea {
  width: 100%;
  padding: 8px;
  margin-bottom: 10px;
  border: 1px solid #ccc;
  border-radius: 4px;
}
form textarea {
  resize: vertical;
}
form button {
  display: block;
  width: 100%;
  padding: 10px;
  background-color: #333;
  color: #fff;
  border: none;
  border-radius: 4px;
  cursor: pointer;
  transition: background-color 0.3s ease;
}
form button:hover {
  background-color: #444;
}
```

## Sign Up

Email:

Pass

Confirm Password:



Veuillez inclure "@" dans l'adresse e-mail. Il manque un symbole "@" dans "ahmed.lounis".

Sign Up

```

<form id="signup-form" action="#" method="post">
  <div class="form-group">
    <label for="email">Email:</label>
    <input type="email" id="email" name="email" required>
    <span class="error" id="email-error"></span>
  </div>

  <div class="form-group">
    <label for="password">Password:</label>
    <input type="password" id="password" name="password" required>
    <span class="error" id="password-error"></span>
  </div>

  <div class="form-group">
    <label for="confirm-password">Confirm Password:</label>
    <input type="password" id="confirm-password" name="confirm-password"
required>
    <span class="error" id="confirm-password-error"></span>
  </div>

  <button type="submit">Sign Up</button>
</form>
<script src="script.js"></script>

```

HTML

```

...
input[type="email"],
input[type="password"] {
  width: 100%;
  padding: 8px;
  margin-bottom: 10px;
  border: 1px solid #ccc;
  border-radius: 4px;
}

.error {
  color: red;
  font-size: 12px;
}
...

```

CSS

```
document.addEventListener('DOMContentLoaded', function() {
  const form = document.getElementById('signup-form');
  const email = document.getElementById('email');
  const password = document.getElementById('password');
  const confirmPassword = document.getElementById('confirm-
password');
  const emailError = document.getElementById('email-error');
  const passwordError = document.getElementById('password-error');
  const confirmPasswordError = document.getElementById('confirm-
password-error');
```

```
form.addEventListener('submit', function(event) {
  let isValid = true;
```

```
  // Email validation
```

```
  if (isValidEmail(email.value)) {
    emailError.textContent = 'Invalid email format';
    isValid = false;
  } else {
    emailError.textContent = "";
  }
```

```
  // Password validation
```

```
  if (password.value.length < 6) {
    passwordError.textContent = 'Password must be at least 6
characters long';
    isValid = false;
  } else {
    passwordError.textContent = "";
  }
```

62

### Javascript : script.js

```
// Confirm password validation
```

```
  if (confirmPassword.value !== password.value) {
    confirmPasswordError.textContent = 'Passwords do not match';
    isValid = false;
  } else {
    confirmPasswordError.textContent = "";
  }
```

```
  if (!isValid) {
    event.preventDefault(); // Prevent form submission if validation fails
  }
});
```

```
function isValidEmail(email) {
  const re = /\S+@\S+\.\S+/;
  return re.test(email);
}
});
```

Filter:

Product Name	Category	Price	Stock
Product 1	Category A	\$10	20
Product 2	Category B	\$20	15
Product 3	Category A	\$15	25

```

<div class="filter">
  <label for="filter">Filter:</label>
  <input type="text" id="filter" placeholder="Search...">
</div>

<table id="product-table">
  <thead>
    <tr>
      <th>Product Name</th>
      <th>Category</th>
      <th>Price</th>
      <th>Stock</th>
    </tr>
  </thead>
  <tbody>
    <!-- Product rows will be inserted here dynamically -->
  </tbody>
</table>

<div class="pagination">
  <!-- Pagination links will be inserted here dynamically -->
</div>

```

Html

```

...
.filter {
  margin-bottom: 20px;
}

#product-table {
  width: 100%;
  border-collapse: collapse;
  margin-bottom: 20px;
}

#product-table th, #product-table td {
  border: 1px solid #ccc;
  padding: 8px;
  text-align: left;
}

.pagination {
  text-align: center;
}
...

```

CSS



```

// Sample data for demonstration
const products = [
  { name: 'Product 1', category: 'Category A', price: '$10', stock:
20 },
  { name: 'Product 2', category: 'Category B', price: '$20', stock:
15 },
  { name: 'Product 3', category: 'Category A', price: '$15', stock:
25 },
  // Add more sample data as needed
];

document.addEventListener('DOMContentLoaded', function() {
  const tableBody = document.querySelector('#product-table
tbody');
  const filterInput = document.getElementById('filter');

  // Function to render products in table
  function renderProducts(products) {
    tableBody.innerHTML = '';
    products.forEach(product => {
      const row = document.createElement('tr');
      row.innerHTML = `
        <td>${product.name}</td>
        <td>${product.category}</td>
        <td>${product.price}</td>
        <td>${product.stock}</td>
      `;
      tableBody.appendChild(row);
    });
  }
65 }

```

JavaScript

```

// Function to filter products based on search input
function filterProducts(keyword) {
  const filteredProducts = products.filter(product =>
    product.name.toLowerCase().includes(keyword.toLowerCase())
    ||
    product.category.toLowerCase().includes(keyword.toLowerCase())
  );
  renderProducts(filteredProducts);
}

// Initial render
renderProducts(products);

// Event listener for filtering
filterInput.addEventListener('input', function() {
  filterProducts(this.value.trim());
});

```

ahmed.lounis@hds.utc.fr