



Intégration HTML/CSS



Présentation :

- Nom
- Prénom
- Expérience en informatique (Expl : formation, langages maîtrisés..)
- Qu'est ce que le WEB ?



But du cours



But du cours

- Comprendre comment fonctionne un site web




But du cours

- Comprendre comment fonctionne un site web
- Savoir Intégrer et réaliser un site web

I - INTRODUCTION




1 - Histoire du WEB



1- Histoire du Web

Voici un bref historique des moments clés :



1- Histoire du Web

Voici un bref historique des moments clés :

- 1969 : création de l'ancêtre d'internet : ARPANET (Advanced Research Projects Agency Network) qui est un réseau militaire décentralisé.

Le réseau a ensuite évolué pour devenir un lieu d'échange universitaire avant de devenir progressivement grand public sous le nom d'internet.



1- Histoire du Web

Voici un bref historique des moments clés :

- 1969 : création de l'ancêtre d'internet : ARPANET (Advanced Research Projects Agency Network) qui est un réseau militaire décentralisé.

Le réseau a ensuite évolué pour devenir un lieu d'échange universitaire avant de devenir progressivement grand public sous le nom d'internet.

- 1972 : apparition des e-mails pour échanger des messages.




1- Histoire du Web

Voici un bref historique des moments clés :

- 1969 : création de l'ancêtre d'internet : ARPANET (Advanced Research Projects Agency Network) qui est un réseau militaire décentralisé.

Le réseau a ensuite évolué pour devenir un lieu d'échange universitaire avant de devenir progressivement grand public sous le nom d'internet.

- 1972 : apparition des e-mails pour échanger des messages.
- 1990 : 1er site web , créé par Tim Berners Lee.



1- Histoire du Web

Voici un bref historique des moments clés :


- 1969 : création de l'ancêtre d'internet : ARPANET (Advanced Research Projects Agency Network) qui est un réseau militaire décentralisé.

Le réseau a ensuite évolué pour devenir un lieu d'échange universitaire avant de devenir progressivement grand public sous le nom d'internet.

- 1972 : apparition des e-mails pour échanger des messages.
- 1990 : 1er site web , créé par Tim Berners Lee.
- 1991 : Apparition du web pour afficher des pages d'information.



2 - Fonctionnement d'un site Web



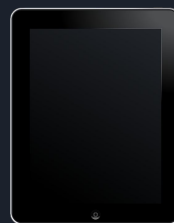
2 - Fonctionnement d'un site Web (1/4)

Surfer sur le Web, c'est un échange :

- C'est un système d'échange entre un client et un serveur

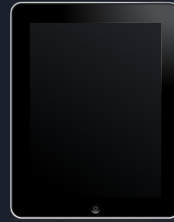
2 - Fonctionnement d'un site Web (2/4)

Le client c'est nous, depuis notre appareil (ordinateur, tablette, smartphone)



2 - Fonctionnement d'un site Web (2/4)

Le client c'est nous, depuis notre appareil (ordinateur, tablette, smartphone)



A partir de notre navigateur qui nous permet de voir le Web



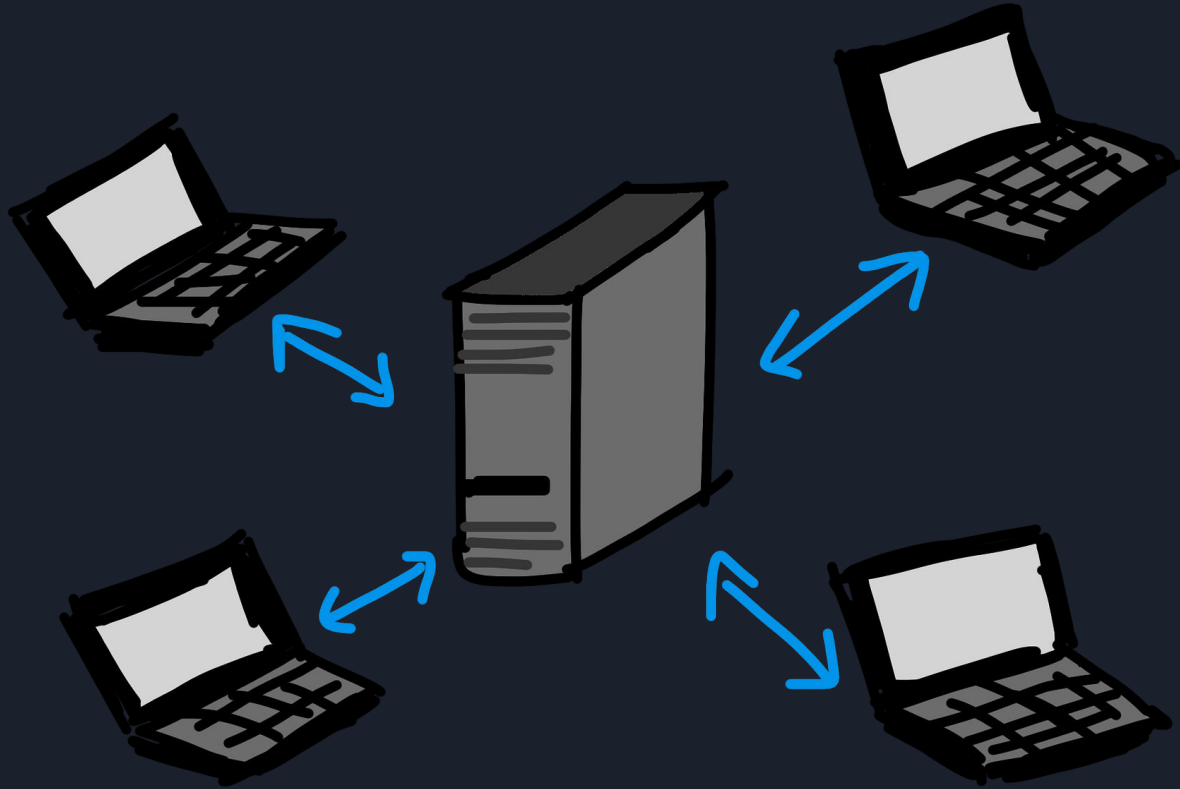
2 - Fonctionnement d'un site Web (3/4)

Le serveur c'est un ordinateur puissant qui stocke et héberge des sites Web.

C'est sur cet ordinateur que se trouvent les pages Web, c'est à dire tous les fichiers du site internet auquel on veut accéder.




2 - Fonctionnement d'un site Web (4/4)





3 - Communication entre client et serveur



3 - Communication entre client et serveur (1/4)

Le client accède à une page Web en utilisant l'adresse d'un site web (appelée URL) dans son navigateur.


Par exemple, l'URL `http://monsite.fr/mondossier/mapage.html`

Le client accède à une page Web en utilisant l'adresse d'un site web (appelée URL) dans son navigateur.

Par exemple, l'URL <http://monsite.fr/mondossier/mapage.html>

On peut décomposer l'URL en plusieurs parties :

- "http" c'est le nom du protocole de communication entre le client et le serveur
- "monsite.com" est le nom de domaine du site Web auquel on veut accéder
- "mondossier/mapage.html" est l'endroit où se trouve la page dans le site web



3 - Communication entre client et serveur (2/4)

La communication entre client et serveur se fait en trois étapes :

La communication entre client et serveur se fait en trois étapes :


1. Premièrement, le client demande une page Web au serveur. Il saisit l'URL d'un site dans son navigateur. Celui-ci envoie immédiatement une requête web au serveur.

La communication entre client et serveur se fait en trois étapes :

1. Premièrement, le client demande une page Web au serveur. Il saisit l'URL d'un site dans son navigateur. Celui-ci envoie immédiatement une requête web au serveur.
2. Ensuite, le serveur prépare cette commande c'est-à-dire la page Web en question. Le serveur va se charger de traiter la requête et renvoyer les données demandées (page web, image, vidéo...).
3. Et enfin, le navigateur interprète les données reçues et les renvoie au client qui va les afficher directement sur notre écran

La communication entre client et serveur se fait en trois étapes :

1. Premièrement, le client demande une page Web au serveur. Il saisit l'URL d'un site dans son navigateur. Celui-ci envoie immédiatement une requête web au serveur.
2. Ensuite, le serveur prépare cette commande c'est-à-dire la page Web en question. Le serveur va se charger de traiter la requête et renvoyer les données demandées (page web, image, vidéo...).
3. Et enfin, le navigateur interprète les données reçues et les renvoie au client qui va les afficher directement sur notre écran



4 - De quoi est composé un site web ?



4- De quoi est composé un site web ? (1/2)

Comme vous l'avez compris, la visite d'un site Web est le résultat d'une communication entre un client et un serveur.



4- De quoi est composé un site web ? (1/2)

Comme vous l'avez compris, la visite d'un site Web est le résultat d'une communication entre un client et un serveur.

Ce que le client, notre navigateur Web, recevra, sera écrit en langage client, c'est la partie visible de l'Iceberg souvent appelée "Front-End".



4- De quoi est composé un site web ? (1/2)

Comme vous l'avez compris, la visite d'un site Web est le résultat d'une communication entre un client et un serveur.

Ce que le client, notre navigateur Web, recevra, sera écrit en langage client, c'est la partie visible de l'Iceberg souvent appelée "Front-End".

Par contre, tout le travail que effectuera le serveur sur nos pages Web avant de les envoyer au client sera écrit en langage serveur c'est la partie cachée de l'Iceberg souvent appelée "Back-End".



4- De quoi est composé un site web ? (2/2)


Les langages principaux d'un site web :

Front-End

- HTML
- CSS
- Javascript

Back-End

- PHP
- Java
- SQL



5- Les différentes manières de concevoir un site web



5- Les différentes manières de concevoir un site web

Il existe de multiples façons de concevoir un site web :

Il existe de multiples façons de concevoir un site web :

- Premièrement, en From-scrach (de zéro). C'est-à-dire par le code pur et simple. Il existe aussi des Framework (Symfony (PHP), Angular (Js), permettant de concevoir des sites ou des applications, sans avoir à réinventer la roue ou à utiliser des CMS.

Ces derniers proposent une architecture toute prête, avec des fonctionnalités clés en main.

Il existe de multiples façons de concevoir un site web :

- Premièrement, en From-scrach (de zéro). C'est-à-dire par le code pur et simple. Il existe aussi des Framework (Symfony (PHP), Angular (Js), permettant de concevoir des sites ou des applications, sans avoir à réinventer la roue ou à utiliser des CMS.

Ces derniers proposent une architecture toute prête, avec des fonctionnalités clés en main.

- Deuxièmement, par le biais d'outils, plus communément appelés CMS. Depuis l'explosion du web dans les années 90, la technique et les langages n'ont cessés d'évoluer. Il était nécessaire de penser à automatiser et simplifier certaines actions, tout en interfaçant de plus en plus et de rendre plus accessible un site quelque qu'il soit. Je recommande d'ailleurs des CMS comme Wordpress, Prestashop, ...



6- Les différents types de sites Web

Il existe deux types de sites Web :

Les sites Statiques

Le rôle du serveur est simple : le client envoie une requête au serveur qui se contente de renvoyer la page demandée. Il ne fait aucun travail sur la page en question, d'où le terme statique.

Utilité des sites statiques : Présenter des informations. (Les sites vitrines)

Les sites Dynamiques


Le rôle du serveur est plus complexe : lorsque le client commande une page au serveur, le serveur prépare cette commande. Il fait un travail dessus avant de la renvoyer au client. C'est grâce à ça que les pages web sont personnalisées en fonction de chaque client.

Par exemple lorsque vous ou une autre personne vous connecterez à Facebook, ce qu'affichera votre fil d'actualité sera différent.

II - HTML




1- Qu'est ce que
l'HTML ?



1- Qu'est ce que l'HTML ? (½)


- HTML (Hyper Text Markup Language) est le langage permettant d'écrire un document Web.
- C'est avant tout un fichier texte pouvant contenir des balises de type HTML.
- L'HTML représente le squelette de la page Web.




1- Qu'est ce que l'HTML ? (2/2)

L'évolution d'HTML :

- La version actuelle d'HTML est HTML5
- De nombreux éléments ont été introduits dans HTML5. Parmi les plus importants figurent les éléments suivants : time, audio, description, embed, fig, shape, footer, article, canvas, navy, output, section, source, track, video, etc.

A decorative graphic on the left side of the slide consisting of two overlapping parallelograms. The front one is blue and the back one is light green. They are positioned diagonally, with the blue one partially covering the green one.

2- La structure d'un document HTML



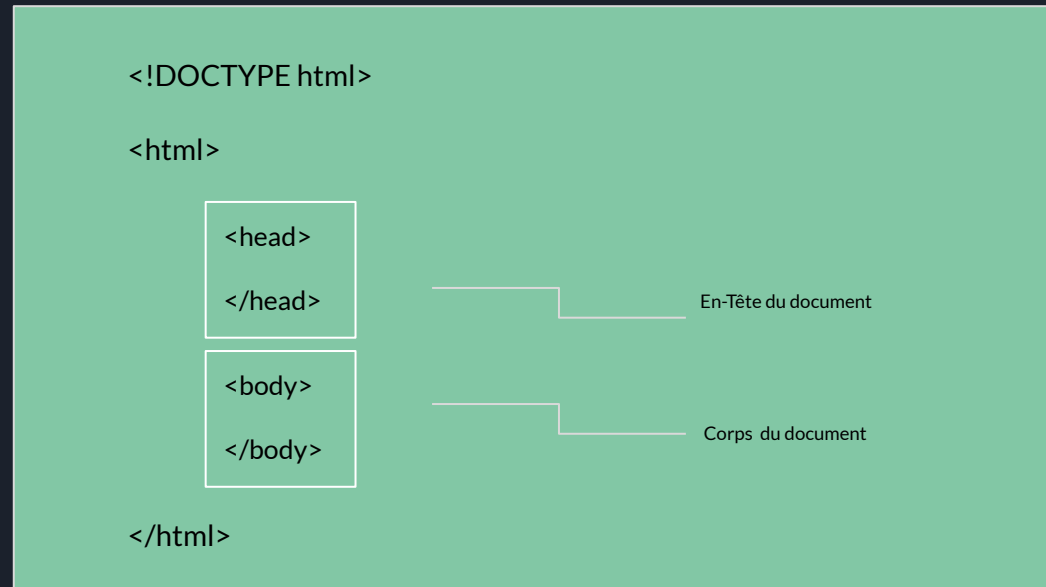
2- La structure d'un document HTML (1/2)

Un document HTML est composé de balises HTML :

- Dans un 1er temps, la balise DOCTYPE, la déclaration du type de document.
- Ensuite la balise HTML, qui contiendra l'ensemble du document.
 - Dans cette balise, nous écrivons le HEAD, l'en-tête du document.
 - Enfin, nous écrivons le BODY, le corps du document.
 - C'est dans le BODY que nous allons écrire la partie visible de la page web


2- La structure d'un document HTML (2/2)

Aperçu d'un document HTML :





3- Le HEAD



3- Le HEAD (½)

Le HEAD, ou l'en-tête du document contient les informations générales de la page, les balises de référencement ainsi que les appels CSS et Javascript.

3- Le HEAD (2/2)

Aperçu du HEAD d'un document HTML :

```
<head>
```

```
<meta charset="utf-8">
```

← Encodage des caractères

```
<title> Titre de la page </title>
```

← Titre de la page

```
<meta name="description" content="Ma description "
```

← Description de la page

```
<meta name="keywords" content="Mot clé "
```

← Mots clés de la page

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

← Appel du CSS de la page

```
<script src="script.js" type="text/javascript"
```

← Appel du JS de la page

```
</head>
```



4- Le BODY



4- Le BODY (1/4)

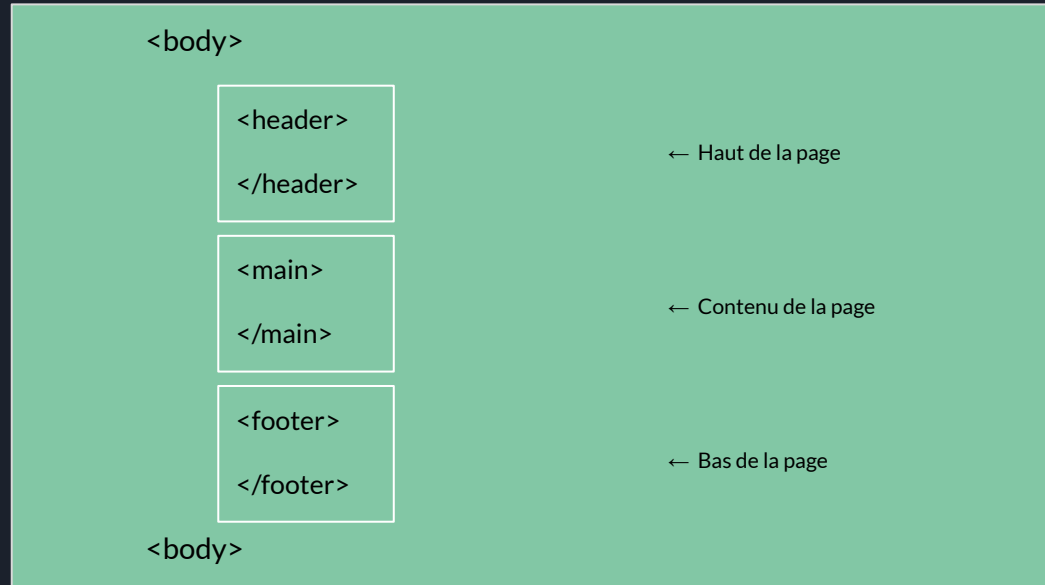
Le BODY, ou corps du document contient la partie visible de la page web.

Dans le body, nous écrivons :

- Dans un 1er temps le HEADER, le haut de la page.
- Ensuite, nous écrivons le MAIN, le contenu de la page.
- Enfin, nous écrivons le FOOTER, le bas de la page.

4- Le BODY (2/4)

Aperçu du BODY d'un document HTML :





4.1- Le HEADER

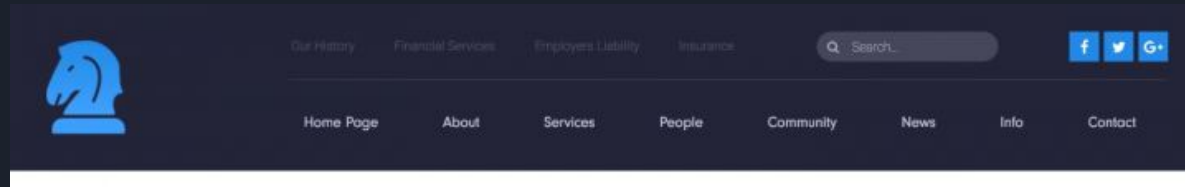


4.1- Le HEADER (1/2)

- Le HEADER correspond au haut de page.
- Le HEADER doit contenir les informations principales du site :
 - Le titre du site, ou logo.
 - Le menu principal

4.1- Le HEADER (2/2)

Exemple de header :





4.2- Le MAIN



4.2- Le MAIN

- Le MAIN correspond au contenu de la page.
- Le MAIN doit contenir les sections de présentation du site. (section, article)



4.3- Le FOOTER

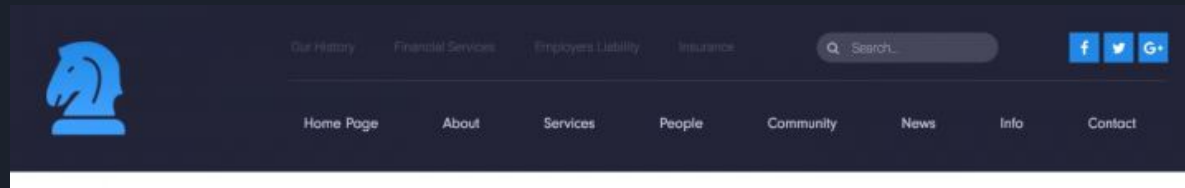


4.3- Le FOOTER (½)

- Le FOOTER correspond au bas de la page.
- Le FOOTER doit contenir un rappel des information principales, des liens internes et des liens externes.

4.3- Le FOOTER (2/2)

Exemple de FOOTER :






5- Les principaux tags



5- Les principaux tags

Certains tags ont une balise ouvrante et une balise fermante, d'autres non.

`<tag>` `</tag>`



5- Les principaux tags

Certains tags ont une balise ouvrante et une balise fermante, d'autres non.

`<tag>` `</tag>`

Les tags possèdent des attributs.

Il existe des attributs universels à tous les tags, d'autres spécifiques à quelques tags.

Exemple d'attributs universels :

- L'attribut `style` : permet d'attribuer du css (style) à l'élément auquel on l'ajoute.
- L'attribut `id` : permet d'attribuer un ID à l'élément auquel on l'applique.
- l'attribut `class` : permet d'attribuer une classe à l'élément auquel on l'applique.

PARAGRAPHE

- Le tag `<p>` :
 - Possède un tag d'ouverture ou de fermeture `<p>...</p>`
 - Sert à écrire des paragraphes.

TITRE

- Les tags titre `<h1>`, `<h2>`, `<h3>`, `<h4>`, `<h5>`, `<h6>` :
 - Possèdent un tag d'ouverture ou de fermeture `<h1>...</h1>`
 - Sert à écrire des titres.
 - Les titres les plus importants d'une page s'écrivent entre une balise `<h1>`
 - Les titres les moins importants d'une page s'écrivent entre une balise `<h6>`

COMMENTAIRE

- Les tags commentaire :

- Possède un seul tag

`<!-- Mon commentaire -->`

LINE BREAK

- Le tag break line `
` :
 - Possède un seul tag `
`
 - Sert à sauter une ligne.

LIEN

- Le tag lien `<a>` :
 - Possède un tag d'ouverture et de fermeture : `<a> ... `
 - Le tag lien doit obligatoirement posséder un attribut href :
` Voici mon lien `
 - Sert à mettre des liens vers une page web ou une partie d'une page web.

LIEN

- Le tag lien `<a>` possède plusieurs attributs :

Attribut	Valeur possible
href	lien vers lequel on veut renvoyer
target	self / blank / parent / top

IMAGE

- Le tag image `` :
 - Possède un seul tag : ``
 - Le tag image doit obligatoirement posséder un attribut `src` :
``
 - Sert à mettre des images dans notre page Web..

IMAGE

- Le tag image `` possède plusieurs attributs :

Attribut	Valeur possible
src	chemin ou lien vers l'image à afficher
alt	text d'alternative dans le cas où l'image ne s'affiche pas

IMAGE

- La source d'une image peut être de deux types :


Type	Chemin
Relatif	Depuis l'emplacement de notre fichier
Absolu	Lien sur internet

DIVISION

- Le tag division `<div>` :
 - Possède un tag d'ouverture et de fermeture : `<div>...</div>`
 - Sert à diviser notre page Web en plusieurs parties.
 - L'élément HTML `<div>` (ou division) est le conteneur générique du contenu du flux. Il n'a aucun effet sur le contenu ou la mise en page tant qu'il n'est pas mis en forme d'une manière quelconque à l'aide de CSS.



6- Les formulaires



6- Les formulaires

Les formulaires en HTML permettent une interaction de l'internaute :

Ils permettent à l'internaute de renseigner des informations et de les stocker.



6- Les formulaires

Aperçu d'un formulaire :

```
<form>
```

```
  <label for="monid"> Entrez le nom </label>
```

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

```
  <label for="monid"> Entrez le nom </label>
```

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

```
</form>
```



6.1 - Les inputs

6.1 - Les inputs

<input type="text" value="Texte ..."/>	← <code><input type="text"></code>
<input type="number" value="12345"/>	← <code><input type="number"></code>
<input type="email" value="test@mail.fr"/>	← <code><input type="email"></code>
<input type="password" value="ooooo"/>	← <code><input type="password"></code>
<input checked="" type="checkbox"/> choix 1	<code><input type="checkbox"></code>
<input checked="" type="checkbox"/> choix 2	<code><input type="checkbox"></code>
<input checked="" type="radio"/> choix 1	<code><input type="radio" name="choix"></code>
<input type="radio"/> choix 2	<code><input type="radio" name="choix"></code>
<input type="submit" value="Envoyer"/>	← <code><input type="submit"></code>

Les autres inputs :

button	Bouton
color	Permet de définir une couleur
date	Permet de définir une date
file	Permet de sélectionner un fichier
tel	Permet de saisir un numéro de tel
url	permet de saisir une URL
reset	bouton qui réinitialise le contenu du formulaire
search	champ texte pour des termes de recherche

Les attributs d'un inputs :

Attribut	Valeur
type	type de l'input
name	nom de l'input
id	'ID de l'input
placeholder	texte par défaut dans l'input
value	valeur du champs
required	Pas de valeur
minlength	taille minimale
maxlength	taille maximale



6.2 - Textarea



6.2 - Les Textarea

- Le tag textarea permet à l'utilisateur de saisir un texte. La longueur du texte pourra être spécifiée grâce à des attributs
- Le tag textarea possède une balise ouvrante et fermante : `<textarea>` `</textarea>`

Attribut	Valeurs possibles
cols	Nombres de colonnes
rows	Nombre de lignes
name	nom du textarea
ID	ID du textarea



6.3 - Select

- Le tag select possède une balise ouvrante et fermante : `<select>` `</select>`

Attribut	Valeurs possibles
name	nom du select
ID	ID du select

Aperçu d'un select:

```
<select>
```

```
  <option>Pizza 1</option>  
  <option>Pizza 1</option>  
  <option>Pizza 1</option>  
  <option>Pizza 1</option>  
  <option>Pizza 1</option>  
  <option>Pizza 1</option>  
  <option>Pizza 1</option>  
  <option>Pizza 1</option>
```

```
</select>
```



7 - Les tableaux



7 - Les tableaux

- On peut créer un tableau en HTML grâce à la balise `<table>`
- Cette dernière contient une balise ouvrante et fermante `<table>` `</table>`
- Le contenu du tableau se trouve entre la balise ouvrante et fermante `<table>`
 - Une ligne se trouve entre deux balises `<tr>` ouvrante et fermante : `<tr>` `</tr>`
 - Les cellules d'une ligne (donc colonnes) se trouvent entre deux balises `<td>` ouvrante et fermante : `<td>` `</td>`

7 - Les tableaux

```
<table>
```

```
  <tr>
```

```
    <td> 1e case de ma 1e ligne </td>
```

```
    <td> 2e case de ma 1e ligne </td>
```

```
    <td> 3e case de ma 1e ligne </td>
```

```
    <td> 4e case de ma 1e ligne </td>
```

```
  </tr>
```

```
  <tr>
```

```
    <td> 1e case de ma 2e ligne </td>
```

```
    <td> 2e case de ma 2e ligne </td>
```

```
    <td> 3e case de ma 2e ligne </td>
```


```
    <td> 4e case de ma 2e ligne </td>
```

```
  </tr>
```

```
</table>
```

Résultat du code précédent :

1e case de ma 1e ligne	2e case de ma 1e ligne	3e case de ma 1e ligne	4e case de ma 1e ligne
1e case de ma 2e ligne	2e case de ma 2e ligne	3e case de ma 2e ligne	4e case de ma 2e ligne



7 - Les tableaux

- On peut aussi fusionner deux cellules ou deux lignes grâce à des attributs.

7 - Les tableaux

- On peut aussi fusionner deux cellules ou deux lignes grâce à des attributs.
- Fusion de deux colonnes :

```
<table>
  <tr>
    <td colspan="3" > Fusion des 3 cases </td>
    <td> 1-4 </td>
  </tr>
  <tr>
    <td> 2-1 </td>
    <td>2-2 </td>
    <td> 2-3 </td>
    <td> 2-4 </td>
  </tr>
</table>
```


7 - Les tableaux

- On peut aussi fusionner deux cellules ou deux lignes grâce à des attributs.
- Fusion de deux colonnes :

```
<table>  
  <tr>  
    <td colspan="3" > Fusion des 3 cases </td>  
    <td> 1-4 </td>  
  </tr>  
  <tr>  
    <td> 2-1 </td>  
    <td>2-2 </td>  
    <td> 2-3 </td>  
    <td> 2-4 </td>  
  </tr>  
</table>
```

Résultat du code :

Fusion des 3 cases			1-4
2-1	2-2	2-3	2-4

7 - Les tableaux

- On peut aussi fusionner deux cellules ou deux lignes grâce à des attributs.
- Fusion de deux lignes :

```
<table>
  <tr>
    <td rowspan = "2"> 1-1 et 2-1 </td>
    <td> 1-2 </td>
    <td> 1-3 </td>
  </tr>
  <tr>
    <td> 2-2 </td>
    <td> 2-3 </td>
  </tr>
</table>
```

7 - Les tableaux

- On peut aussi fusionner deux cellules ou deux lignes grâce à des attributs.
- Fusion de deux colonnes :

```
<table>
  <tr>
    <td rowspan = "2"> 1-1 et 2-1 </td>
    <td> 1-2 </td>
    <td> 1-3 </td>
  </tr>
  <tr>
    <td> 2-2 </td>
    <td> 2-3 </td>
  </tr>
</table>
```


Résultat du code :

1-1 et 2-1	1-2	1-3
	2-2	2-3

III - CSS



1- Qu'est ce que le CSS ?




1- Qu'est ce que le CSS ?

- CSS (Cascading Style Sheets) est le langage permettant de mettre en forme des pages web.
- C'est grâce au CSS qu'on peut appliquer le style et le design qu'on souhaite à n'importe quel page WEB.



2- Où écrire notre CSS ?




2- Où écrire notre CSS ?

On peut écrire notre CSS dans 3 endroits différents : (1/3)

- Dans le head de notre HTML :

```
<!DOCTYPE html>
<html>
  <head>
    <style> J'écris mon CSS </style>
  </head>
  <body>
    ...
  </body>
</html>
```





2- Où écrire notre CSS ?

On peut écrire notre CSS dans 3 endroits différents : (2/3)

- En attribut d'un élément dans notre HTML :

```
<!DOCTYPE html>
<html>
  <head>
    ...
  </head>
  <body>
    <p style="j'écris mon CSS"> Texte </p>
  </body>
</html>
```



2- Où écrire notre CSS ?

On peut écrire notre CSS dans 3 endroits différents : (3/3)

- Dans un fichier CSS, il faut aussi lier ce fichier dans le head du fichier HTML auquel on veut l'appliquer :

```
<!DOCTYPE html>
<html>
  <head>
    <link rel="stylesheet" type="text/css" href="lien relatif vers notre fichier CSS">
  </head>
  <body>
    ...
  </body>
</html>
```



2- Syntaxe du CSS



2 - Syntaxe du CSS

Voici la syntaxe du CSS :


```
selecteur {  
  
    propriété1 : valeur1 ;  
  
    propriété2 : valeur2 ;  
  
}  
  
/* Ceci est un commentaire */
```



3- Comment écrire notre CSS ?



3.1- Sélectionner l' élément



3.1- Sélectionner l'élément

Pour sélectionner un tag :


```
tag { propriété : valeur ; }
```

Pour sélectionner une classe :

```
.class { propriété : valeur ; }
```

Pour sélectionner un ID :

```
#id { propriété : valeur ; }
```



3.1- Sélectionner l'élément

Pour sélectionner plusieurs éléments :


```
a, b { propriété : valeur ; }
```

Pour sélectionner un élément qui est dans un élément (b qui est dans a) :


```
a b { propriété : valeur ; }
```

Pour appliquer à tous les éléments :

```
* { propriété : valeur ; }
```





3.2- Les différentes propriétés et valeurs



3.2- Les différentes propriétés et valeurs :

- Les propriétés de couleur :
 - ★ Codage RGB
 - ★ Hexadécimale
 - ★ Codage RGBA (identique que RGB avec le A pour l'opacité)




3.2- Les différentes propriétés et valeurs :

- Les unités de mesure :

- ★ Pixels (px)

- ★ Em (em)

- ★ Pourcentage (%)



3.2- Les différentes propriétés et valeurs :


- Les polices d'écritures : font-family (1/4)

```
p { font-family : police ; }
```

★ Polices installées sur l'ordinateur

★ Polices intégrées au CSS

- serif
- sans-serif
- cursive




3.2- Les différentes propriétés et valeurs :

- Les polices d'écritures : (2/4)

- ★ Polices importées depuis un lien sur internet

- Aller sur un site de police (Exemple google font)
- Copier le lien et le mettre dans le head du HTML




3.2- Les différentes propriétés et valeurs :

- Les polices d'écritures : (3/4)

★ Polices importées depuis un fichier

- Télécharger le fichier depuis internet (Exemple Pixelify)
 - Fichier True Type
 - Fichier Open Type
- Utiliser un générateur de fichier font family (Exemple font squirell)
- Télécharger le dossier
- Créer un dossier font dans le dossier du projet
- Copier le contenu du dossier téléchargé et le coller dans le dossier font
- Copier le contenu du fichier steelsheet.css dans notre CSS



3.2- Les différentes propriétés et valeurs :

- Les polices d'écritures : (4/4)

★ Système de rattrapage :

si un police n'est pas disponible, celle qui vient après est prise en compte.

Exemple :

```
p { font-family : verdana, serif, sans-serif ; }
```

3.2- Les différentes propriétés et valeurs :


- Les propriété arrière-plan (background) (½) :

- Le background peut être une couleur :

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

- Le background peut aussi être une image :

```
p { background : URL(lien vers une image) ; }  
p { background-image : URL(lien vers une image) ; }
```

3.2- Les différentes propriétés et valeurs :

- Les propriété arrière-plan (background) (2/2):
 - Le background peut être un dégradé de couleurs :

```
p { background: radial-gradient(couleur1, couleur2);}
```

- Le background ne se répète pas :


```
p { background : no-repeat URL (lien vers une image) ; }
```

3.2- Les différentes propriétés et valeurs :

- Les propriété du texte :
 - S'appliquent à tous les éléments html contenant du texte (p, h1...) :
 - weight : épaisseur
 - color : couleur du texte
 - align : alignement

Exemple :

```
p {  color : red ;  
      weight : bold ;  
      align : center ;  
}
```




3.2- Les différentes propriétés et valeurs :

- Les propriété de taille :


- height : longueur
- width : largeur

Exemple :

```
div {  
    height : 80%;  
    width : 30%;  
}
```




3.3- Les pseudos-classes



3.3- Les pseudos-classes :

- La syntaxe des pseudos-classes :

```
sélecteur : pseudo-classe {  
    propriété : valeur ;  
}
```



3.3- Les pseudos-classes :

- Les différentes pseudos-classes :
 - a. `hover` : lorsque l'élément survolé
 - b. `link` : lorsque le lien n'a pas été visité
 - c. `visited` : lorsque le lien a déjà été visité
 - d. `active` : lorsqu'un élément est activé par l'utilisateur (le moment entre l'appui et le relâchement)



3.4- La mise en page



3.4- La mise en page :

La propriété display :

La propriété display permet de gérer le comportement d'affichage d'un élément.

Il existe plusieurs valeurs possibles pour la propriété Display.

3.4- La mise en page :

```
div { display : block }
```

Le display : block permet à l'élément de prendre toute la largeur disponible.

Exemple:

<DIV>

<DIV>

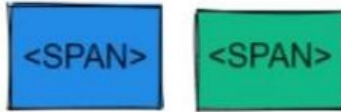
3.4- La mise en page :

```
div { display : inline }
```

Le `display : inline` permet à l'élément de prendre que l'espace disponible pour le contenu

La hauteur et la largeur ne s'applique pas ici.

Exemple:



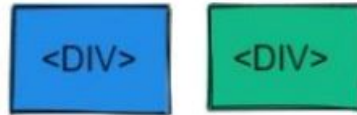
3.4- La mise en page :

```
div { display : inline-block }
```

Le `display : inline-block` permet de mettre les éléments en ligne mais en bloc

La hauteur et la largeur s'appliquent ici.

Exemple:





3.4- La mise en page :

```
div { display : none}
```

Le display : none , l'élément ne s'affiche pas.


3.4- La mise en page :

```
div { display : flex}
```

Le display : flex : permet de mettre les éléments les uns à côté des autres sur un plan horizontal.

Exemple:






3.4- La mise en page :

```
div { display : flex;  
      flex-direction: row ;  
}
```

Le `flex-direction` : permet d'agencer les éléments dans le sens qu'on veut.

- `row, column`
- `row-reverse`
- `column-reverse`



3.4- La mise en page :

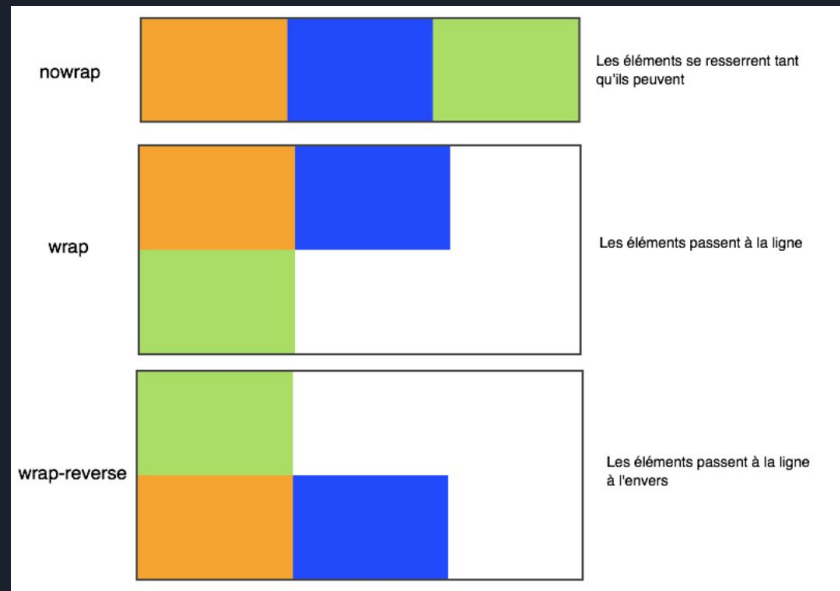
```
div { display : flex;  
      flex-wrap : no wrap ;  
}
```

Le flex-direction : permet de gérer le retour à la ligne des éléments du conteneur :

- nowrap : pas de retour à la ligne
- wrap : retour à la ligne lorsqu'il n'y a plus de place
- wrap-reverse : retour à la ligne lorsqu'il n'y a plus de place en sens inverse

3.4- La mise en page :

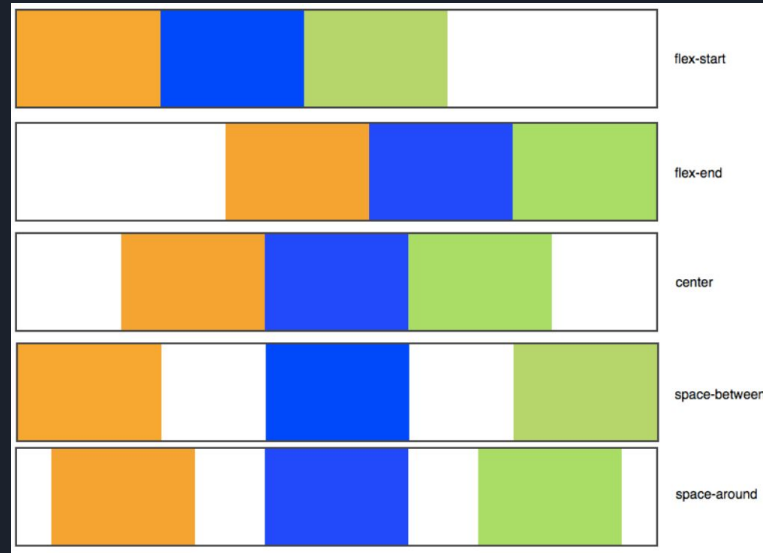
```
div { display : flex;  
      flex-wrap : no wrap ;  
}
```



3.4- La mise en page :

```
div { display : flex;  
      justify-content : ... ;  
}
```

Le justify content permet d'aligner les élément contenus dans le conteneur auquel on a appliqué display : flex.





3.5- La propriété margin

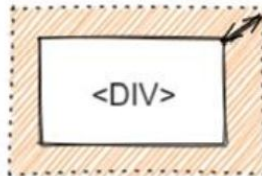
3.5- La propriété margin :

```
div { margin : 15px; }
```

La propriété margin permet de gérer la marge extérieure d'un élément.

On peut exprimer en PX ou en %.

Exemple:



(margin: 15px)



3.5- La propriété padding

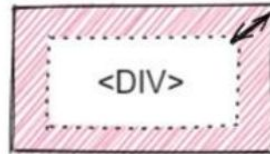
3.5- La propriété padding:

```
div { padding : 15px; }
```

La propriété padding permet de gérer la marge intérieure d'un élément.

On peut exprimer en PX ou en %.

Exemple:



(padding: 15px)



3.5- La propriété float



3.5- La propriété float:

```
div { float: left; }
```

La propriété float permet de gérer l'alignement d'un élément.

3.5- La propriété float:

```
div { float: left; }
```

La propriété float:left il est possible d'aligner de gauche à droite :

Exemple:

<DIV>

<DIV>

3.5- La propriété float:

```
div { float: right; }
```

La propriété float : right il est possible d'aligner de droite à gauche :

Exemple:





3.5- La propriété position



3.5- La propriété position :

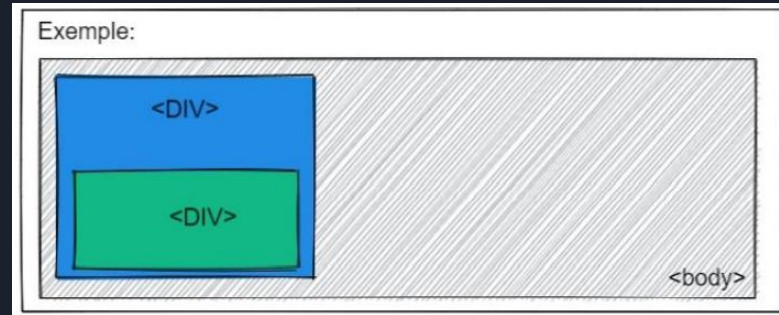
```
div { position : ... ; }
```

La propriété position permet de gérer le positionnement d'un élément.

3.5- La propriété position :

```
div { position : static ; }
```

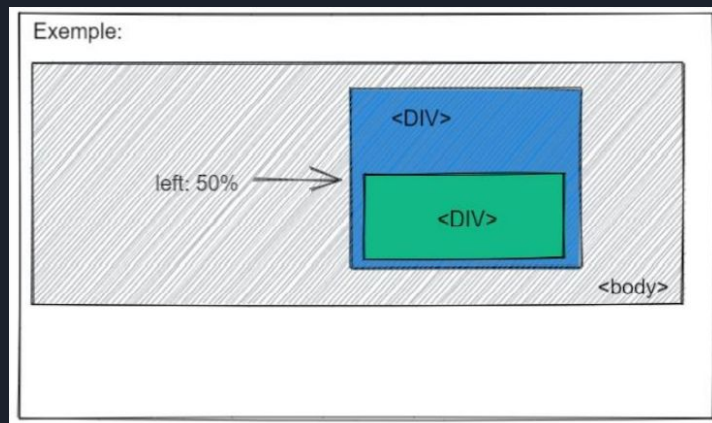
La propriété position : static est le positionnement par défaut d'un élément.



3.5- La propriété position :

```
div { position : relative;  
      left : 50%;  
}
```

Le position : relative est le positionnement de repère d'un élément.

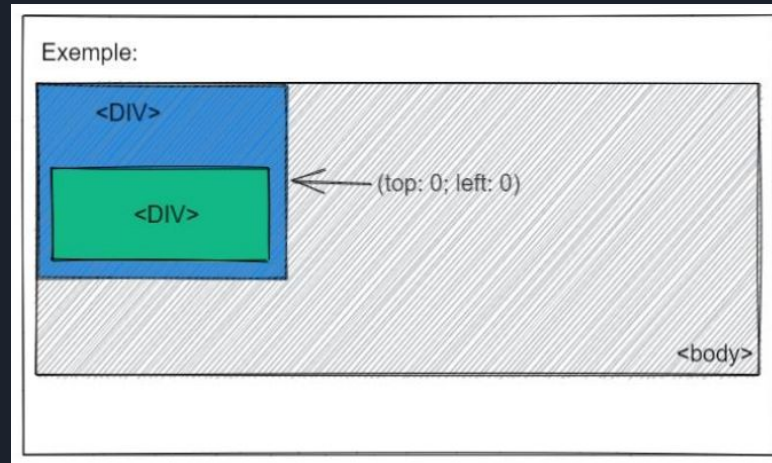


3.5- La propriété position :

```
div { position : absolute;  
      top : 0;  
      left : 0; }
```

Le position : absolute est le positionnement de superposition d'un élément sur un autre

Pour qu'un élément soit en position absolue il faut que son élément parent soit en position relative.





3.5- La propriété position :

```
div { position : absolute;  
      top : 0;  
      left : 0;  
      transform : translate( , )  
    }
```

Centrer un élément en position absolute.

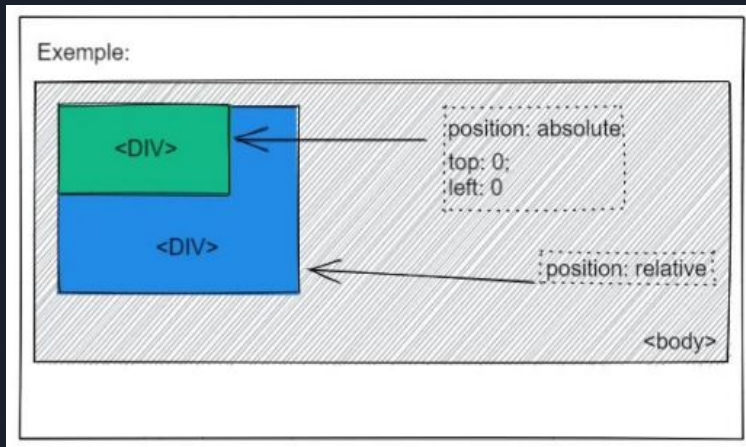
3.5- La propriété position :

Relation entre position relative et position absolue :

En réalité, un élément positionné en absolue cherchera toujours à se repérer par-rapport à son premier élément parent en relative.

Par défaut, le seul élément en relative qui existe dans la page est le BODY.

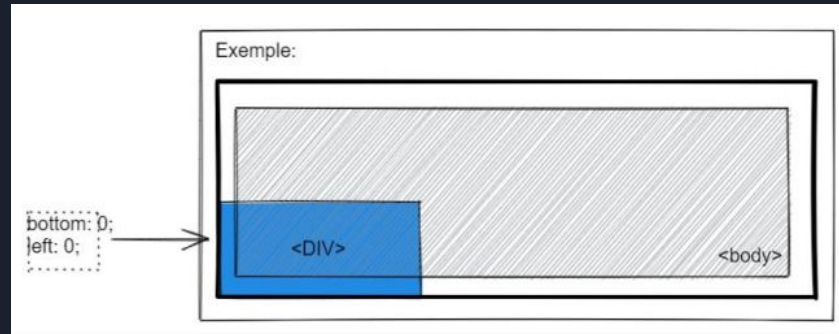
Si un autre élément est en relative et contient un élément en absolue, l'élément en absolue se repèrera non plus par-rapport au BODY mais par-rapport à l'élément parent en relative.



3.5- La propriété position :


```
div { position : fixed;  
      top : 0;  
      left : 0; }
```

Le position : fixed est le positionnement fixé par rapport à l' écran.



A decorative graphic on the left side of the slide consisting of overlapping geometric shapes. There is a blue parallelogram and a light green parallelogram, both tilted at an angle. The background is a dark navy blue with subtle diagonal lines.

4.-Le responsive design



4- Le responsive design :

Le responsive concerne le fait d'adapter son site à tous types d'écrans.

Pour que son site s'adapte à tous types d'écran, il faut dans un 1er temps, déclarer la balise meta viewport dans le HEAD du document.

Ecrivez cette balise dans le HEAD de votre document, après la balise TITLE

```
<meta name="viewport" content="viewport-fit=cover, width=device-width, initial-scale=1.0,  
minimum-scale=1.0, maximum-scale=1.0, user-scalable=no" />
```

Cela permet de déclarer votre site comme prêt pour être responsive.



4- Le responsive design :

La propriété `width` contrôle la taille de la zone d'affichage.

Elle peut être définie sur un nombre spécifique de pixels comme `width=600` ou sur la valeur spéciale `device-width`, qui est la largeur de l'écran en pixels CSS à une échelle de 100%. (Il existe des valeurs `height` et `device-height` correspondantes, qui peuvent être utiles pour les pages comportant des éléments qui changent de taille ou de position en fonction de la hauteur du viewport).

La propriété `initial-scale` contrôle le niveau de zoom lors du premier chargement de la page.

Les propriétés `maximum-scale`, `minimum-scale` et `user-scalable` contrôlent la manière dont les utilisateurs et utilisatrices sont autorisé-e-s à zoomer ou dézoomer la page.

```
<meta name="viewport" content="viewport-fit=cover, width=device-width, initial-scale=1.0, minimum-scale=1.0, maximum-scale=1.0, user-scalable=no" />
```



4.1-Les media-queries



4.1 - Les media-queries

Les media queries sont les points d'arrêts CSS des tailles d'écrans pour lesquelles on veut travailler.

Vous pouvez ainsi définir tous les comportements CSS en fonction de la taille d'écran de l'appareil utilisé par le client.

A la fin de votre fichier style.css, écrivez ceci :

```
@media (max-width: 480px) {  
    body { background: red; }  
}
```

Cela indique que le body change de couleur de fond en rouge pour tous les appareils aux tailles d'écrans inférieurs ou égales à 480 pixel