



HTML-CSS-JS



Semifir

contact@semifir.com
13 Avenue du Président John F. Kennedy,
59000 Lille.

Sommaire

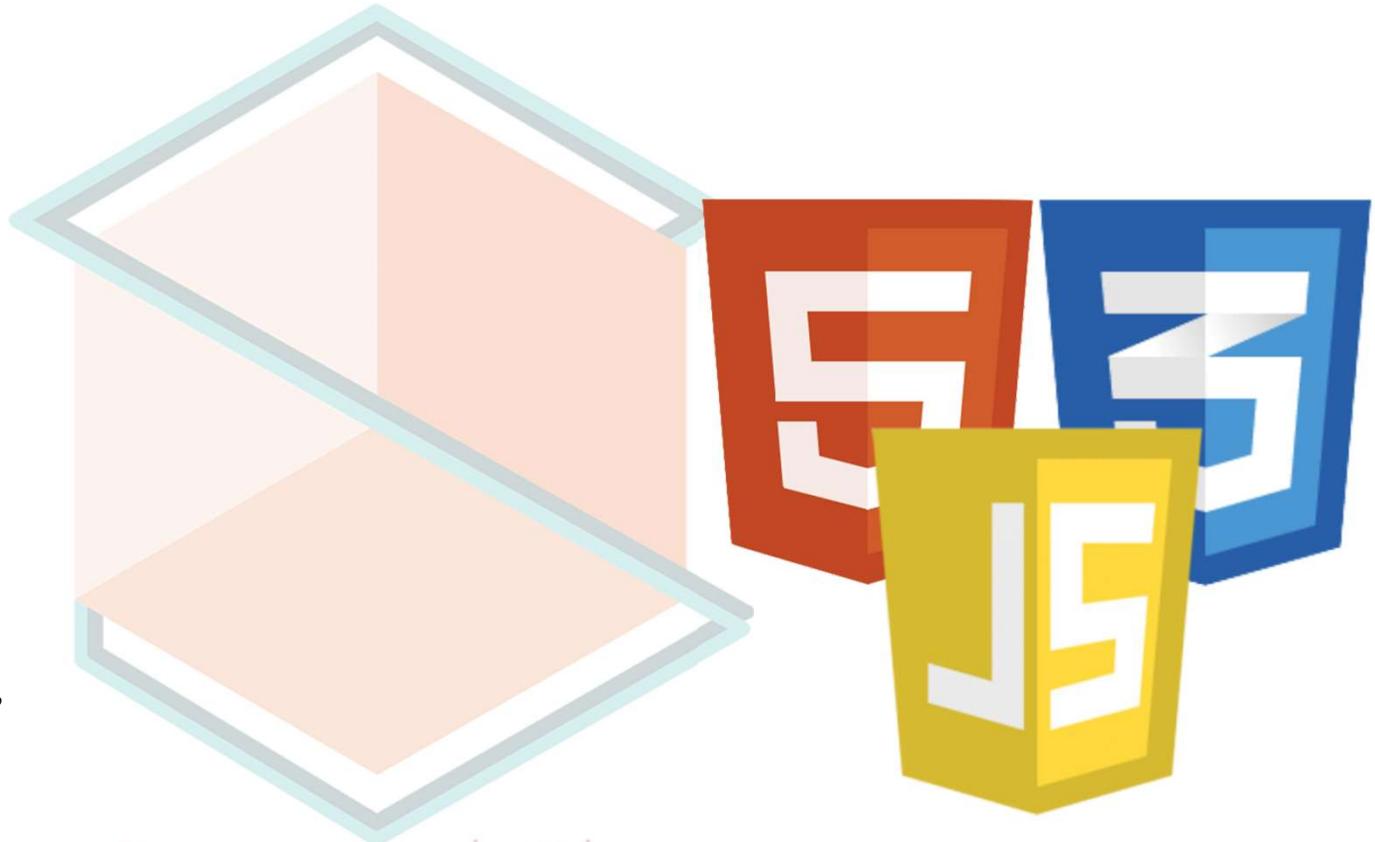
I. HTTP

II. HTML 5

- Les balises de structure,
- Les balises de texte,
- Les attributs,
- Les images,
- Les formulaires,
- Les bonnes pratiques.

III. CSS 3

- Sélecteurs,
- Boites et positionnement,
- Media Queries,
- Mise en forme du texte,



I. HTTP

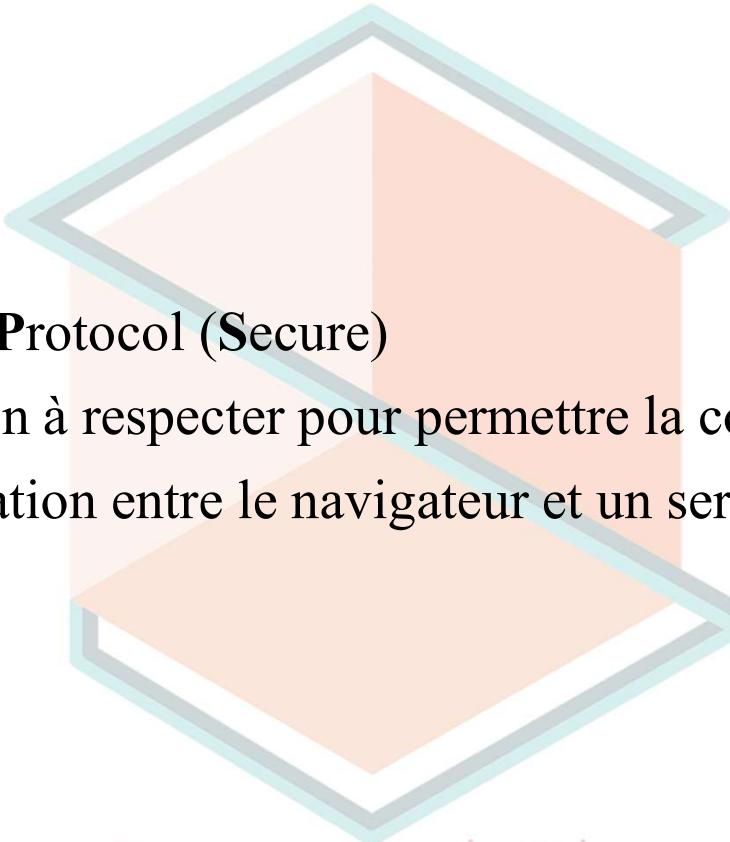


HTML-CSS-JS

HTTP(S)

Présentation :

- HyperText Transfer Protocol (**Secure**)
- Protocole : convention à respecter pour permettre la communication
- Permet la communication entre le navigateur et un serveur web



Semifir

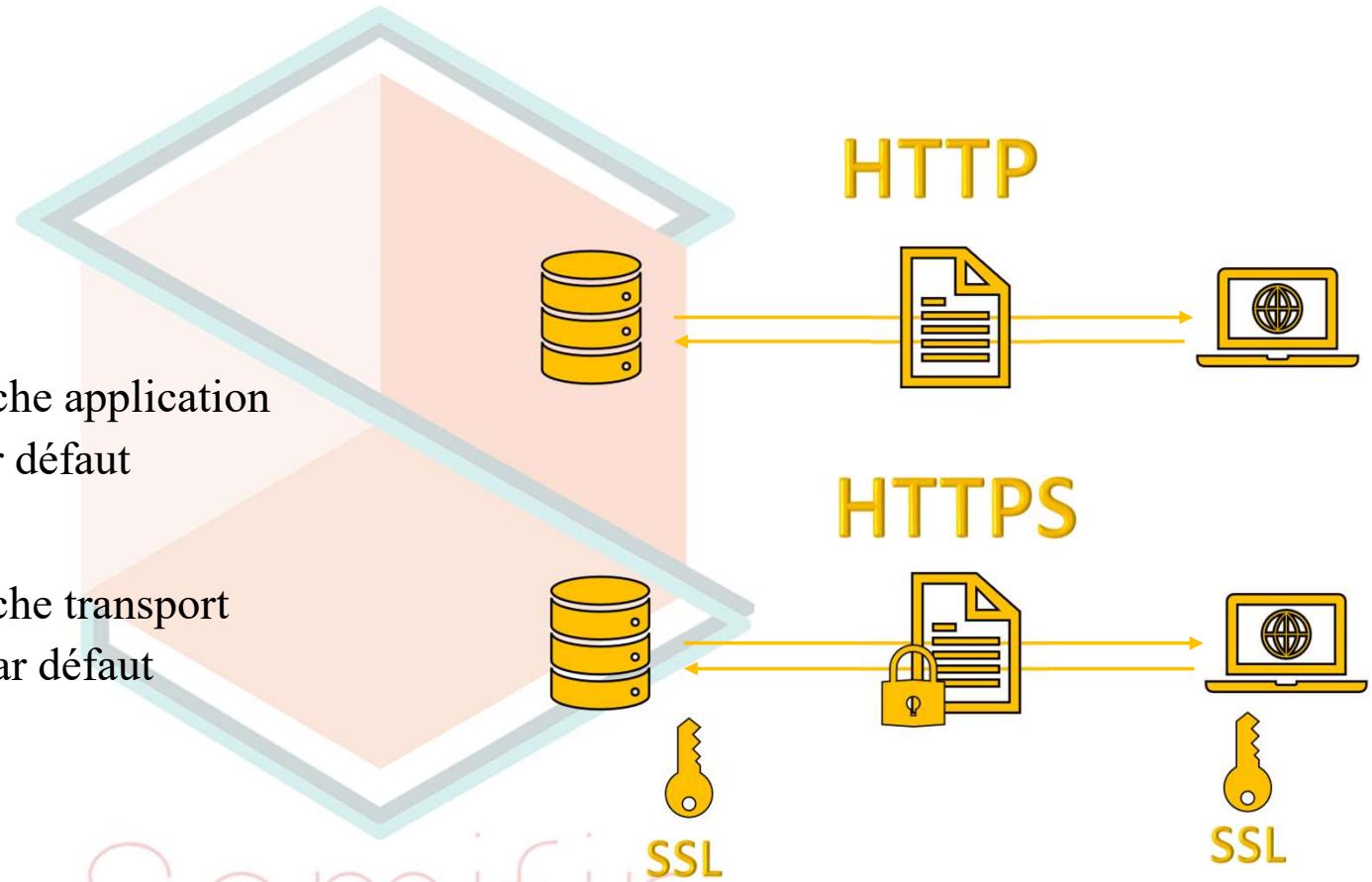
***Le protocole HTTP(S) est indispensable au fonctionnement
d'internet !***

HTML-CSS-JS

HTTP(S)

Présentation :

- HTTP :
 - Fait partie de la couche application
 - Utilise le port 80 par défaut
- HTTPS :
 - Fait partie de la couche transport
 - Utilise le port 443 par défaut



Depuis quelques années, tous les sites ou presque sont en HTTPS

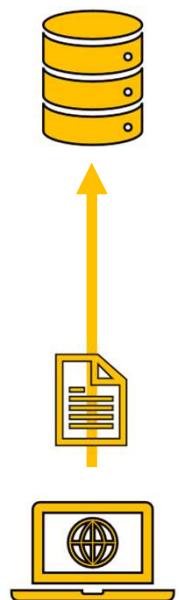
HTML-CSS-JS

HTTP(S)

La requête :

Il existe plusieurs types de requêtes HTTP, appelées « méthodes » :

Méthode	Description
Get	Demande une ressource au serveur (fichier, page ...)
Post	Envoyer des données au serveur (formulaire...)
Put	Créer ou modifier une ressource existante
Delete	Supprimer une ressource
Head	Demande d'informations sur une ressource



Semifir

Une requête est une demande faite au serveur

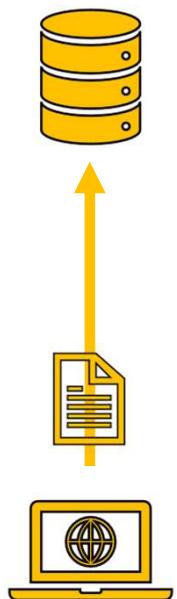
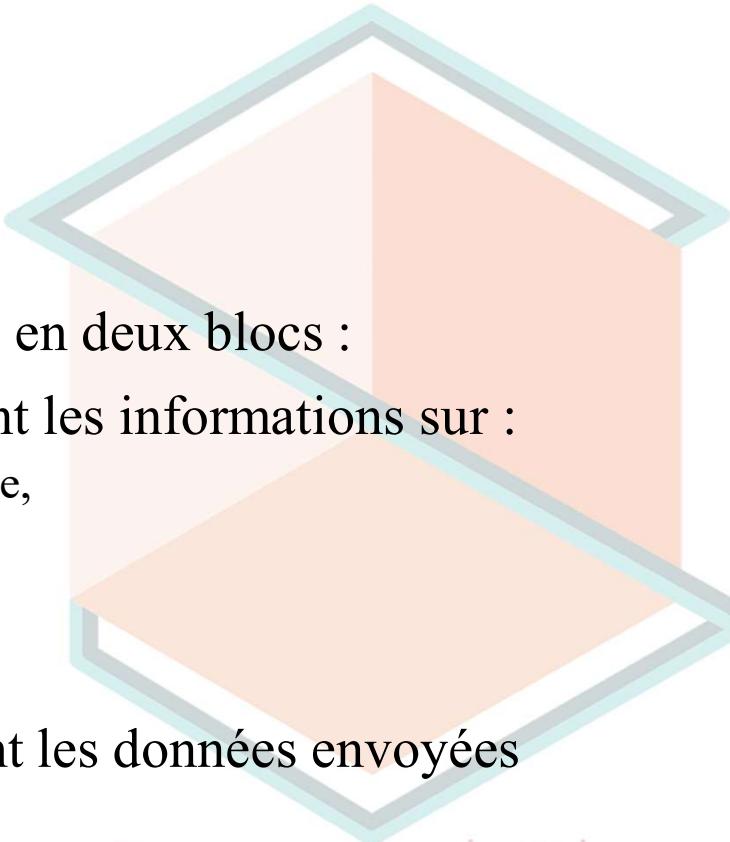
HTML-CSS-JS

HTTP(S)

La requête :

Une requête est divisée en deux blocs :

- **L'en-tête**, qui contient les informations sur :
 - La méthode employée,
 - Le serveur,
 - Le client
- **Le corps**, qui contient les données envoyées



Semifir

HTML-CSS-JS

HTTP(S)

Exemple avec GET :

En-tête :

```
GET /chemin/hello.htm HTTP/1.1
User-Agent: Mozilla/4.0 (compatible; MSIE5.01; Windows NT)
Host: www.site.com
```

Semifir

HTML-CSS-JS

HTTP(S)

Exemple avec POST :

En-tête :

```
POST /chemin/hello.htm HTTP/1.1
User-Agent: Mozilla/4.0 (compatible; MSIE5.01; Windows NT)
Host: www.site.com
```

Corps :

```
{
  Login: Adrien,
  Password: 1234
}
```

HTML-CSS-JS

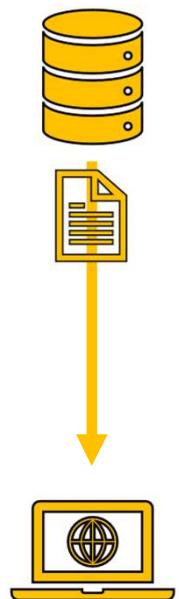
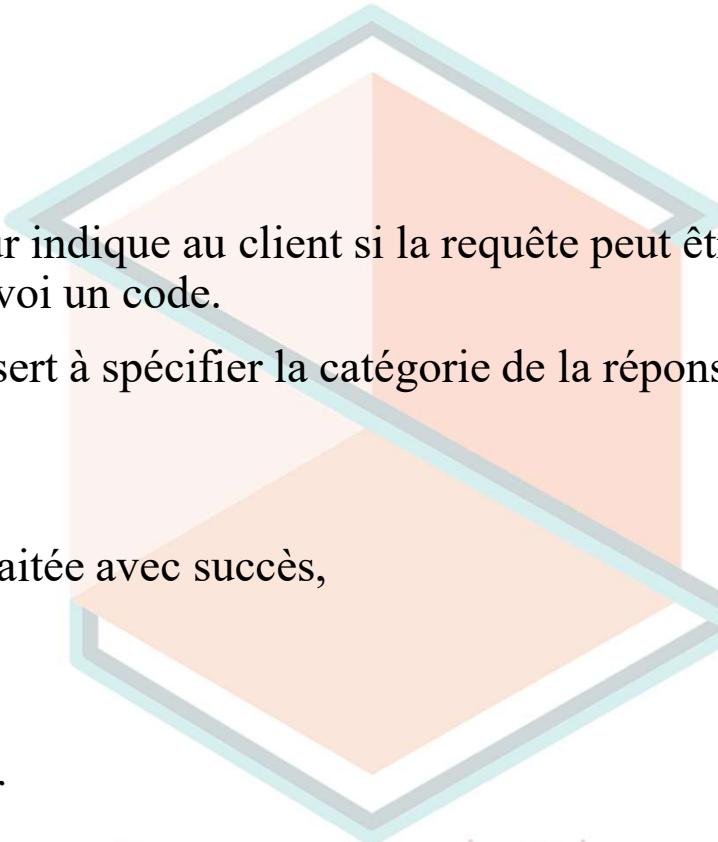
HTTP(S)

La réponse :

Avant toute chose, le serveur indique au client si la requête peut être satisfaite ou non. Pour ce faire, il lui renvoi un code.

Le premier chiffre du code sert à spécifier la catégorie de la réponse, parmi plusieurs possibilités :

- **1xx** : Informations,
- **2xx** : La demande a été traitée avec succès,
- **3xx** : Redirections
- **4xx** : Erreurs coté client
- **5xx** : Erreurs coté serveur

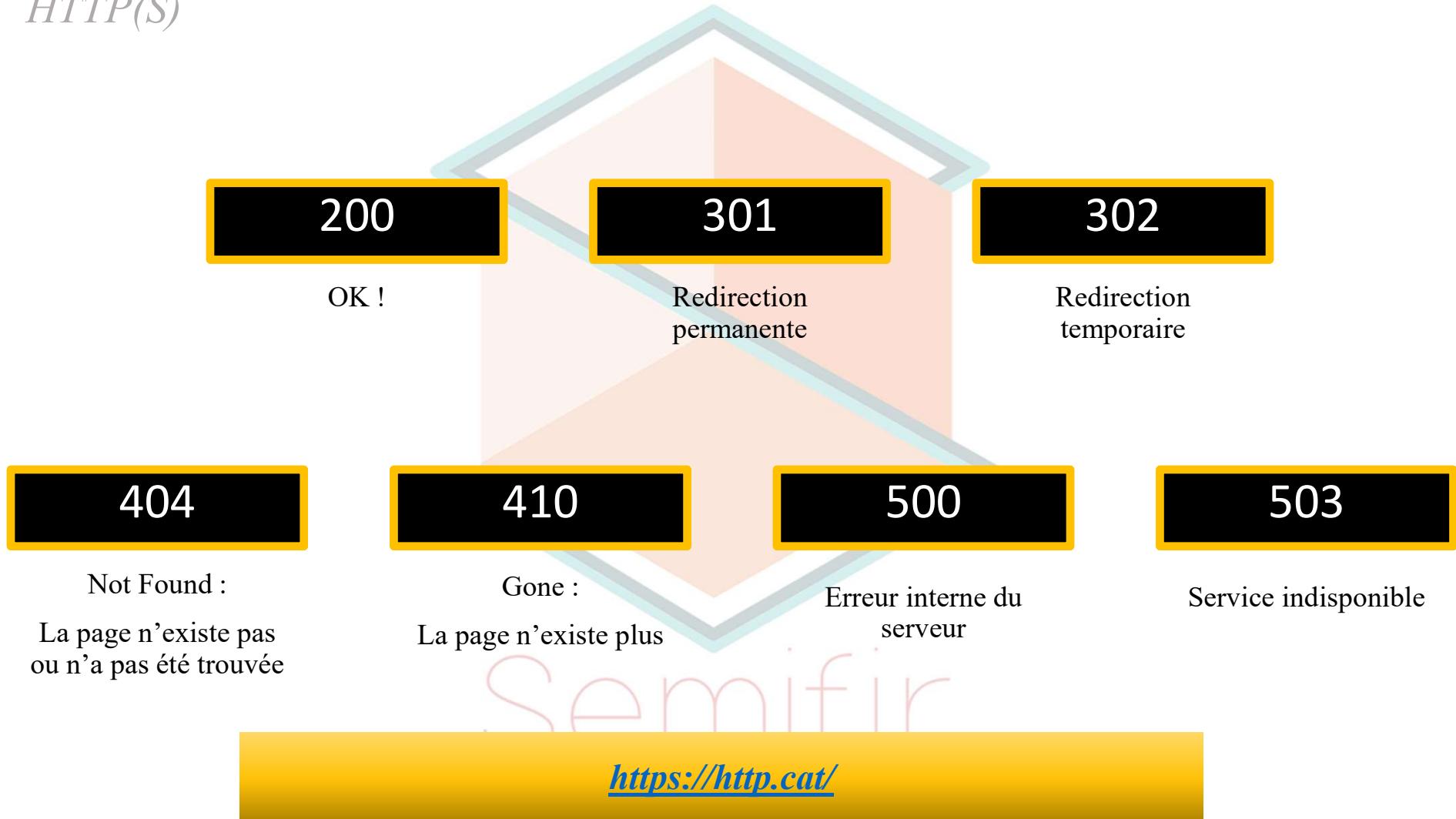


Semifir

Vous retrouverez ici les « status code » HTTP : <https://http.cat/>

HTML-CSS-JS

HTTP(S)



HTML-CSS-JS

HTTP(S)

La réponse émet elle aussi une entête et un corps:

En-tête :

```
HTTP/1.1 200 OK
Date: Fri, 24 December 2021 16:04:09 GMT
Server: Apache/2.4.6 (CentOS) PHP/5.4.16
```

Corps :

```
<!DOCTYPE html>
<head>
<meta charset=" utf-8 >>
</head>
<body>
```

Préparez votre environnement :

Quelques extensions utiles pour VS Code :

- Auto Close Tag
- Auto Rename Tag
- HTML Preview
- HTML Snippets

II. HTML 5

Balises de structure

HTML



HTML-CSS-JS

HTML

HTML

- HyperText Markup Languages
- Comme tous les langage, il a sa propre grammaire, vocabulaire ...
- Le HTML n'est que texte !
- Le HTML n'est pas un langage de programmation :

C'est un langage de balisage

Semifir

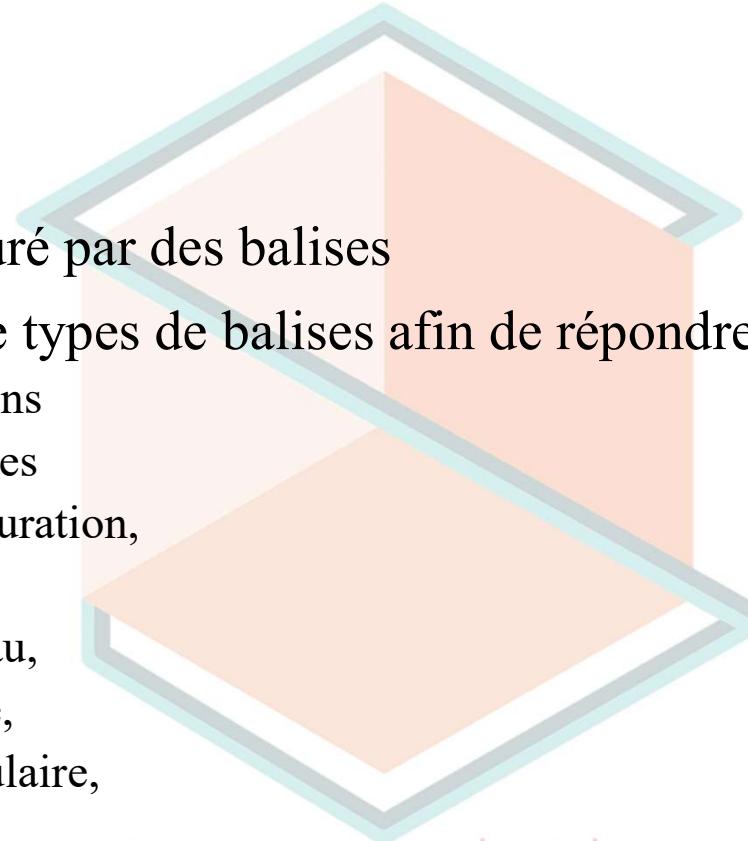
Le HTML descend du SGML, inventé dans les années 1990

HTML-CSS-JS

Balises

Balises

- Le HTML est structuré par des balises
- Il existe beaucoup de types de balises afin de répondre à des besoins précis :
 - Les balises de sections
 - Les balises génériques
 - Les balises de structuration,
 - Les balises de liste,
 - Les balises de tableau,
 - Les balises d'en-tête,
 - Les balises de formulaire,



Semifir

HTML-CSS-JS

Balises

Une balise commence et se termine comme suit

```
<div> Contenu </div>
```

Certaines balises sont « auto-fermant » :

```
<input type="text">
```

Avec les bonnes extensions, VS Code est un allié de choix pour ne pas oublier de refermer vos balises !

HTML-CSS-JS

Balises

Une balise peut contenir d'autres balises :

```
<section>
  <article>
    <p></p>
  </article>
</section>
```

Semifir

HTML-CSS-JS

Balises

Les balises de premier niveau permettent de structurer la page. Elles représentent le **minimum** à insérer

```
<!DOCTYPE html>
<!-- "html" indique le début et la fin du fichier html -->
<html>

    <!-- "head" contient des données relatives au paramétrage, qui
        ne seront pas affichées-->
    <head>
        <meta charset="utf-8" />
        <title>Document</title>
    </head>

    <!-- "body" contient les éléments visibles de notre site -->
    <body>

    </body>
</html>
```

Tapez « `html:5` » dans VS Code pour générer cette partie automatiquement !

HTML-CSS-JS

HTML – Best Practices

<head> peut contenir beaucoup d'informations concernant notre site : emplacement des feuilles de style, des scripts icone des favoris etc.

```
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="keywords" content="demo, html5, html, CSS">
    <meta name="author" content="Alexandre Devos">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Titre du document</title>
    <link rel="stylesheet" type="text/css" href="chemin/vers/css">
    <link rel="shortcut icon" href="chemin/vers/icone" type="image/x-icon">
</head>
```

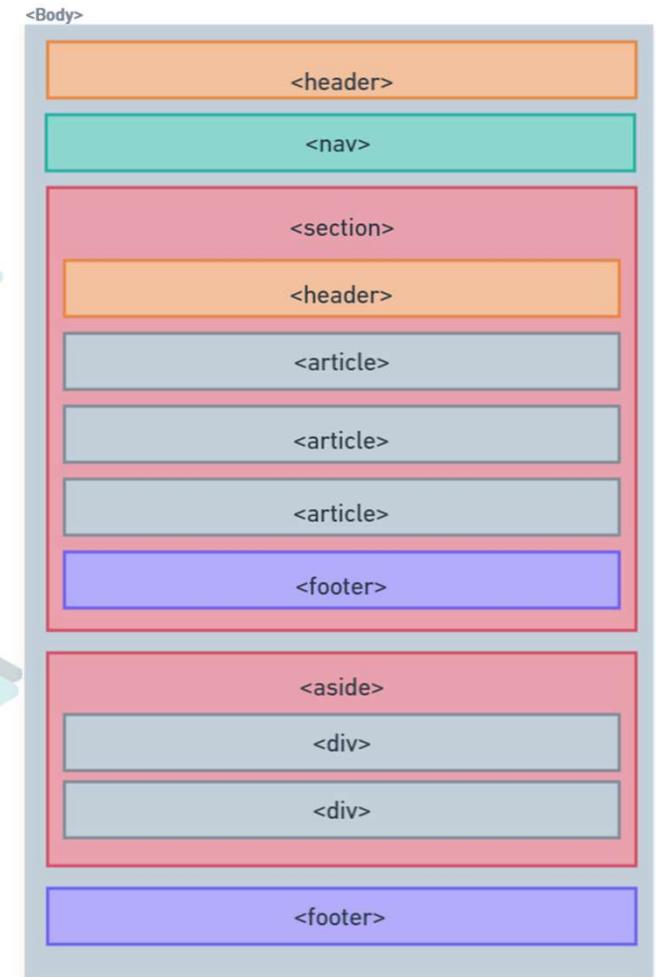
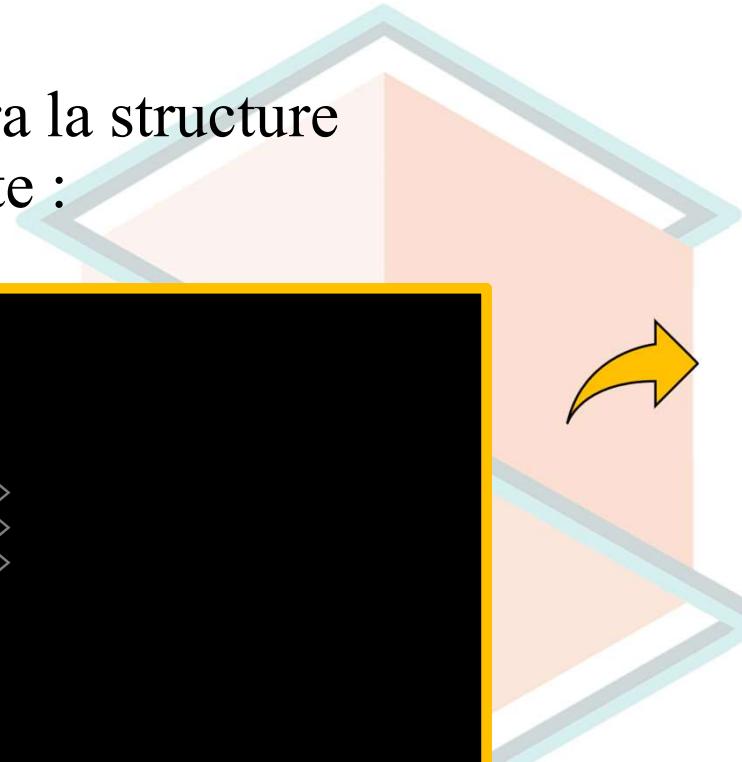
Les données présentes dans HEAD ne sont pas directement visibles sur notre site !

HTML-CSS-JS

Balises

<body> contiendra la structure visible de notre site :

```
<body>
  <header></header>
  <nav></nav>
  <section>
    <header></header>
    <article></article>
    <article></article>
    <article></article>
    <footer></footer>
  </section>
  <aside>
    <div></div>
    <div></div>
  </aside>
  <footer></footer>
</body>
```



Exemple de structure dans le body

HTML-CSS-JS

Balises

Les balises de sections sont contenues dans le body permettent de construire le squelette du site :

```
<!-- En-tête de la page -->
<header></header>

<!-- Liens de navigation -->
<nav></nav>

<!-- Section de page -->
<section></section>

<!-- Article (contenu autonome) -->
<article></article>

<!-- Pied (bas) de page-->
<footer></footer>
```

C'est à l'intérieur de ces balises que l'on insérera notre contenu

HTML-CSS-JS

Balises

Les balises de sections sont contenues dans le body permettent de construire le squelette du site :

En-tête de la page

```
<header></header>
```

Liens de navigation

```
<nav></nav>
```

Sections :

```
<section></section>
```

Articles (contenu autonome)

```
<article></article>
```

Pied de page :

```
<footer></footer>
```

HTML-CSS-JS

Balises

Balises génériques :

Balise de division :

Elle est de type « Bloc » (prend donc toute la largeur de l'écran)

```
<div></div>
```

Balise span :

Elle est de type « Inline »

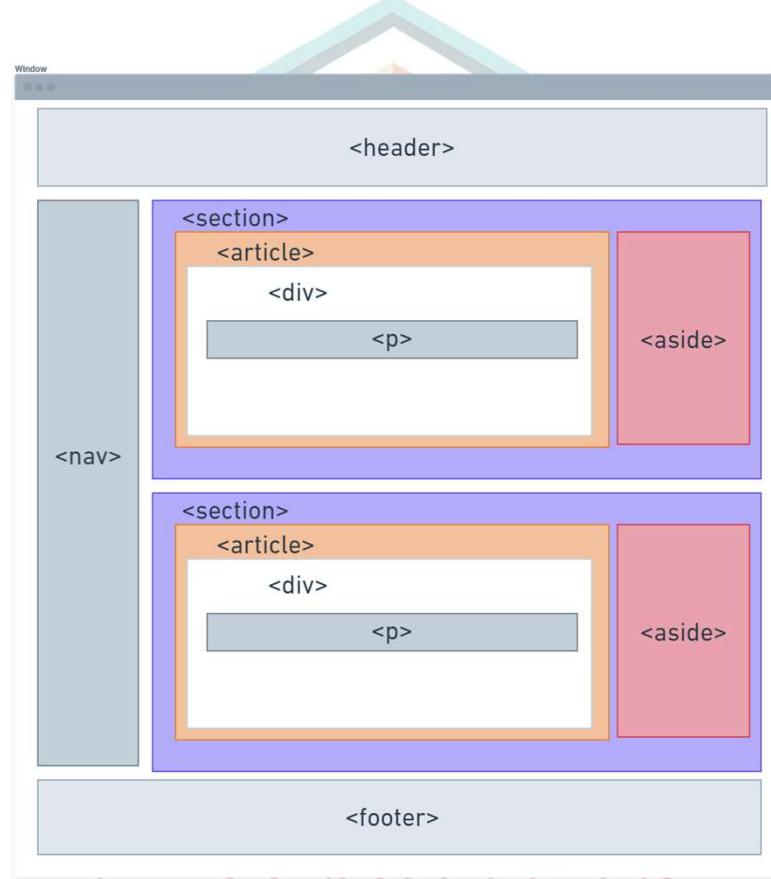
```
<span></span>
```

Les balises génériques ont pour but d'être des conteneurs : on s'en sert pour ranger notre contenu et faciliter leur mise en forme

Semifir

HTML-CSS-JS

Balises



Exemple de structure de site

II. HTML 5

Balises de texte



HTML

5

HTML-CSS-JS

Balises de texte

Les titres :

```
<!-- Balises de titre -->
<h1>Ceci est un titre principal</h1>
<h2>Ceci est un titre</h2>
<h3>Ceci est un sous-titre</h3>
<h4>Ceci est un sous sous-titre</h4>
```

Ceci est un titre principal
Ceci est un titre
Ceci est un sous-titre
Ceci est un sous sous-titre

Il est conseillé de n'utiliser qu'une seule balise de type « h1 » pour un bon référencement

HTML-CSS-JS

Balises de texte

Balise de paragraphe

```
<p>Du texte</p>
```

Balise d'insertion d'image

```
<img src= "path" alt= "remplacement txt">
```

Balise de citation (quote)

```
<q>Un grand pouvoir implique de grandes responsabilités.</q>
```

Structuration du texte :

Balise de retour à la ligne

```
<br />
```

Balise de citation de titre

```
<cite>Le seigneur des anneaux</cite>
```

Balise d'ancre :

```
<a href="http://lien.com"> Lien </a>
```

HTML-CSS-JS

Balises de texte

```
<article>
  <header>
    <h1>Ceci est un titre principal dans un header</h1>
  </header>
  <div>
    <!-- Balises de titre -->
    <h2>Ceci est un titre</h2>
    <p>
      Lorem ipsum dolor, sit amet consectetur adipisicing elit. Pariatur,
      totam? Placeat, reiciendis.
      <br />
      <strong>un retour à la ligne</strong>
      <br />
      Illum delectus quasi voluptatibus hic saepe quis fuga eum temporibus
      voluptatum suscipit corrupti itaque aliquid dolor? Dicta, illo.
      <br />
      <cite>Une citation</cite>
    </p>
    <h3>Ceci est un sous titre</h3>
    <p>
      Lorem ipsum dolor sit amet consectetur, adipisicing elit. Animi odio
      quisquam molestias similique! Molestiae minima tempore ratione
      repudiandae minus modi facilis, commodi quia illum vitae, alias hic
      delectus quos enim!
    </p>
    <h2>Ceci est un autre titre</h2>
    <h3>Ceci est un sous titre</h3>
    <p>
      Lorem ipsum dolor sit amet consectetur, adipisicing elit. Animi odio
      quisquam molestias similique! Molestiae minima tempore ratione
      repudiandae minus modi facilis, commodi quia illum vitae, alias hic
      delectus quos enim!
    </p>
  </div>
</article>
```



Ceci est un titre principal dans un header

Ceci est un titre

Lorem ipsum dolor, sit amet consectetur adipisicing elit. Pariatur, totam? Placeat, reiciendis.

un retour à la ligne

Illum delectus quasi voluptatibus hic saepe quis fuga eum temporibus voluptatum suscipit corrupti itaque aliquid dolor? Dicta, illo.

Une citation

Ceci est un sous titre

Lorem ipsum dolor sit amet consectetur, adipisicing elit. Animi odio quisquam molestias similique! Molestiae minima tempore ratione repudiandae minus modi facilis, commodi quia illum vitae, alias hic delectus quos enim!

Ceci est un autre titre

Ceci est un sous titre

Lorem ipsum dolor sit amet consectetur, adipisicing elit. Animi odio quisquam molestias similique! Molestiae minima tempore ratione repudiandae minus modi facilis, commodi quia illum vitae, alias hic delectus quos enim!

Exemple de structure de texte

HTML-CSS-JS

Balises de texte

Structuration du texte :

Balise « important »:

```
<strong>Gras</strong>
```

Balise « Italique » :

```
<em>Italique</em>
```

Balise « Mettre en valeur »:

```
<mark>Surligné</mark>
```

NB : Si le HTML permet la mise en gras du texte, ce n'est pas son rôle !
Péferez l'utilisation du CSS pour mettre en valeur des éléments

HTML-CSS-JS

Balises de texte

Les Listes ordonnées et non ordonnées :

```
<!-- Liste non ordonnée : -->
<article>
  <div>
    <h2> Liste de courses </h2>
    <ul>
      <li>Beurre</li>
      <li>Pain</li>
      <li>Oeufs</li>
    </ul>
  </div>
</article>
```

```
<!-- Liste ordonnée : -->
<article>
  <div>
    <h2> Trucs à faire </h2>
    <ol>
      <li>Faire fondre le beurre dans la poele</li>
      <li>Ajouter un oeuf</li>
      <li>Couper une tranche de pain</li>
      <li>Mettre l'oeuf sur la tartine</li>
    </ol>
  </div>
</article>
```

ligne

Liste de courses

- Beurre
- Pain
- Oeufs

Trucs à faire

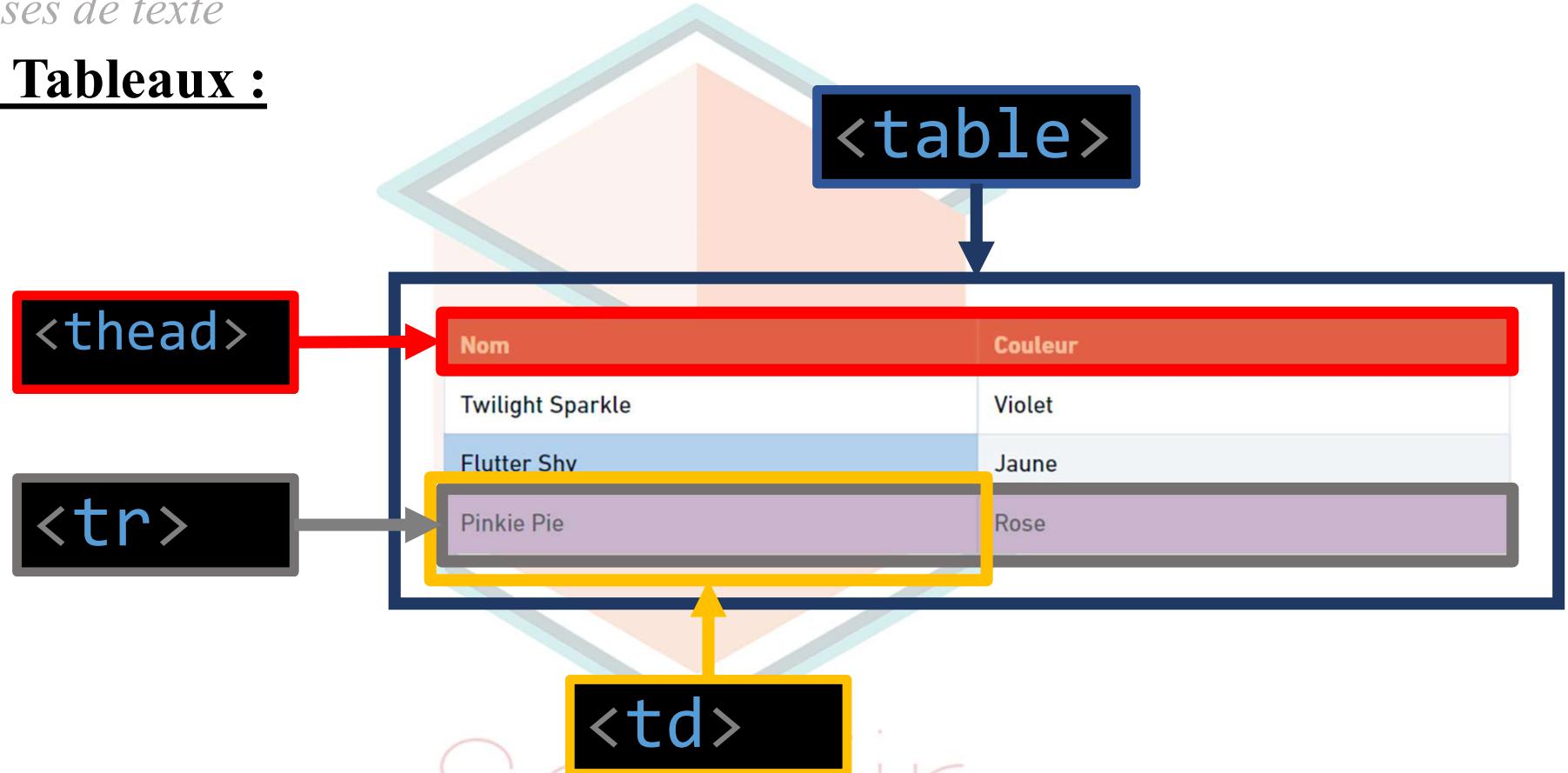
1. Faire fondre le beurre dans la poele
2. Ajouter un oeuf
3. Couper une tranche de pain
4. Mettre l'oeuf sur la tartine

NB : Il est possible de mettre une liste à l'intérieur d'une liste

HTML-CSS-JS

Balises de texte

Les Tableaux :



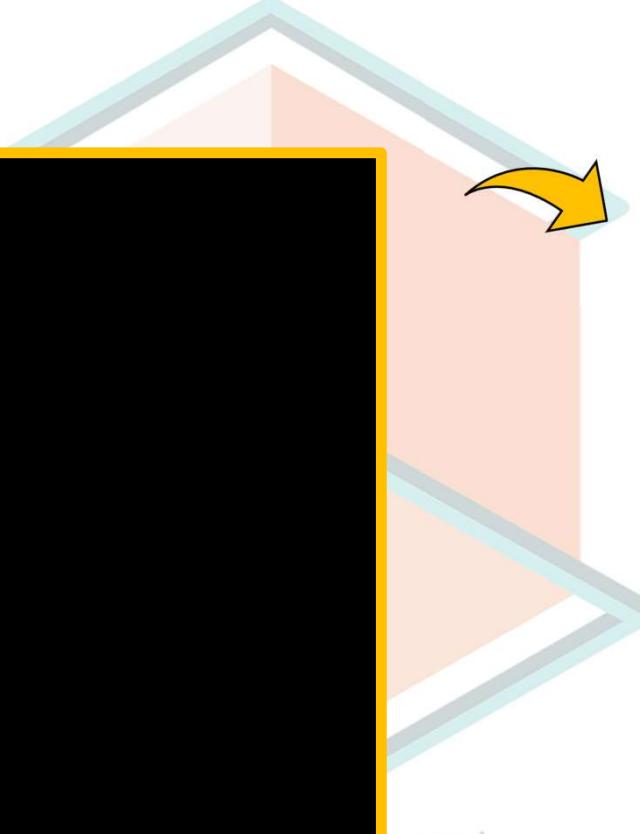
NB : Un tableau ne doit pas être utilisé pour mettre en forme le site !

HTML-CSS-JS

Balises de texte

Les Tableaux :

```
<table>
  <caption>Poneys</caption>
  <!-- En-tête -->
  <thead>
    <tr>
      <th>Nom</th>
      <th>Couleur</th>
    </tr>
  </thead>
  <!-- Pied -->
  <tfoot>
    <tr>
      <td>Twilight Sparkle</td>
      <td>Violet</td>
    </tr>
    <tr>
      <td>Fluttershy</td>
      <td>Jaune</td>
    </tr>
    <tr>
      <td>Pinkie Pie</td>
      <td>Rose</td>
    </tr>
  </tfoot>
</table>
```



A large yellow arrow points from the left towards a central area containing a house-shaped diagram. This diagram has a light blue roof and a pink body. A smaller yellow arrow points from the bottom right corner of the house towards the right side of the slide, where the table is displayed.

Poneys	
Nom	Couleur
Twilight Sparkle	Violet
Fluttershy	Jaune
Pinkie Pie	Rose

NB : Un tableau ne doit pas être utilisé pour mettre en forme le site !

II. HTML 5

Les attributs



HTML-CSS-JS

Les attributs

Les attributs

- Complètent une balise
- Peut être obligatoire ou facultatif
- Peut permettre de changer le comportement d'une balise
- Certaines balises disposent de leurs attributs spécifiques

NB : Il est possible de changer le style d'une balise grâce à des attributs, mais c'est une mauvaise pratique : c'est le rôle du CSS !

HTML-CSS-JS

Les attributs

Quelques attributs universels :

class=""

Permet de définir la classe de la balise. Est utilisé avec le CSS ou le JavaScript pour manipuler la balise

id=""

Permet d'affecter un identifiant à une balise. Pratique pour retrouver un élément via le CSS ou le JS.

hidden = true

Permet de cacher un élément, pour l'afficher plus tard une fois une condition remplie (avec le js)

style=""

Permet de faire des déclarations CSS. Préférez l'utilisation d'un fichier CSS plutôt que son utilisation !

Liste des attributs :

<https://developer.mozilla.org/fr/docs/Web/HTML/Attributes>

HTML-CSS-JS

Les attributs



Les attributs sont déclarés dans la balise ouvrante, comme suit :

```
<div class="MonStyle">  
    <!-- Contenu -->  
</div>
```

Ici, c'est un attribut qui permet d'appliquer un style nommé « MonStyle », présent dans le CSS.

HTML-CSS-JS

Les attributs

Balise `<a>` et ses attributs :

```
<a href=""></a>
```

- Permet d'indiquer la cible du lien. Il peut être externe (<https://..>) ou interne (autre emplacement sur la page/le site)
- On peut y indiquer l'ID d'un autre élément de notre page en précisant « # » devant le nom de l'ID : **#IdDeMonElement**
- On peut spécifier un autre fichier (page) du site en précisant son nom. Pensez aux « `./file.html` » et « `/dir/file.html` » dans le cas d'utilisation de plusieurs répertoires.
- `./file.html#id` : pointe directement vers l'élément `#id` d'une page `./file.html`.

<code><a> est une balise qui permet d'insérer des liens</code>

HTML-CSS-JS

Les attributs

Quelques attributs avec <a> :

```
<a href=""></a>
```

- Permet d'ajouter un fichier téléchargeable dans le lien.
- On peut insérer des fichiers téléchargeables en précisant leur nom suivi de l'extension du fichier : « href=« MonFichier.zip »»
- L'attribut **title=""** peut être ajouté afin d'afficher une infobulle au survol
- **target= "_blank"** permet d'ouvrir une nouvelle fenêtre au lieu de rediriger

L'utilisation de _blank est déconseillé par souci de fluidité de navigation : l'utilisateur choisira lui-même quand ouvrir une nouvelle fenêtre.

II. HTML 5

Les images



HTML-CSS-JS

Les images

Le format JPEG :

- Joint Photographic Expert Group
- Connu pour apporter un poids léger aux images
- Peut comporter plus de 16 Millions de couleurs différentes
- Provoque une légère perte de qualité sur les images
- Utilisation recommandée pour les photos, mais pas pour les images (préférez le png)

La compression JPEG est optimisé pour les photographies

HTML-CSS-JS

Les images

Le format PNG :

- Portable Network Graphics
- Format assez récent
- Adapté pour tous les graphiques (= image qui n'est pas une photo)
- Peut être rendu transparent et n'altère pas la qualité de l'image
- Existe en 2 versions :
 - 8 bits (256 couleurs)
 - 24 bits (16 Millions de couleurs, comme un JPEG)

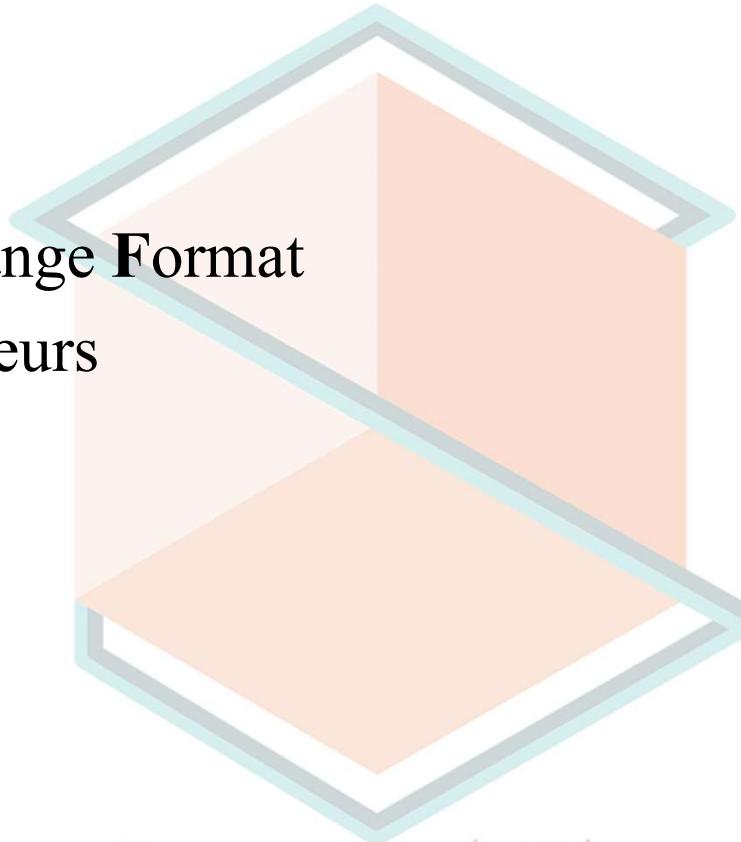
La compression de PNG est optimisé pour les images

HTML-CSS-JS

Les images

Le format GIF :

- Graphics Interchange Format
- Limité à 256 couleurs
- Peut être animé



Semifir

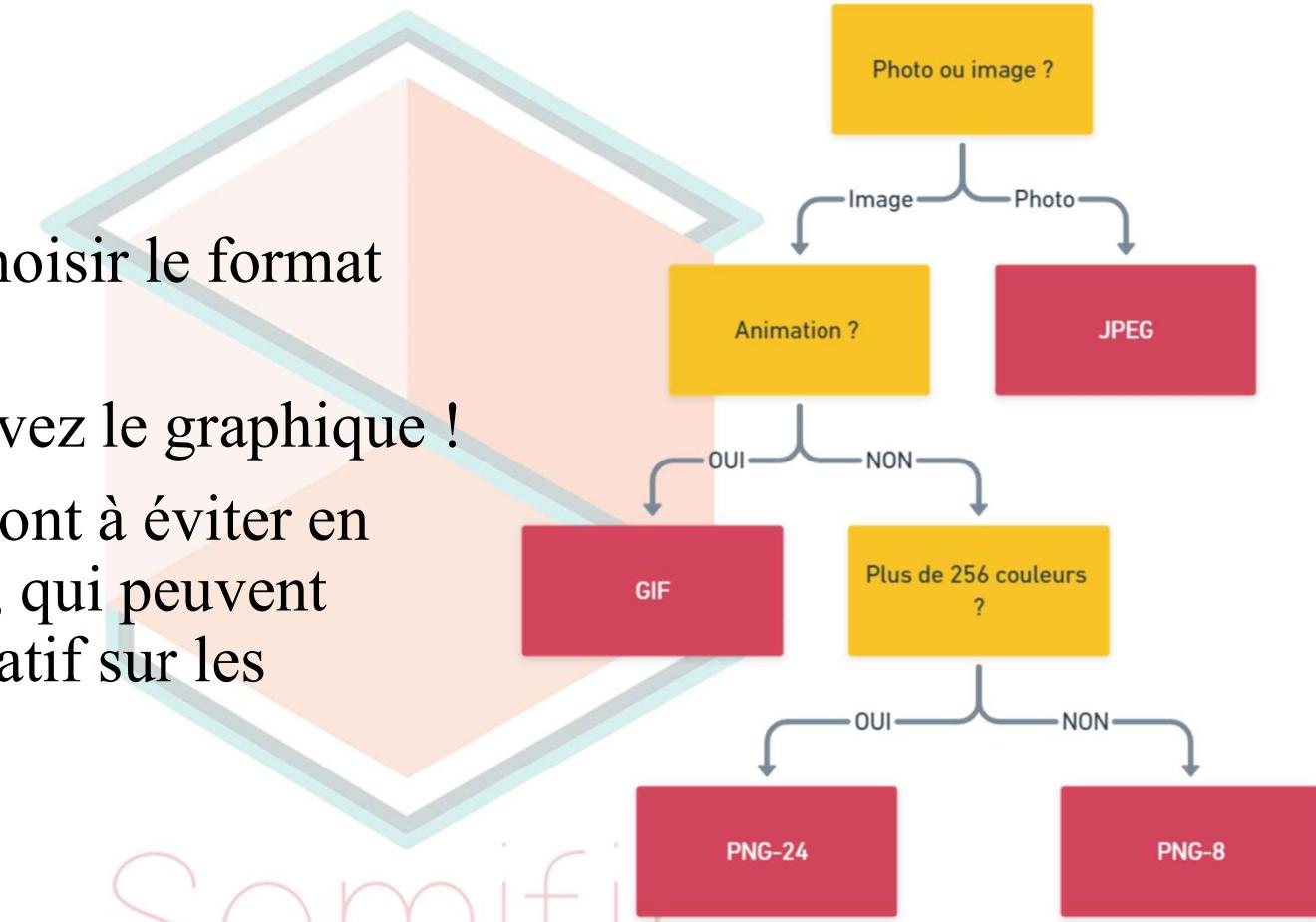
*Steve Wilhite, le créateur du GIF a tranché : ça se prononce
« djif » [dʒif]*

HTML-CSS-JS

Les images

Mémo :

- Toujours veiller à choisir le format adapté !
- En cas de doute, suivez le graphique !
- Les autres formats sont à éviter en raison de leur poids, qui peuvent avoir un impact négatif sur les performances !



Semifinal

Un doute ? Un mémo !

HTML-CSS-JS

Les attributs

Balise d'image et ses attributs :

```
<img src="" alt="">
```

- SRC : pointe la source de l'image dans les fichiers de notre site : « /src/img/image.png ». Est **obligatoire**
- ALT : Texte de remplacement, sert à décrire l'image et est **obligatoire** !
- TITLE : Permet d'ajouter un titre à l'image, qui s'affichera lorsque survolé.

Nota Bene :

- Une image doit forcément se trouver dans un paragraphe ou une balise figure !
- Pensez à ne pas utiliser d'espace ni d'accents dans les noms de vos images et des répertoires !

Si l'image est trop grosse, pensez à insérer une miniature dans une balise <a>, qui redirigera vers le fichier à taille réelle.

II. HTML 5

Les formulaires



HTML-CSS-JS

Les Formulaires

- Les formulaires permettent de demander des informations à l'utilisateur (ex : formulaire d'inscription...)
- L'information pourra ensuite être traitée, notamment par des scripts JavaScript.
- On utilise la balise **<form>**

```
<form>
```

```
</form>
```

Nous insérerons nos différents balises de champs dans la balise form.

HTML-CSS-JS

Les Formulaires

Attributs de la balise <form>:

action=""

Adresse de la page ou du programme qui va traiter les informations du formulaire.

method=""

Permet de définir la méthode qui sera utilisée (GET ou POST). Dans le cas d'un formulaire, on sera sur en général « POST ».

Semifir

HTML-CSS-JS

Les Formulaires

Les zones de saisie <input> : le type texte

```
<input type="text">
```

Permet d'insérer une zone de texte vide que l'utilisateur pourra remplir.

Pour être identifiable, on lui ajoute une balise « label ».

```
<label for="">Label</label>
```

Permet d'afficher un libellé à l'utilisateur.

Coté machine, permet d'identifier le champs

HTML-CSS-JS

Les Formulaires

- ***id*** : correspond à l'élément en tant que tel
- ***name*** : correspond à la variable du formulaire
- ***for*** : est dans le label, il permet de lier la label à l'id de l'input en question (ici « nom »)

```
<form method="POST">
  <label for="nom">Entrez votre nom :</label>
  <input type="text" name="nom" id="nom">
</form>
```

Avoir plusieurs éléments de même « name » permettra l'usage des radio button et des checkbox.

HTML-CSS-JS

Les Formulaires



On peut avoir plusieurs balises ayant le même attribut « name », mais on ne peut pas avoir plusieurs balises ayant le même attribut « id »

Semifir

HTML-CSS-JS

Les Formulaires

Les attributs supplémentaires de <input> :

size=""

Permet de modifier la taille du champs

maxlength=""

Permet d'affecter une longueur maximale.

value=""

Permet de saisir une valeur par défaut.
Est prioritaire sur le placeholder.

placeholder=""

Permet de préremplir la zone de texte
(en général avec un exemple). S'efface
au premier caractère tapé, contrairement
à Value

En plus du de l'attribut « type » bien entendu.

HTML-CSS-JS

Les Formulaires

Les zones de saisie <input> : le type Password

<input type="password">

Permet d'insérer une zone de texte qui n'affichera que des « dots » lors de la saisie.

Pseudo : Alex

Mot de passe :

Il est important de penser à indiquer le bon type !

HTML-CSS-JS

Les Formulaires

Les zones de saisie <input> : le type email

```
<input type="email" name="email" id="email">
```

Permet d'insérer une zone de texte de type « email ». Permet l'auto complétion.



Pseudo :

Mot de passe :

Courriel : de

Décrivez :

devos.alexandre@live.fr

Sur les navigateurs mobiles, le clavier s'adaptera pour afficher l'@ sur la page principale du clavier !

HTML-CSS-JS

Les Formulaires

Les zones de saisie <input> : le type tel

```
<input type="tel" name="tel" id="tel">
```

Comme pour le champ email, permet l'auto complétion et adaptera le clavier d'un smartphone pour afficher directement le pavé numérique.

Pseudo :
Mot de passe :
Courriel
Téléphone

HTML-CSS-JS

Les Formulaires

Les zones de saisie <input> : le type number

```
<input type="number" name="number" id="number">
```

Affichera une box qui n'acceptera que les nombres,
et proposera des flèches pour augmenter ou
diminuer la valeur.

min="1" : Indique une valeur minimum

max="1" : Indique la valeur maximum

step="1" : Indique le « saut » (1 par défaut)

Pseudo :
Mot de passe :
Courriel
Téléphone
Age 30

HTML-CSS-JS

Les Formulaires

Les zones de saisie <input> : le type range

```
<input type="range" name="range" id="range">
```

Affiche un curseur de progression

min="1" : Indique une valeur minimum

max="1" : Indique la valeur maximum

step="1" : Indique le « saut » (1 par défaut)



Pseudo :
Mot de passe :
Courriel :
Téléphone :
Age :
Humeur du jour :

Les autres paramètres peuvent être trouvés sur la doc :
<https://developer.mozilla.org/fr/docs/Web/HTML/Element/Input/range>

HTML-CSS-JS

Les Formulaires

Les zones de saisie <input> : le type color

```
<input type="color" id="color" name="color">
```

Affiche un « *color picker* » sur la page.

value="#ff0000" : définit une couleur par défaut.



La doc :

<https://developer.mozilla.org/fr/docs/Web/HTML/Element/Input/color>

HTML-CSS-JS

Les Formulaires

Les zones de saisie <input> : le type date

<input type="date">

Affiche un « *date picker* » sur la page.

Exsistent aussi les types « *time* » pour l'heure,
« *week* » pour la semaine, « *month* » pour le mois etc.

Pseudo :
Mot de passe :
Courriel
Téléphone
Age
Humeur du jour
Couleur préférée
Date de naissance :

*En cas de doute sur la compatibilité des navigateurs, pensez à vérifier
sur <https://caniuse.com/input-datetime>*

HTML-CSS-JS

Les Formulaires

Les zones de texte :

```
<textarea name="description" id="description" cols="30" rows="5"></textarea>
```

Permet d'insérer une zone de texte.

- COLS : largeur (en nombre de caractères)
- ROWS : Lignes

Il est possible de préremplir la textarea en indiquant du texte entre les balises.

Décrivez vous :

La mise en forme peut aussi être faite coté CSS avec les attributs width et height.

HTML-CSS-JS

Les Formulaires

Les checkbox:

```
<form action="d" method="post">
  <h2>Quel langage connaissez vous ?</h2>
  <p>
    <input type="checkbox" name="js" id="js"> <label for="js">JavaScript</label> <br>
    <input type="checkbox" name="ts" id="ts"> <label for="ts">TypeScript</label> <br>
    <input type="checkbox" name="ps" id="ps"> <label for="ps">PowerShell</label> <br>
    <input type="checkbox" name="shell" id="shell"> <label for="shell">Linux Shell</label>
  </p>
</form>
```

Chaque checkbox doit avoir un name différent, afin de pouvoir exploiter leur valeur !

Quel langage connaissez vous ?

- JavaScript
- TypeScript
- PowerShell
- Linux Shell

L'attribut « checked » peut être ajouté afin que la box soit cochée par défaut. L'attribut en question n'a pas besoin de valeur.

HTML-CSS-JS

Les Formulaires

Les radio button:

```
<form action="d" method="post">
  <h2>Quel est votre langage préféré ?</h2>
  <input type="radio" name="langage" id="js" value="js"> <label for="js">Javascript</label> <br>
  <input type="radio" name="langage" id="ts" value="ts"> <label for="ts">Typescript</label> <br>
  <input type="radio" name="langage" id="shell" value="shell"> <label for="shell">Shell</label>
</p>
</form>
```

Comme pour la checkbox, on précise un label.

Cette fois, tous nos boutons ont le même name !

Quel est votre langage préféré ?

- Javascript
- Typescript
- Shell

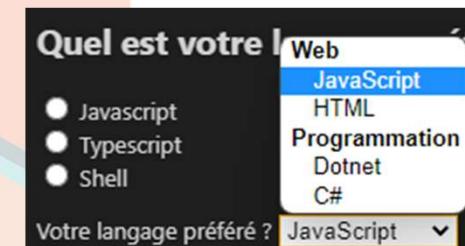
Value correspond à la valeur que nous récupérerons !

HTML-CSS-JS

Les Formulaires

Les radio listes de sélection :

```
<form action="d" method="post">
<p>
  <label for="favLang">Votre langage préféré ?</label>
  <select name="favLang" id="favLang">
    <optgroup label="Web">
      <option value="js">JavaScript</option>
      <option value="ts">HTML</option>
    </optgroup>
    <optgroup label="Programmation">
      <option value="dotnet">Dotnet</option>
      <option value="cs">C#</option>
    </optgroup>
  </select>
</p>
</form>
```



<optgroup> permet de ranger les options dans des groupes et est facultatif.

<option> sert à définir les différentes options de notre menu déroulant.

<value> permettra de récupérer la valeur saisie.

HTML-CSS-JS

Les Formulaires

Un peu de rangement avec les `fieldset` :

```
<form action="d" method="post">
  <fieldset>
    <legend>Identité</legend>

    <p>
      <label for="name">Pseudo : </label>
      <input type="name" name="name" id="name" autocomplete="name" />
    </p>
    <p>
      <label for="email">Courriel</label>
      <input type="email" name="email" id="email" />
    </p>
    <p>
      <label for="pswd">Mot de passe :</label>
      <input type="password" name="pswd" id="pswd" />
    </p>
    <p>
      <label for="tel">Téléphone</label>
      <input type="tel" name="tel" id="tel" />
    </p>
    <p>
      <label for="date">Date de naissance : </label>
      <input type="date" />
    </p>
  </fieldset>

  <fieldset>
    <legend>Skills</legend>
    <p>
      <h2>Quel est votre langage préféré ?</h2>
      <input type="radio" name="langage" id="js" value="js"> <label for="js">Javascript</label> <br>
      <input type="radio" name="langage" id="ts" value="ts"> <label for="ts">TypeScript</label> <br>
      <input type="radio" name="langage" id="shell" value="shell"> <label for="shell">Shell</label>
    </p>
  </fieldset>
</form>
```



The screenshot shows a web page with a dark background. It features two `fieldset` elements. The first `fieldset` is labeled "Identité" and contains five input fields: "Pseudo", "Courriel", "Mot de passe", "Téléphone", and "Date de naissance" with a date input field. The second `fieldset` is labeled "Skills" and contains the question "Quel est votre langage préféré ?" followed by three radio buttons for "Javascript", "TypeScript", and "Shell".

`<fieldset>` permet de regrouper les champs de notre formulaire sous des labels.

`<legend>` nous permet de nommer les groupes de champs.

HTML-CSS-JS

Les Formulaires

Quelques attributs pour les champs:

```
<input type="text" name="name" id="id" autofocus>
```

Autofocus : centrera automatiquement la page sur le champs

```
<input type="text" name="name" id="id" required>
```

Required : Indique que ce champs ne peut être vide

Semifir

HTML-CSS-JS

Les Formulaires

Les boutons de validation:

```
<input type="submit" value="Submit">
```

Permet de créer un bouton d'envoi standard.
Renverra vers la page de l'attribut « action » du formulaire.

Submit

```
<input type="reset" value="Reset">
```

Remet le formulaire à zéro (vide les champs)

Reset

Semifir

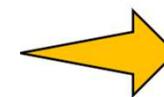
HTML-CSS-JS

Les Formulaires

Les boutons de validation:

```
<input type="button" value="Button">
```

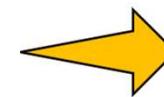
Bouton générique. N'a aucun effet par défaut. Sera en général relié à un script.



Button

```
<input type="image" src="https://pic.onlinewebfonts.com/svg/img_80698.png" height="50" alt="">
```

Bouton de type « submit » avec une image personnalisée à spécifier dans l'attribut « src »



II. HTML 5

Bonnes pratiques

HTML



HTML-CSS-JS

HTML – Best Practices

Les bonnes pratiques

Comme pour tous les langages, il existe des bonnes pratiques.

Si les bonnes pratiques ne sont pas obligatoires, elles sont vivement recommandées !

Semifir

Be Best Practice

HTML-CSS-JS

HTML – Best Practices



Déclarez le type du document :

```
<!DOCTYPE html>
<html lang="fr">
```

Semifir

Comme pour tout les langages, il existe des bonnes pratiques !

HTML-CSS-JS

HTML – Best Practices



Remplissez le header de vos pages HTML :

```
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" >
    <meta http-equiv="X-UA-Compatible" content="IE=edge" >
    <meta name="keywords" content="demo, html5, html, CSS" >
    <meta name="author" content="Alexandre Devos" >
    <meta name="viewport" content="width=device-width, initial-scale=1.0" >
    <title>Titre du document</title>
    <link rel="stylesheet" type="text/css" href="chemin/vers/css" >
    <link rel="shortcut icon" href="chemin/vers/icon" type="image/x-icon" >
</head>
```

Semifir

Dans VS CODE : tapez juste « html:5 »

HTML-CSS-JS

HTML – Best Practices



Pensez à refermer vos balises :



```
<h1>Balise bien fermée</h1>
<h1>Balise mal fermée
<input type="text" value="balise auto-fermante">
```

Semifir

Ne pas respecter les conventions est le meilleur moyen de se faire haïr par ses collègues !

HTML-CSS-JS

HTML – Best Practices

Indiquez le nom des attributs, id et classes en **minuscule** :



```
<balise attribu="mavaleur" id="en-minuscule"></balise>
```

Ne pas respecter les conventions est le meilleur moyen de se faire haïr par ses collègues !

HTML-CSS-JS

HTML – Best Practices

Ne croisez pas vos balises !

```
<a>
  <b> Texte </b>
</a>
```

```
<a>
  <b> Texte </a>
</b>
```



Semifir

Sérieusement : ne faites pas ça.

HTML-CSS-JS

HTML – Best Practices



Les commentaires servent en général à donner des informations sur la structure ou donner des informations pertinentes.

```
<niveau0>
    <niveau1></niveau1>
    <niveau1></niveau1>
    <!-- Gérard, n'oublie pas de retirer la faille de sécurité dans le PHP -->
</niveau0>
```

Semifir
Ne confondez pas les commentaires avec un tchat !

HTML-CSS-JS

HTML – Best Practices

Respectez l'indentation, et ce quel que soit le langage !

```
<niveau0>
    <niveau1></niveau1>
        <niveau1></niveau1>
    <niveau1>
        <niveau2>
<niveau3>MAL indenté<niveau3>
        </niveau2>
    </niveau1>
```



```
<niveau1>
    <niveau2>
        <niveau3>BIEN indenté<niveau3>
    </niveau2>
</niveau1>
</niveau0>
```



Alt+Shift+F pour indenter automatiquement !

III. CSS 3

Présentation



HTML-CSS-JS

CSS - Présentation

Présentation

- Cascading Style Sheet
- Également appelée « feuille de style »
- Langage vient compléter le HTML
- Sert à la mise en forme du site
- Permet de gérer la couleur du texte, changer la police, ajouter des bordures, des couleurs de fond...



Pensez à toujours vérifier la compatibilité du CSS avec les navigateurs avec www.caniuse.com

HTML-CSS-JS

CSS - Présentation

Présentation

- Le CSS peut être rédigé :
 - Dans un fichier .css
 - Dans le head du fichier HTML
 - Dans les balises
- Il est vivement recommandé de rédiger son css dans un fichier .css :
 - Best practice,
 - Un même fichier pourra être appelé par plusieurs pages HTML,

Soyez propres, soyez best practice : utilisez un fichier .css !

HTML-CSS-JS

CSS - Présentation

- Pour relier notre page HTML à notre feuille de style, nous ajoutons la ligne
`<link rel="stylesheet" href="style.css" />`
dans le HEAD de notre page.
- Ici, la feuille de style se nomme « style.css » et se trouve dans le même répertoire.

III. CSS 3

Les sélecteurs



HTML-CSS-JS

CSS

Appliquer un style

On peut appliquer un style :

- A toutes les balises d'un même type
- A toutes les balises ayant l'attribut un « classe » spécifique
- A une balise ayant un id particulier
- A du texte contenu dans une balise avec

Pour désigner ces éléments, nous utiliseront ce que l'on appelle

un sélecteur

La doc :

https://developer.mozilla.org/fr/docs/Glossary/CSS_Selector

HTML-CSS-JS

CSS – Les sélecteurs

Sélecteur de type :

Il suffit de préciser le nom de la balise pour que toutes les balises de ce type soient stylisées comme indiqué !

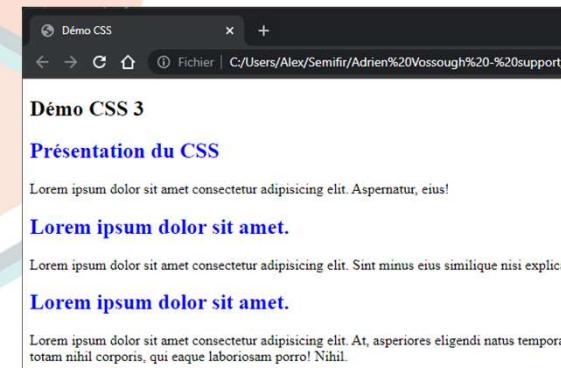
Sélecteur

CSS

```
h2
{
    color: blue;
}
```

Propriété

Valeur



Si on souhaite indiquer plusieurs balises, on peut les lister en les séparant d'une virgule

HTML-CSS-JS

CSS – Les sélecteurs

Sélecteur de classe :

On préfixe d'un point le nom de la classe pour appliquer notre style à toutes les balises ayant cette classe en attribut :

CSS

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

HTML

```
<p class="maClasse">  
    Lorem ipsum dolor sit amet  
    consectetur adipisicing elit. Aspernatur,  
    eius!  
</p>
```



Très pratique pour appliquer notre style à plusieurs éléments !

HTML-CSS-JS

CSS – Les sélecteurs

Sélecteur d'ID:

On écrit le nom de l'ID précédé d'un « # » pour appliquer notre style à la balise ayant cet ID :

CSS

```
#h2_amet
{
    color: green
}
```

HTML

```
<h2 id="h2_amet">Lorem ipsum dolor sit amet.</h2>
```



Démo CSS 3

Présentation du CSS

Lorem ipsum dolor sit amet consectetur adipisicing elit. Aspernatur, eius!

Lorem ipsum dolor sit amet.

Lorem ipsum dolor sit amet consectetur adipisicing elit. Sint minus eius similique nisi explicabo iure :

Lorem ipsum dolor sit amet.

Lorem ipsum dolor sit amet consectetur adipisicing elit. At, asperiores eligendi natus tempora vel voit totam nihil corporis, qui eaque laboriosam porro! Nihil.

Chaque ID devant être unique, seule une balise sera affectée !

HTML-CSS-JS

CSS – Les sélecteurs

Appliquer un style à un élément contenu dans une balise :

On utilisera la même méthode que pour appliquer du style à une classe.

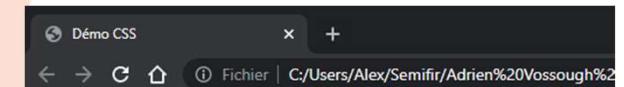
Coté HTML, nous insérerons notre contenu dans une balise

CSS

```
.green
{
    color: green
}
```

HTML

```
<p class="maClasse">
    Lorem ipsum <span class="green"> dolor </span> sit amet
    consectetur adipisicing elit. Aspernatur,
    eius!
</p>
```



Démo CSS 3

Présentation du CSS

Lorem ipsum dolor sit amet consectetur adipisicing elit. Aspernatur, eius!

Lorem ipsum dolor sit amet.

 Lorem ipsum dolor sit amet consectetur adipisicing elit. Sint minus eius similiq

Lorem ipsum dolor sit amet.

 Lorem ipsum dolor sit amet consectetur adipisicing elit. At, asperiores eligendi totam nihil corporis, qui eaque laboriosam porro! Nihil.

Ici, un seul mot a été colorisé en vert, alors que le paragraphe lui-même disposait de son propre style !

HTML-CSS-JS

CSS – Les sélecteurs avancés

Combinateurs de sélecteurs :

`div h2 /* Style */`

Appliquera le style aux h2 qui ont pour parent une division

`h2 ~ h3 /* Style */`

Permet de cibler un élément précédé d'un autre même s'il ne sont pas voisins directs

`h2 > p /* Style */`

Cible tous les paragraphes `<p>` enfants directs d'un `<h2>`

`h2 + p /* Style */`

Applique le style à la première balise P située après un h2

Retenez cette syntaxe : elle sera utile plus tard pour cibler des éléments en JavaScript !

HTML-CSS-JS

CSS – Les sélecteurs avancés

Quelques sélecteurs d'attributs :

```
h2[title] /* Style */
```

Appliquera le style aux paragraphes <p> ayant un attribut « title »

```
h2[title="monTitre"] /* Style */
```

Appliquera le style aux balises <h2> dont l'attribut « title » est égal à « monTitre ».

Doc :

https://developer.mozilla.org/fr/docs/Web/CSS/Attribute_selectors

HTML-CSS-JS

CSS – Les sélecteurs avancés

Les pseudo-éléments :

Un pseudo-élément permet de désigner une partie d'un élément. Il existe beaucoup de possibilités que vous pouvez retrouver sur la doc. On utilise le sélecteur habituel suivit de « :: » pour préciser à quel partie la propriété doit être appliquée.

En voici quelques exemples :

`p::first-line /* Style */`

Appliquera le style à la première ligne
d'un paragraphe <p>

`::placeholder /* Style */`

Applique le style au texte de
remplacement d'une textbox.

<https://developer.mozilla.org/fr/docs/Web/CSS/Pseudo-elements>

HTML-CSS-JS

CSS – Les sélecteurs avancés

Les pseudo-classes :

Utilisent une notation similaire aux pseudo-éléments mais concernent l'état de l'élément (contrairement au pseudo élément qui concerne une partie de l'élément).

Comme pour les pseudo-éléments, il en existe beaucoup !

```
a:visited /* Style */
```

Applique le style aux liens déjà visités

```
p:not(.monStyle) /* Style */
```

Concerne tous les paragraphes <p> qui ne sont pas concernés par la classe « monStyle »

<https://developer.mozilla.org/fr/docs/Web/CSS/Pseudo-classes>

III. CSS 3

Les boîtes

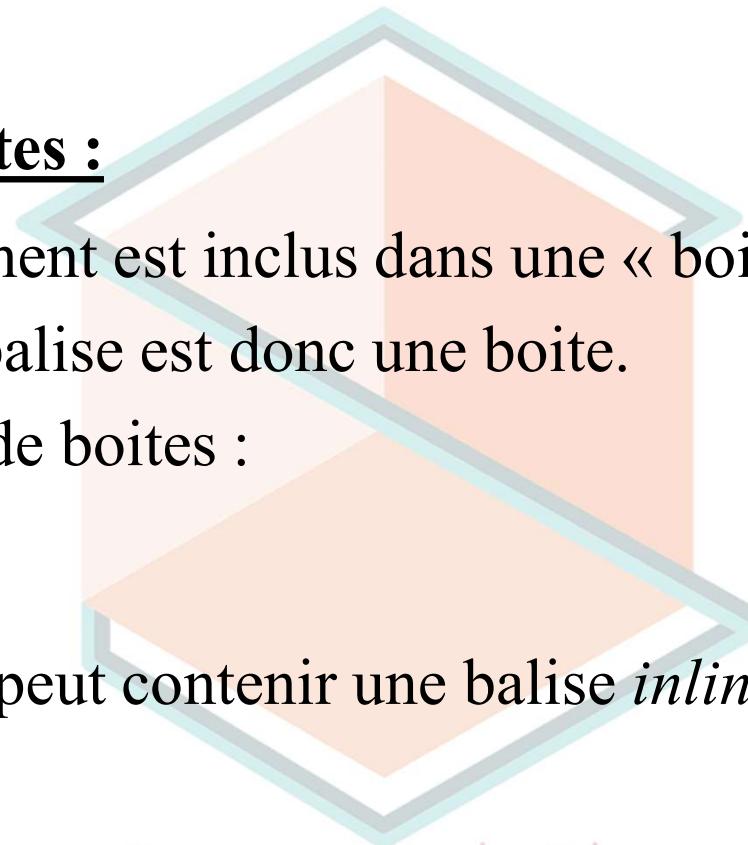


HTML-CSS-JS

Les boites

Rappel sur les boites :

- En CSS, tout élément est inclus dans une « boite » (box)
- Finalement, une balise est donc une boite.
- Il y a deux types de boites :
 - *Block*
 - *Inline*
- Une balise *Block* peut contenir une balise *inline*.

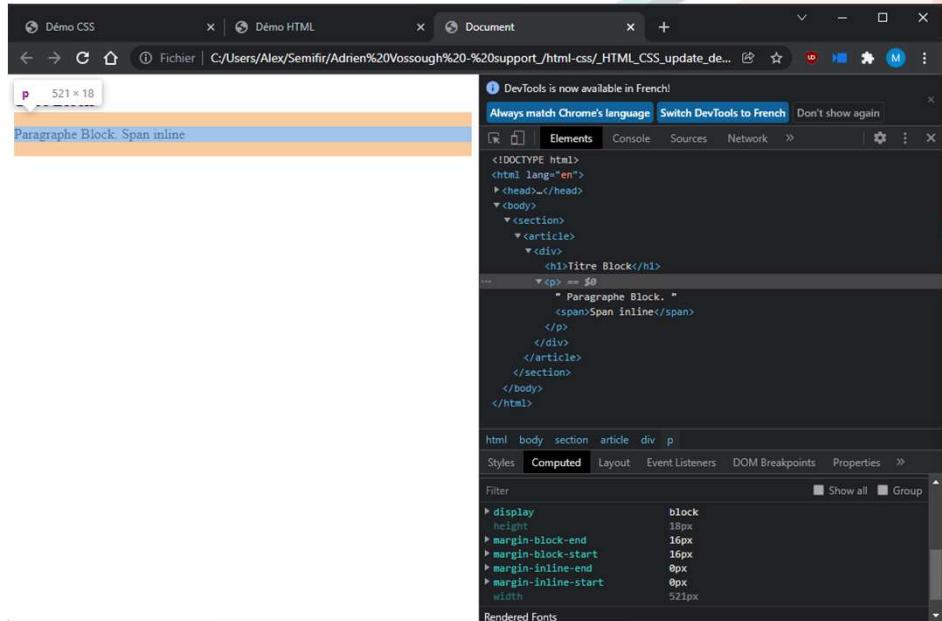


Semifir

Doc : https://developer.mozilla.org/fr/docs/Learn/CSS/Building_blocks/The_box_model

HTML-CSS-JS

Les boîtes



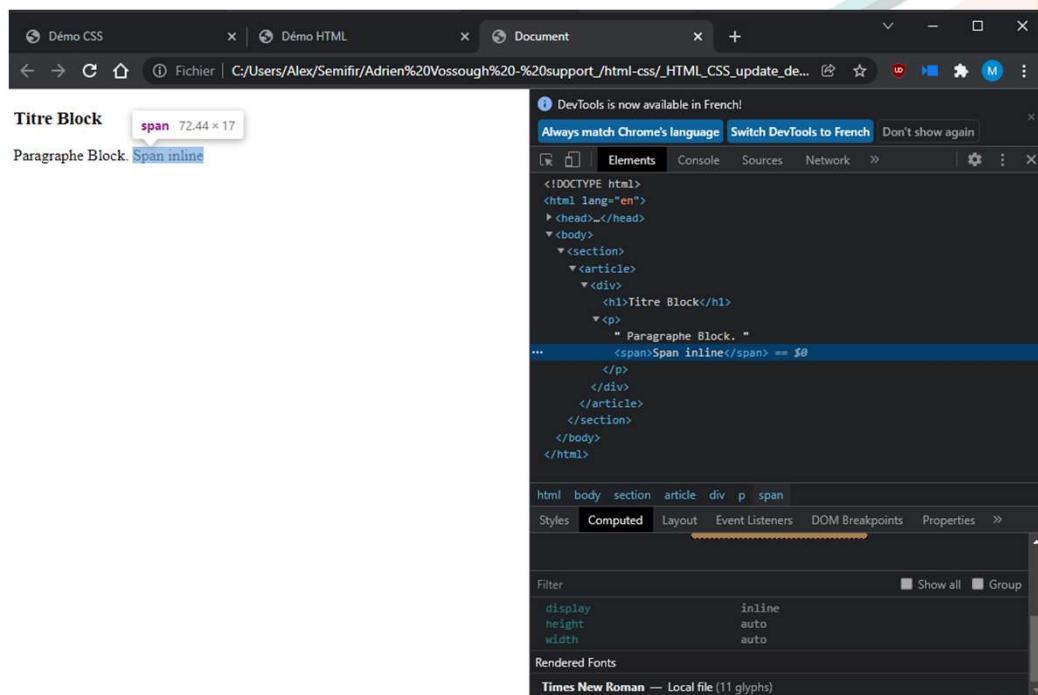
Le type : *Block*

- Une balise de type *Block* prend 100% de la largeur disponible,
- Elle peut contenir des balises de type *Inline*,
- Un retour à la ligne est automatiquement créé avant et après la balise.

Appuyez sur F12 : vous pourrez survoler les éléments constituant la page HTML et voir la place qu'ils prennent.

HTML-CSS-JS

Les boites



Le type : *Inline*

- Une balise de type *Inline* prend la place dont elle a besoin.
- Sa taille ne peut être redéfinie

Appuyez sur F12 : vous pourrez survoler les éléments constituant la page HTML et voir la place qu'ils prennent.

HTML-CSS-JS

Les boites

Transformation :

`display: block`

- Grâce à ce paramètre, un block peut passer d'un type à un autre
- On peut le transformer en block :
 - *Inline*,
 - *Block*,
 - *Inline-block*
 - Elément *inline* ayant les propriétés d'un élément *block*
 - Il peut contenir n'importe quel élément,
 - On peut définir sa taille.

HTML-CSS-JS

Les boites

Changer la taille d'un élément :

width: 50%;

Permet de définir la largeur.

height: 100%;

Permet d'affecter la hauteur.

Ces paramètres sont à indiquer dans le block CSS de l'élément concerné en tant que paramètre. Il est dans les deux cas possible d'utiliser les pourcentages (de la taille disponible), les em (taille relative à la police utilisée) ou les pixels en guise d'unité.

Il existe également le paramètre « auto »

Les unités de mesure (cm, pt) existent mais sont fortement déconseillés.

Tout se passe dans le CSS !

HTML-CSS-JS

Les boites

Il est aussi possible de déterminer des hauteurs/largeurs minimum et maximum afin de permettre au site de s'adapter aux différents supports et résolutions.

min-height: 50%;

Hauteur minimum

min-width: 50%;

Largeur minimum

max-height: 50%;

Hauteur maximum

max-width: 50%;

Largeur maximum

Doc : <https://developer.mozilla.org/fr/docs/Web/CSS/width>

HTML-CSS-JS

Les boîtes



Les marges

- Les blocs peuvent aussi disposer de marges
- Leur valeurs s'ajoutent à la hauteur et à la largeur définie si le bloc en possède.
- Chaque élément disposera de ses propres marges par défaut !

Semifir

Les marges par défaut peuvent être à zéro

HTML-CSS-JS

Les boîtes



Les marges

- Padding : Taille de la marge intérieure
- Border : Epaisseur de la bordure
- Margin : taille de la marge extérieure

Semifir

HTML-CSS-JS

Les boites

Changer la taille des marges :

margin: 50px;

Permet de définir la marge extérieure (entre la bordure et l'extérieur)

padding: 50px;

Permet de définir la marge intérieure (entre l'élément et la bordure)

Les marges ainsi paramétrées dans le CSS seront appliquée sur les bords à la fois en haut, en bas, à gauche et à droite !

Semifir

Tout se passe dans le CSS !

HTML-CSS-JS

Les boites

Changer la taille d'une marge :

Il est aussi possible de déterminer la taille des marges pour un seul des 4 cotés :

margin-top: 50px;

Définit la taille de la marge haute

margin-bottom: 50px;

Définit la taille de la marge basse

margin-left: 50px;

Définit la taille de la marge de gauche

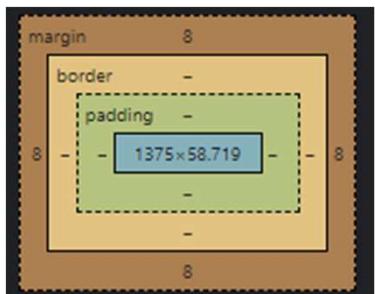
margin-right: 50px

Définit la taille de la marge de droite

Les paramètres sont aussi valables pour padding et border, en suivant la même logique.

HTML-CSS-JS

Les boites



Les contenu

- Si une boite possède des marges définies, il peut arriver que le texte « déborde » car il prend plus de place que la boite
- Pour palier à ce problème, nous utiliseront « **overflow** »

mifir

HTML-CSS-JS

Les boites

Régler les problèmes de dépassement du block:

Le paramètre « overflow » permet d'indiquer quoi faire si le contenu du bloc dépasse :

overflow: visible;

Par défaut : si le texte dépasse, il reste visible et sort du block

overflow: hidden;

Si le texte dépasse, il est coupé

overflow: scroll;

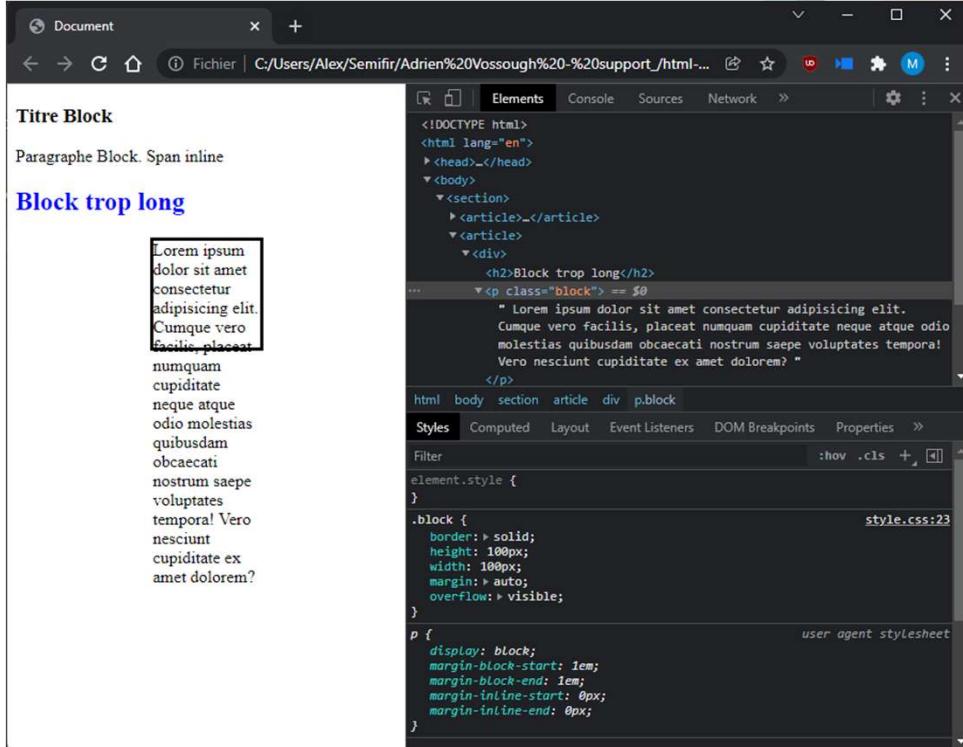
Ajoute une barre de navigation au block en cas de dépassement du texte.

overflow: auto;

Laisse le navigateur définir s'il faut ajouter une barre de défilement ou non.

HTML-CSS-JS

Les boites : overflow visible



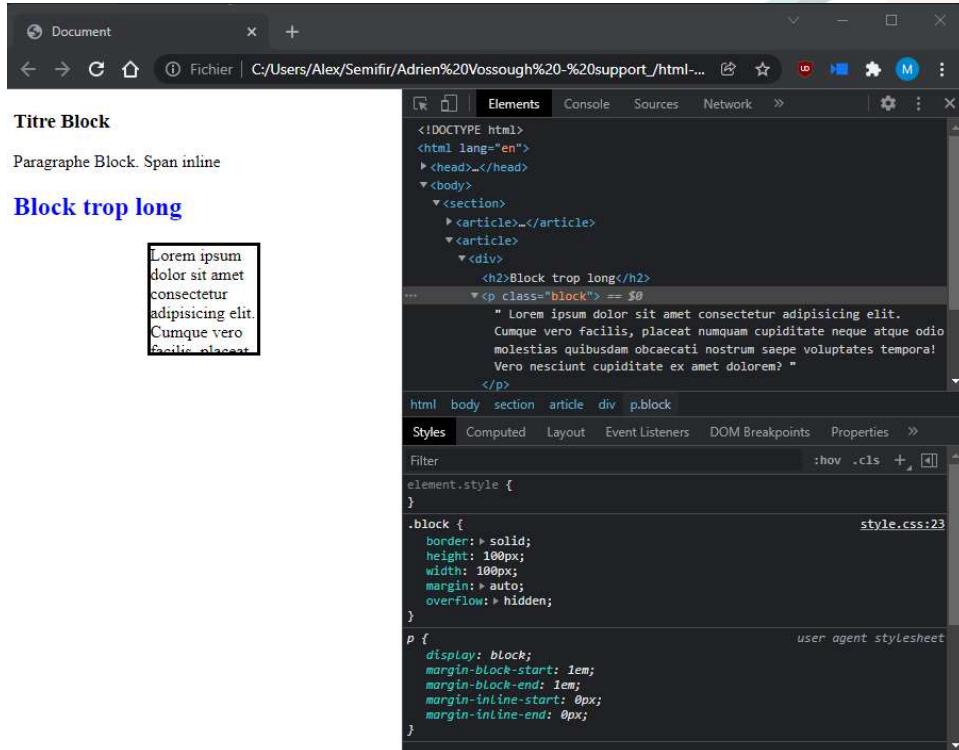
Exemple avec « visible »

```
.block
{
    border: solid;
    height: 100px;
    width: 100px;
    margin: auto;
    overflow: visible;
}
```

NB : Visible est la valeur par défaut

HTML-CSS-JS

Les boites : overflow visible

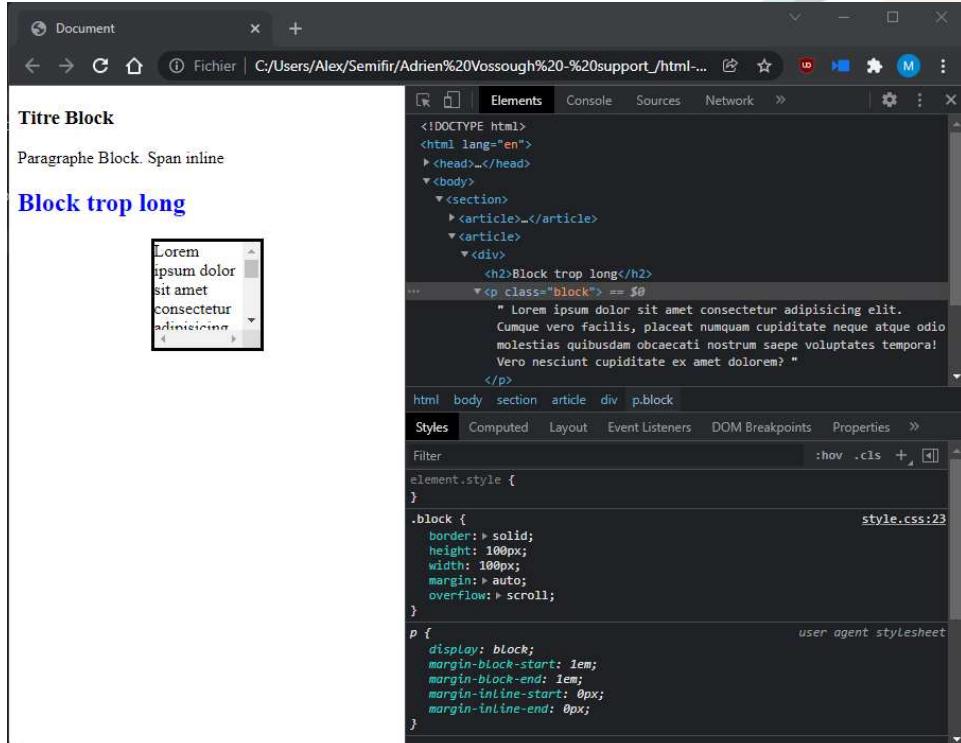


Exemple avec « hidden »

```
.block
{
    border: solid;
    height: 100px;
    width: 100px;
    margin: auto;
    overflow: hidden;
}
```

HTML-CSS-JS

Les boites : overflow visible

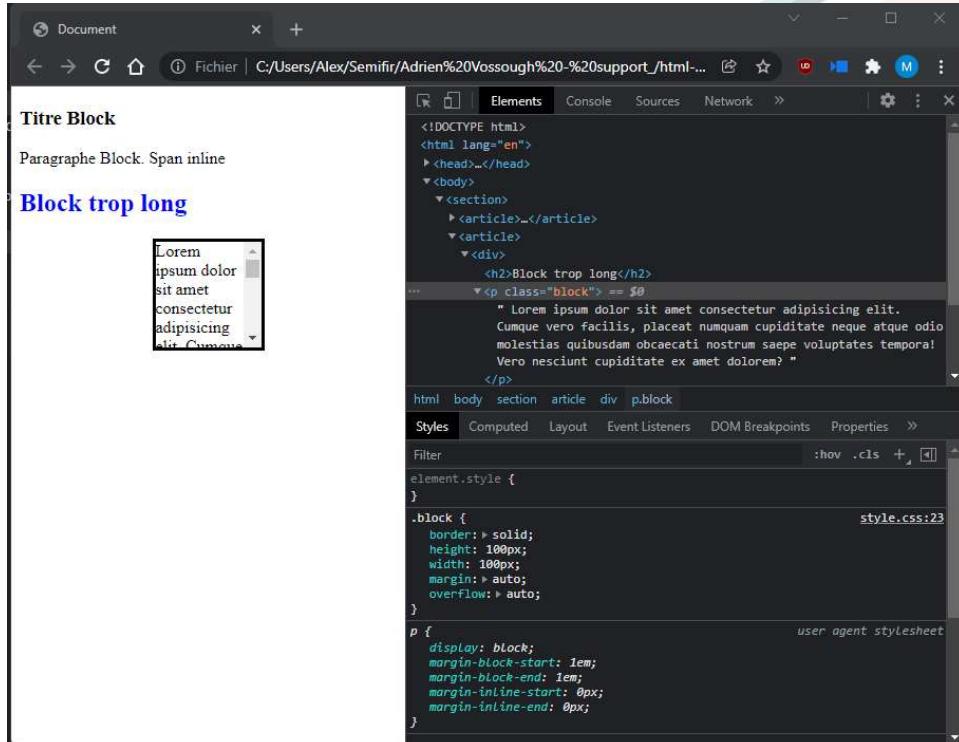


Exemple avec « scroll »

```
.block
{
    border: solid;
    height: 100px;
    width: 100px;
    margin: auto;
    overflow: scroll;
}
```

HTML-CSS-JS

Les boites : overflow visible



Exemple avec « auto »

```
.block
{
    border: solid;
    height: 100px;
    width: 100px;
    margin: auto;
    overflow: auto;
}
```

Le paramètre est le plus souvent utilisé en mode automatique

III. CSS 3

Les flexbox

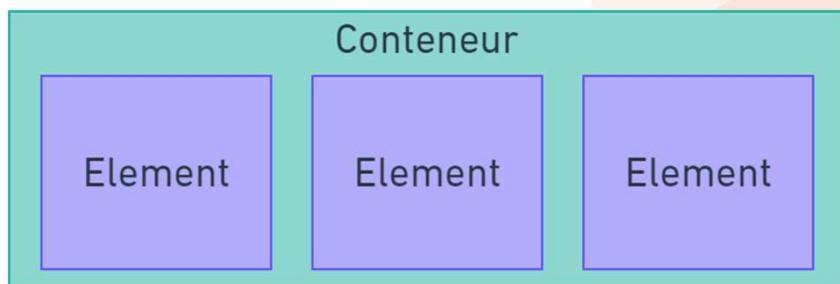


HTML-CSS-JS

Les FlexBox

Les flexbox

- Les flexbox fonctionnent avec des **conteneurs** et des **éléments**.
- Un conteneur est un élément qui contient d'autres éléments.



```
<div id="conteneur">  
  <div class="element 1">Element 1</div>  
  <div class="element 2">Element 2</div>  
  <div class="element 3">Element 3</div>  
</div>
```

Semifir

HTML-CSS-JS

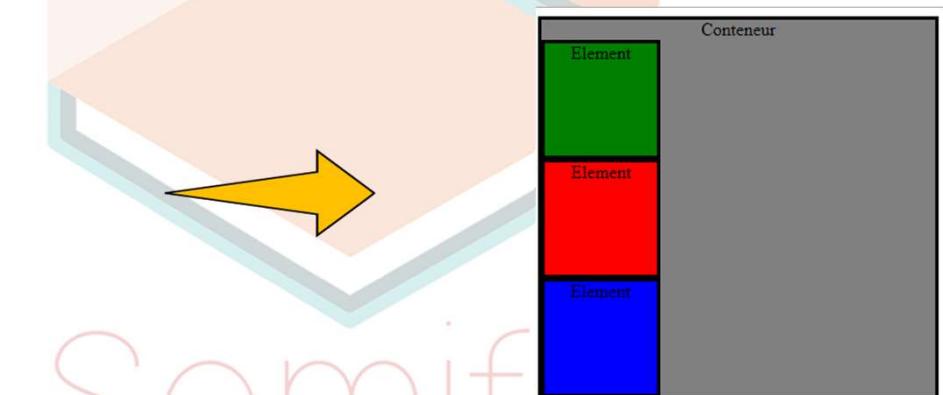
Les FlexBox

```
#conteneur
{
    border: solid;
    background-color: grey;
    text-align: center
}

#conteneur div
{
    border: solid;
    height: 100px;
    width: 100px;
    text-align: center;
}

.element1
{
    background-color: green
}
.element2
{
    background-color: red
}
.element3
{
    background-color: blue
}
```

- Par défaut, les éléments contenus dans un conteneur sont placés les uns en dessous des autres.
- Flexbox permet de définir comment ces derniers vont dynamiquement se positionner en fonction de l'espace disponible.

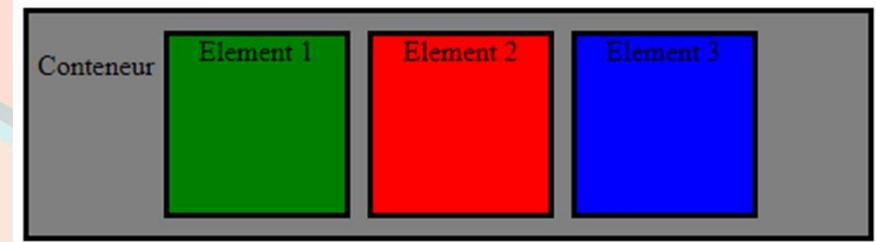


HTML-CSS-JS

Les FlexBox

Voici le résultat avec flexbox :

```
#conteneur
{
    display: flex;
}
```



- Le paramètre est ajouté au sélecteur de notre conteneur

Semifir

Le tout en ajoutant une seule ligne !

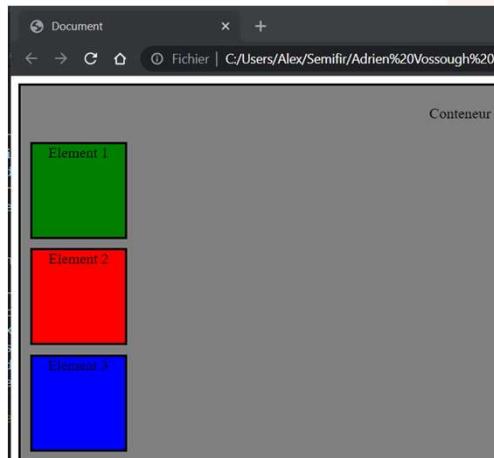
HTML-CSS-JS

Les FlexBox

On peut choisir comment seront disposés nos éléments dans le conteneur avec un paramètre :

flex-direction: column;

flex-direction: row;



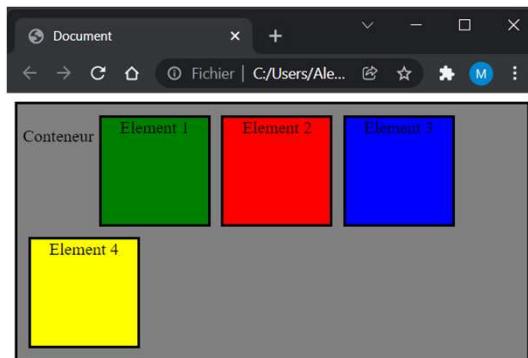
Chaque option dispose d'une version « reverse » qui inverse l'ordre des colonnes.

HTML-CSS-JS

Les FlexBox

L'option « flex-wrap » permet de changer dynamiquement la position des éléments en fonction de la taille disponible (fenêtre/écran). Il vient en complément de « flex-direction »

flex-wrap: wrap;



flex-wrap: nowrap;



Semifir

Dispose également d'un version « reverse »

HTML-CSS-JS

Les FlexBox

L'axe sur lequel sont organisés nos éléments (horizontal par défaut) est appelé :

l'axe principal

L'autre axe est appelé **l'axe secondaire** (cross axis)

justify-content: option;

Changera l'alignement sur l'axe principal

align-items: option;

Changera l'alignement sur l'axe secondaire

Semifir

Dispose également d'un version « reverse »

HTML-CSS-JS

Les FlexBox

Les valeurs possibles pour *jutify-content* :

`justify-content: flex-start`

Alignés au début

`justify-content: center`

Alignés au centre

`justify-content: space-around`

Idem between, mais avec espace sur les cotés en plus.

`justify-content: flex-end`

Alignés à la fin

`justify-content: space-between`

Etirés sur tout l'axe. De l'espace est laissé entre les éléments.

HTML-CSS-JS

Les FlexBox

Les valeurs possibles pour *align-items* :

align-items: flex-start

Alignés au début

align-items: center

Alignés au centre

align-items: baseline

Alignés sur la ligne de base (semblable à flex-start).

align-items: flex-end

Alignés à la fin

align-items: stretch

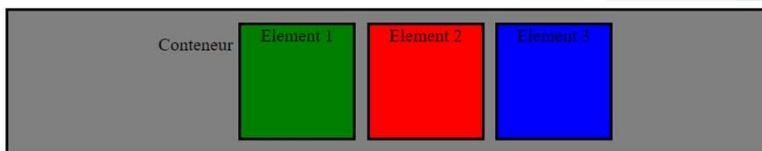
Les éléments sont étirés sur tout l'axe

HTML-CSS-JS

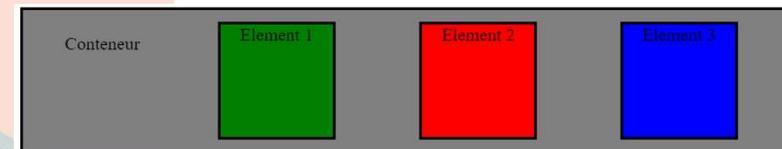
Les FlexBox

Quelques exemples :

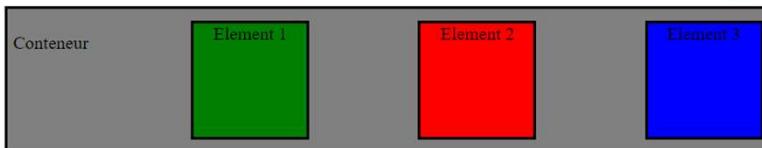
justify-content: center;



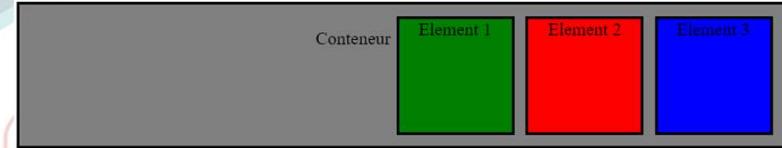
justify-content: space-around;



justify-content: space-between;



justify-content: flex-end;



semifir

Training time !

Réalisez les exercices de « Flexbox Froggy » :

<https://flexboxfroggy.com/#fr>

III. CSS 3

Positionnement



HTML-CSS-JS

Le positionnement

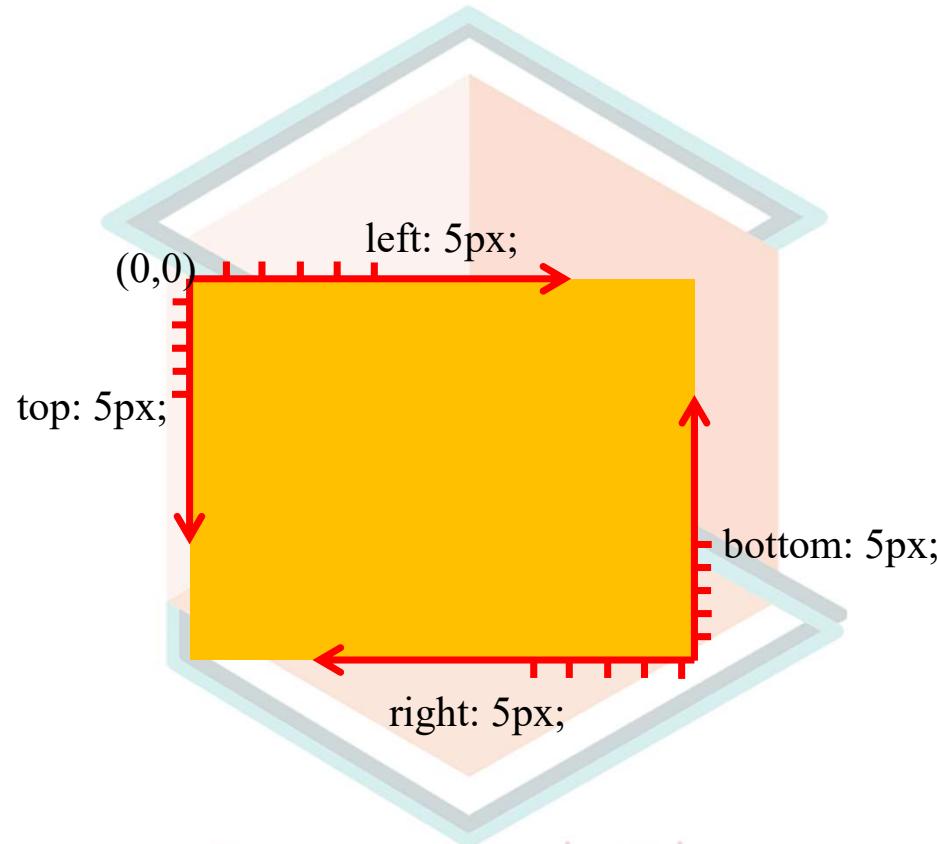
Présentation :

- Le l'attribut « position » nous permet de définir la position d'un élément
- Il dispose de plusieurs paramètres :
 - Static : positionnement par défaut,
 - Relative : position de l'élément par rapport à son emplacement d'origine,
 - Absolute : position de l'élément par rapport au premier parent,
 - Fixed : position par rapport à l'écran

Semifir

HTML-CSS-JS

Le positionnement



Semifir

Le point zéro se trouve dans le coin supérieur gauche.

HTML-CSS-JS

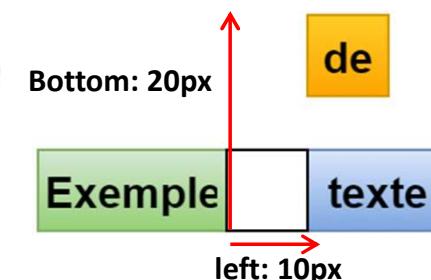
Le positionnement

Positionnement relatif :

La position relative se fait par rapport à la position initiale de l'élément :

```
<span>exemple </span>
<span class="relat">de</span>
<span> texte</span>
```

```
.relat
{
    position: relative;
    bottom: 20px;
    left: 10px;
}
```



HTML-CSS-JS

Le positionnement

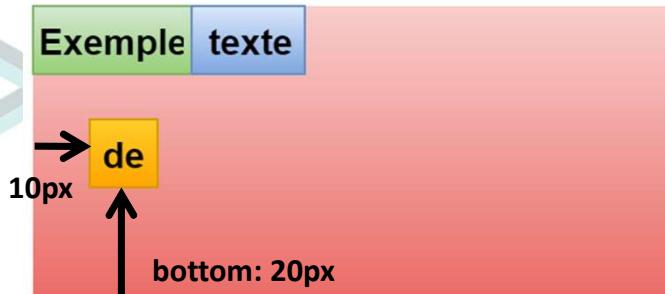
Positionnement absolu :

- La position « absolute » se fait par rapport au premier parent **positionné** (premier parent ayant un attribut « position: »).
- Si aucun parent n'est positionné, l'élément se positionnera par rapport à son conteneur principal

```
<div>
  <span>exemple </span>
  <span class="relat">de</span>
  <span> texte</span>
</div>
```



```
.div {
  position: relative;
  height: 100px;
  background-color: red;
}
.relat {
  position: absolute;
  bottom: 20px;
  left: 10px;
}
```



NB : l'élément en position absolute sors du flux !

HTML-CSS-JS

Le positionnement

Positionnement fixe :

- Quasi identique au positionnement absolu,
- L'élément restera visible et ne bougera pas, même si on descend la page

```
.fixed {  
background: orange;  
width: 100px;  
height: 100px;  
margin: 20px;  
color: white;  
text-align: center;  
}  
#txt {  
position: fixed;  
top: 80px;  
left: 10px;  
}  
.corpo {  
width: 500px;  
height: 300px;  
overflow: scroll;  
padding-left: 150px;  
}
```



```
<div class="corpo">  
  <p>  
    Lorem ipsum [...] eum?  
  </p>  
  <p>  
    Lorem ipsum [...]quod! Nemo, a.  
  </p>  
  <div class="fixed" id="txt">Semifir</div>  
</div>
```

namquam ipsam ratione dolore, aliquam non error sed: explicabo officia
aspernatur impedit architecto atque unde voluptatibus deleniti quas
pariatur. Aut pariatur libero nostrum doloribus exercitationem eaque id
quos, obcaecati aliquam nulla quasi culpa placeat! Laudantium maxime et,
nam est vel omnis delectus cum eius corporis eligendi praesentium, quam
tempora amet soluta blanditiis quas voluptatem dicta. Quasi praesentium
qui laudantium aut. Vel praesentium repellat in explicabo quo doloribus?
Eveniet illum harum eumque soluta adipisci, fugiat sapiente, quos optio
explicabo fuga architecto eum?

lorem ipsum dolor sit amet consectetur adipisicing elit. Perspiciatis quis
ipsa expedita, ducimus ut in inventore ipsum, dolorem rem vitae voluptas,
odio similiquid? Voluptate perferendis laudantium exercitationem, dolorum,
ad vero fugit suscipit voluptas, optio facilis aut aspernatur labore
consequuntur ex voluptates. Voluptas dolores et non est esse. Molestias
nisi, nemo quod doloremque commodi nulla deleniti, neque quidem,

Semifir

Semifir

Très pratique pour les navbar par exemple !

HTML-CSS-JS

Le positionnement

« float » :

- Sort l'élément du flux et le place du côté gauche ou droit de son parent

```
<div>
  <span>exemple </span>
  <span class="flt flt0">de</span>
  <span class="flt"> texte</span>
</div>
```

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

```
.flt0 {
  float: left;
}
```

de Exemple texte

```
.flt0 {
  float: right;
}
```

Exemple texte

```
.flt {
  float: left;
}
```

de texte Exemple

```
.flt {
  float: right;
}
```

Exemple

texte de

L'attribut peut aussi être utilisée pour entourer une image de texte

HTML-CSS-JS

Le positionnement

Position z-index :

- Donne la profondeur des éléments,
- La valeur est un entier,
- Il est possible d'utiliser des valeurs négatives,
- Plus la valeur est élevée, plus l'élément sera « en haut de la pile ».



III. CSS 3

Media queries



HTML-CSS-JS

Media Queries

Présentation

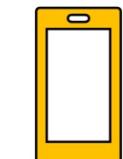
- Permettent d'adapter dynamiquement un site aux différents supports
- Sont des règles CSS conditionnées par la taille de l'écran
- Utilise des « points de rupture » (breakpoints) qui représentent le moment où l'affichage doit changer (en fonction de la largeur/hauteur de la fenêtre)

Responsivité : désigne l'adaptabilité de la page sur les différents supports

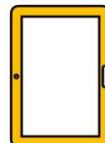
HTML-CSS-JS

Media Queries

Points de ruptures :



- Téléphone : règle $< 768\text{px}$



- Tablette en portrait : règle $< 1024\text{px}$



- Ordinateur de bureau : Règles $> 1024\text{px}$

Semifir

Exemples de points de ruptures classiques

HTML-CSS-JS

Media Queries

Breakpoints de Bootstrap (v5.1) :

Breakpoint	Class infix	Dimensions
X-Small	<i>None</i>	<576px
Small	<code>sm</code>	≥576px
Medium	<code>md</code>	≥768px
Large	<code>lg</code>	≥992px
Extra large	<code>xl</code>	≥1200px
Extra extra large	<code>xxl</code>	≥1400px

Semifir

Le site officiel : <https://getbootstrap.com/>

HTML-CSS-JS

Media Queries

Deux méthodes pour appliquer des media queries :

- Charger différentes règles dans une feuille CSS
 - On insère le code CSS dans block @media, qui sera valable que si la condition est remplie :

```
@media type (caractéristique) {  
    /*Sélecteur 1*/  
    /*Sélecteur 2*/  
    /*...*/  
}
```

- Charger une feuille CSS différente
 - On ajoute la règle dans le <head> du site, et on pointera vers le bon fichier :

```
<link rel="stylesheet" media="screen and (max-width: 768px)" href="telephone.css" />
```

Quelques exemples de media queries :

<https://media-queries.aliasdmc.fr/media-queries-live.php>

HTML-CSS-JS

Media Queries

- Syntaxe pour créer une media querie
- Les types de media (all, print, screen ou speech) sont précisés directement après le « @media »
- Les paramètres de caractéristique de média sont précisés entre parenthèses

```
@media type (caractéristique) {  
    /* Sélecteur 1 */  
    /* Sélecteur 2 */  
    /* ... */  
}
```

Les autres types (tty, tv, projection ...) sont dépréciés depuis Media Queries 4 et ne doivent plus être utilisés

HTML-CSS-JS

Media Queries

Quelques caractéristiques média :

`@media (width) /*CSS*/`

Largeur de la zone

`@media (height) /*CSS*/`

Hauteur de la zone

`@media (orientation) /*CSS*/`

Orientation (portrait ou paysage)

`@media (resolution) /*CSS*/`

Densité de pixels de l'appareil

`@media (aspect-ratio) /*CSS*/`

Le rapport largeur/hauteur

Doc :

https://developer.mozilla.org/fr/docs/Web/CSS/Media_Queries/Using_media_queries

HTML-CSS-JS

Media Queries

Quelques caractéristiques média :

Il est possible d'utiliser des opérateurs logiques :

and

Permet d'enchaîner de multiples critères les uns à la suite des autres.

not

Opérateur de négation à placer devant une query.

only

Empêche l'exécution du style par les navigateurs non compatibles avec la règle.

,

Permet de lister plusieurs conditions. Appliquera la query si au moins une des conditions est remplie.

Doc :

https://developer.mozilla.org/fr/docs/Web/CSS/Media_Queries/Using_media_queries

III. CSS 3

Mise en forme du texte

CSS



HTML-CSS-JS

Mise en forme

Changer la taille du texte :

- La taille absolue :
 - Est indiquée en pixels « px »
 - Définit la taille du texte en pixels
- La taille relative :
 - Peut être indiquée en « em* »
 - Définit la taille en multiple d'elle-même
 - Est basé sur la taille par défaut
 - Peut être indiquée avec des mots
 - (xx-small, medium, large ...)

`font-size: 14px;`

`font-size: 3.5em;`

`font-size: large;`

**On peut utiliser « ex » pour exprimer la taille relative en pourcentage.*

HTML-CSS-JS

Mise en forme

Changer la police :

- La police d'écriture peut être changée avec le paramètre :

```
font-family: Arial;
```

- Par souci préventif, on va saisir plusieurs polices pour être sur qu'il n'y ait pas de problème lors de l'affichage :

```
font-family: police1, police2, police3, serif;
```

- Le navigateur essaie de charger les polices les unes après les autres jusqu'à trouver une police qu'il possède.
- Serif est une police par défaut

NB : si le nom de la police contient des espaces, pensez à l'entourer de guillemets.

HTML-CSS-JS

Mise en forme

Les polices personnalisées :

- On peut déclarer l'utilisation d'une police personnalisée comme suit :

```
@font-face {  
    font-family: "font";  
    src: url("font.ttf");  
}
```

- Le chemin de la police doit être indiqué dans le paramètre *src: url()*

Semifir

Les polices supportées par défaut :

https://www.w3schools.com/cssref/css_websafe_fonts.asp

HTML-CSS-JS

Mise en forme

Les polices personnalisées :

- Certains navigateurs ne prennent pas en charge certains formats de fichiers (exemple : <https://caniuse.com/?search=eot>).
- Par sécurité, il est préférable d'ajouter plusieurs formats pour garantir une meilleure prise en charge des navigateurs :

```
@font-face {  
    font-family: "font";  
    src: url("font.eot");  
    url("font.woff");  
    url("font.truetype");  
    url("font.svg");  
}
```

*N'hésitez pas à vérifier la comptabilité de votre format de police avec
www.caniuse.com !*

HTML-CSS-JS

Mise en forme

Gras :

```
font-style: bold;
```

Souligné :

```
font-style: underline;
```

Ligne au dessus :

```
font-style: overline;
```

Mise en forme :

Italique :

```
font-style: italic;
```

Barré:

```
font-style: line-through;
```

Normal (défaut) :

```
font-style: none;
```

HTML-CSS-JS

Mise en forme

Alignement :

Aligné à gauche :

`text-align: left;`

Aligné à droite :

`text-align: right;`

Centré :

`text-align: center;`

Justifié

`text-align: justify;`

*NB : l'alignement ne fonctionne que sur les balises de type
« block » !*

HTML-CSS-JS

Mise en forme

Coloriser le texte:

- Le paramètre « color » est utilisé pour colorisé le texte.
- On lui donne en paramètre la valeur hexadécimale d'une couleur
- On peut récupérer cette valeur sur des sites comme <https://www.webfx.com/web-design/color-picker/> ou grâce à certains modules de VS Code

```
p {  
    color: #f00000  
}
```

Il est possible d'utiliser directement le nom de la couleur en anglais pour les couleurs « classiques »

HTML-CSS-JS

Mise en forme

Coloriser le fond :

- Le paramètre « background-color » est utilisé pour colorisé le texte.
- Comme pour « color », on lui donne en paramètre la valeur hexadécimale d'une couleur

```
p {  
    background-color: #f00000  
}
```

NB : les balises enfants héritent de la couleur de fond du parent !

HTML-CSS-JS

Mise en forme

Mettre une image de fond :

- On utilise le paramètre « background-image »
- On indique l'emplacement du fichier dans « url() »

```
p {  
    background-image: url("neige.png");  
}
```

NB : les balises enfants héritent de l'image de fond du parent !

HTML-CSS-JS

Mise en forme

Mettre une image de fond – les options :

background-repeat: no-repeat;

Permet de définir la répétition de l'image. On peut répéter l'axe « x », « y », la totalité de l'image ...

background-attachment: fixed;

On peut déterminer si l'image suit le scroll, reste à sa place ...

background-position: 0px;

Détermine la position initiale de l'image

`background: url("semifir.png") fixed no-repeat top right;`

Permet de combiner les options dans une même balise.

Les propriétés sont combinables : on utilise alors la balise « background

HTML-CSS-JS

Mise en forme

Les bordures :

- La propriété « border » permet de gérer les bordures des balises.
- Elle permet de combiner les propriétés en une seule balise
- On peut y gérer :
 - La largeur (épaisseur),
 - Le type (trait, pointillés ...),
 - La position,
 - Etc.

```
div {  
    border: red solid  
}
```

Semifir

HTML-CSS-JS

Mise en forme

Bordures – les options :

`border: 2px;`

Permet de régler l'épaisseur de la bordure

`border: solid;`

Permet de définir le type de bordure.

`border: red;`

Définit la couleur de la bordure

`border: red solid`

Les paramètres sont indiqués les uns à la suite des autres.

Liste des types de bordure :

<https://developer.mozilla.org/fr/docs/Web/CSS/border-style>

HTML-CSS-JS

Mise en forme

Bordures – les options :

`border-top: dashed red 3px ;`

On peut définir des options pour un côté spécifique. Fonctionne aussi avec bottom, left et right.

`border-radius: 5px;`

Définit l'importance de l'arrondi en pixels.

`border-radius: 5px 10px 15px 20px;`

On peut définir des importances différentes selon les coins. Dans l'ordre : haut-gauche, haut-droit, bas-gauche, bas-droit

Semifir

Liste des types de bordure :

<https://developer.mozilla.org/fr/docs/Web/CSS/border-style>