Leçon 7 : Présentation et éléments de base du CSS

UVCI
APR-2102-2-L7

DIABATE Nagbégna,

novembre 2017
1.0

Table des matières

I - Objectifs **4**

II - Section 1 : Présentation du CSS **5**

1. Qu'est ce que CSS ? A quoi sert-il ? 5

2. Exercice : Pouvons-nous maintenant passer au quiz ? 6

III - Section 2 : Éléments de base du CSS **7**

1. Éléments de base du CSS 7

2. Exercice : Entraînez-vous avec ce mini-quiz ! 8

IV - Section 3 : Les class/id et div/span **10**

1. 1. Les class/id et div/span 10

2. Exercice 12

V - Section 4 : Propriétés de mise en forme du contenu avec CSS **13**

1. 1. La mise en page des paragraphes 13

2. 2. La police de caractère ou font 13

3. 3. Les couleurs et les fonds 13

4. 4. Les listes 14

5. 5. Les bordures 14

6. Exercice 16

VI - Section 5 : les blocs et leur positionnement **17**

1. 1. Marges externes, marges internes et dimensions 17

2. 2. Positionnement flottant 20

3. 3. Positions absolues, fixes et relatives 20

4. Exercice 21

VII - Travaux Pratiques N° 10 **23**

1. Les feuilles de style (CSS) 23

VIII - Solutions des exercices **24**

Objectifs

À la fin de cette leçon, vous serez capable de :

- *savoir* ce qu'est le CSS ;
- *savoir* pourquoi utiliser une feuille de style ;
- *créer* et intégrer une feuille de style de style à une page web;
- *connaître* les éléments de base du langage CSS
- *mettre* en forme le texte des pages avec CSS ;
- *mettre* en forme les liens et listes ainsi que les tableaux ;
- *uniformiser* la mise en forme des textes dans toutes les pages ; et
- *créer* et positionner des formes avec CSS.

Section 1 : Présentation du CSS

1. Qu'est ce que CSS ? A quoi sert-il ?

Dans la construction d'un site web, l'utilisation des feuilles de styles en cascade, appelées communément *CSS* (*Cascading Style Sheets*), permet de séparer le fond de la forme, la source HTML contenant uniquement le texte et les images structurés par les balises. Si le code est bien écrit, un site doit pouvoir être consulté sans voir de feuilles de styles.

Les *CSS* associées au code source donnent toutes les informations de présentation utilisées dans les pages :

- police utilisées dans les titres, les blocs d'information, les listes couleurs de fond ou du texte
- bordures de texte
- positionnement des blocs, marges internes et externes etc.

Les feuilles de styles sont consignées dans un fichier caractérisé par l'extension *.css* et sont référencées dans l'en-tête du fichier HTML pour créer la liaison entre le fichier HTML et la feuille la CSS associée .

Dans l'exemple suivant, on se propose de lier la feuille "*style.css*" situé dans le répertoire "*css/*" de même niveau que le fichier de la page web.

```
<head>
   <meta charset="utf-8">
   <title> Titre de ma page UVCI </title>
   <link type="text/css" rel="stylesheet" href="css/style.css">
</head>
```

Les attributs HTML sont un principe qu'on retrouve en CSS : une paire de mot représente un *attribut* (un type de comportement) et sa *valeur* (le comportement exact). Ici la balise *<link>* prend trois attributs.

- *type="text/css"* : le *type*, qui permet de donner le format et le langage de données reçues.
- *rel="stylesheeet"* : l'attribut *rel* , qui indique à quoi servent les données en question.
- *href="css/style.css"* : l'attribut *href* , qui indique ou se trouvent les données. La valeur du *href* doit être le chemin (relatif) sur votre ordinateur/serveur qui mène à votre feuille de style par rapport à l'emplacement de la page en cours.

Remarque

Bien qu'il soit possible d'écrire du CSS directement dans un document *html* à l'aide de la balise *<style></style>*, cette pratique devient *désuète et n'est pas recommandée.*

Complément : NB :

Consultez la documentation sur le site : *http://simplonline.co/13-ressources/67-introduction-au-css-comment-ca-marche* pour avoir plus d'informations sur les feuilles de style afin de mieux répondre aux questions du quiz.

2. Exercice : Pouvons-nous maintenant passer au quiz ?

[Solution n°1 p 24]

Exercice

1) Que veut dire CSS ? :

- ◯ Cascading Style Sheets
- ◯ Computer Style Sheets
- ◯ Colorfull Style Sheets

Exercice

2) Quel est le code HTML correct définissant un CSS externe ? :

- ◯ `<stylesheet>mon_style</stylesheet>`
- ◯ `<link rel="stylesheet" type="text/css" href="mon_style.css">`
- ◯ `<stylesheet>mon_style.css</stylesheet>`

Exercice

3) Quelles balises HTML définissent un attribut de style interne à une page HTML? :

- ◯ <css>... </css>
- ◯ <script>... </script>
- ◯ <style>... </style>

Exercice

4) Où place-t-on les déclarations de feuilles de style ? :

- ◯ Au début du document
- ◯ A la fin du document
- ◯ Entre les balises <body>et </body>
- ◯ Entre les balises <head> et </head>

Exercice

5) Pour créer des styles CSS pour une page web, je :

- ◯ peux utiliser Microsoft Word.
- ◯ peux utiliser un éditeur de texte quelconque tel que bloc-note.
- ◯ dois utiliser Microsoft Powerpoint.

Section 2 : Éléments de base du CSS

II

1. Éléments de base du CSS

Les *sélecteurs* sont le cœur du *CSS*, ils servent comme leur nom l'indique à sélectionner un élément en particulier. Voici la forme que prend un sélecteur:

```
sélecteur{
   propriété : attribut;
}
```

- *Sélecteur* : le nom de la/les balise(s) que vous souhaitez sélectionner ;
- *Propriété* : la propriété de la balise que sur laquelle vous voulez travailler ;
- *Attribut* : ce que vous voulez faire de cette propriété.

Prenons un exemple plus parlant. Pour modifier la couleur des paragraphes il suffit donc d'écrire:

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

Ici, toutes les balises p verront la couleur de leur texte passer au rouge grâce à l'attribut color.

Chaque paire de propriété/attribut doit se terminer par un point-virgule, ce qui permet de placer une autre paire propriété/attribut. Il est donc possible de faire autant de modifications que l'on souhaite au sein d'un même sélecteur:

- Chaque couple *propriété: valeur* terminé par un <<;>> est une déclaration.
- Une feuille de style peut définir *autant de déclarations que le sélecteur peut autoriser.*
- Une *propriété* correspond à un *attribut d'élément* HTML de type bloc (*margin-top , font-size, background-color,* etc.)
- Une valeur doit être cohérente avec la propriété CSS à laquelle elle est affectée (par exemple une couleur pour background-color et pas une mesure). Dans le cas contraire, la déclaration est ignorée.

Des commentaires peuvent être insérés en tout endroit du fichier sous la forme : */* Ceci est un commentaire */*

Cependant, des fois, il peut arriver qu'on ne souhaite pas accéder à toutes les balises d'un même type, mais plutôt certaines d'entre elles.

On utilisera alors *l'imbrication des balises.* En effet, en nommant judicieusement son sélecteur on peut atteindre uniquement les éléments se trouvant à l'intérieur d'autres éléments. Il suffit de nommer le sélecteur en *commençant par le plus grand conteneur, d'ajouter un espace, puis d'ajouter le nom d'une balise se trouvant à l'intérieur.*

```
main section article p {
    color : red;
}
```

Ici, seuls les paragraphes se trouvant à l'intérieur d'un "*article*", qui eux-mêmes se trouvent dans une "*section*", qui se trouve eux-mêmes dans le "*main*".

On peut par ailleurs travailler en même temps sur plusieurs sélecteurs. Par exemple, si vous voulez

appliquer les même propriétés/valeurs à un paragraphe et à un titre, il vous suffit de les placer tous les deux, séparés d'une virgule avant l'ouverture des accolades.

2. Exercice : Entraînez-vous avec ce mini-quiz !

[Solution n°2 p 24]

Exercice

1) Pour documenter les styles, j'utilise l'écriture suivante :

- ◯ // un commentaire //
- ◯ */ un commentaire /*
- ◯ /* un commentaire */

Exercice

2) En vous basant sur les exemples du cours, qu'est-ce que "color" en CSS ?

- ◯ Un sélecteur
- ◯ Une propriété
- ◯ Une déclaration

Exercice

3) Dans l'écriture suivante,

```
h1{ font-family:tahoma; }
```

, h1 désigne :

- ◯ une règle
- ◯ une propriété
- ◯ un sélecteur
- ◯ rien du tout

Exercice

4) L'écriture suivante

```
h1,.main,.plug {
   font-family:verdana;
   font-size:18px;
}
```

permet de définir les propriétés pour :

- ◯ h1 seulement
- ◯ h1 ou .main ou .plug
- ◯ h1, .main et .plug
- ◯ rien, cette écriture est fausse

Exercice

5) Quelle est la déclaration de style correct ? :

- ◯ {p color : blue;}
- ◯ p {color : blue;}
- ◯ p : color : blue

Section 3 : Les class/id et div/span

1. 1. Les class/id et div/span

Il existe des attributs *html* que l'on peut rajouter dans n'importe quelle balise et qui facilitent l'accès via le CSS. Ce sont les attributs "*class*" et "*id*".

Ces deux attribut bien que utilisé dans les balises n'ont pas le même rôle. En effet, *on peut répéter une classe autant de fois qu'on le souhaite au sein d'un document HTML* afin de travailler sur divers élément via un même sélecteur. En revanche, *chaque ID est unique et ne doit être utilisé qu'une et une seule fois* !

```
<p class="ecole">Tadam</p>
<p class="ecole ">Tadam</p>
<p id="unique">Tadam</p>
<p class="ecole ">Tadam</p>
```

On peut nommer les class et les id comme on veut, mais il est conseillé d'être plus lucide et donner des noms plus indicatifs afin de permettre une plus facile compréhension du code. Ensuite pour y accéder en CSS, rien de plus simple: Ils se comportent comme des sélecteurs classiques, à un détail près. Les *class* doivent être précédées d'un *point* et les *id* d'un signe *dièse*.

```
. ecole {
       color : red;
font-size:18px;
font-weight:bold;
}
#unique {
     color : blue; font-style:italic; fontsize:14px; line-height:16px;
}
```

Remarque : Comme pour le sélecteur *#unique*, le retour à la ligne a la fin de fin de chaque propriété n'est pas obligatoire.

On parle aussi de *Pseudo-classe* et de *Pseudo-élément* en CSS. Une *pseudo-classe* est un mot-clé qui peut être ajouté à un sélecteur afin d'indiquer l'état spécifique dans lequel l'élément doit être pour être ciblé par la déclaration. La pseudo-classe :*hover*, par exemple, permettra d'appliquer une mise en forme spécifique lorsque l'utilisateur survole l'élément ciblé par le sélecteur.

Les pseudo-classes permettent d'appliquer un style à un élément non seulement en fonction de son contenu dans l'arbre du document mais aussi en fonction de facteurs externes (l'historique de navigation par exemple avec :*visited* ; le statut du contenu avec :*checked* ; la position de la souris : *hover*).

À la différence des pseudo-classes, les *pseudo-éléments* peuvent être utilisés afin d'appliquer un style sur une certaine partie d'un élément.

Syntaxe :

sélecteur:pseudo-classe {

propriété: valeur;

}

Exemple :

```
.survol:hover {
  background-color: palegreen;
}
```

Vous trouverez des informations plus détaillées sur le site *des développer Mozilla*. Cependant, la suite du cours présente les pseudos classes les plus usuelles applicables à CSS2.1 le CSS3 proposant encore plus d'éléments et de classes:

- *:link* : cible les liens non visités ;
- *:hover* : cible un élément pointé visuellement (survolé) grâce à une souris par exemple ;
- *:active* : cible un élément activé par l'utilisateur, par exemple au moment du clic sur le bouton gauche de la souris. Intervient entre la pression sur le bouton et son relâchement ;
- *:focus* : cible un élément pointé physiquement, grâce à la touche tab du clavier par exemple, ou après le relâchement du bouton gauche de la souris sur un élément pouvant être ciblé de la sorte ;
- *:visited* : cible un lien déjà visité par l'utilisateur ;
- *:first-child* : cible le premier élément enfant.

Les pseudo-éléments, quant à eux, sont assez peu connus finalement, mais voici ce que propose CSS2.1.

- *:before* : permet d'insérer un contenu avant un élément et de le styler « à la volée » ;
- *:after* : permet d'insérer un contenu après un élément et de le styler « à la volée » ;
- *:first-letter* : permet de styler la première lettre d'un élément ;
- *:first-line* : permet de styler la première ligne d'un bloc.

Vous pouvez visiter ce site pour avoir plus d'infos et pratique de CSS

Les deux balises * * et ** sont des conteneurs destinés à structurer le contenu, ils ont un rôle complémentaire et des règles de rendu différentes. ** sert surtout à associer un style à une partie d'un texte tandis que * * sert à agencer le contenu de la page. Les deux balises doivent avoir une balise de début et de fin.

Ces deux balises sont utilisées pour appliquer le formatage et la mise en forme décrite dans le CSS au contenu de la page HTML.

*La balise * est un bloc, donc un objet rectangulaire qui ne peut être réparti sur plusieurs lignes. Par défaut, elle occupe toute la largeur disponible dans son élément parent. Elle possède les attributs *margin(marges extérieures), padding(marges intérieures), width(largeur), height(hauteur), border (bordures)*. Elle est précédée et suivie d'un saut de ligne.

*La balise * peut être placée à l'intérieur d'un paragraphe pour délimiter une partie de celui-ci sans affecter en soi l'apparence du texte ; elle est dite est *inline*. On ne peut lui spécifier une hauteur ni une largeur ni l'entourer d'une marge.

Cet élément va s'avérer pratique dans certains cas où nous aimerions mettre en forme une portion de texte à l'intérieur d'un élément. La balise span sera souvent utilisée avec un attribut class afin de pouvoir le cibler plus facilement en CSS.

2. Exercice

[Solution n°3 p 25]

Exercice

1) Laquelle de ces propositions est exacte ?

- ◯ On doit donner une valeur unique à un attribut *id* mais pas forcément à des attributs *class*
- ◯ On doit donner une valeur unique à un attribut *class* mais pas forcément à des attributs *id*
- ◯ On doit donner une valeur unique à chaque attribut *class* et *id* dans une page

Exercice

2) Quelle est la différence majeure entre les éléments div et span ?

- ◯ L'élément *span* est de *type block*, l'élément *div* est de *type inline*
- ◯ L'élément *div* est de *type block*, l'élément *span* est de *type inline*
- ◯ L'élément div sert de conteneur, au contraire de l'élément *span*

Exercice

3) Un élément de type block...

- ◯ occupe toute la largeur disponible dans la page
- ◯ n'occupe que la largeur nécessaire à son contenu
- ◯ occupe toute la largeur disponible dans son élément parent

Exercice

4) Quel sélecteur CSS permet de cibler par son id ?

- ◯ .toto
- ◯ {toto}
- ◯ #toto

Exercice

5) Quel sélecteur permet de cibler par sa class ?

- ◯ <titi>
- ◯ .titi
- ◯ [titi]

Section 4 : Propriétés de mise en forme du contenu avec CSS

IV

1. 1. La mise en page des paragraphes

- *text-align* : aligne le paragraphe, à droite : *right*, à gauche : *left*, centré : *center*, justifié : *justify* , ou le même alignement que son parent : *inherit*.
- *text-indent* : règle la tabulation de la première ligne d'un paragraphe : en pixel ou en em
- *text-transform* : précise la casse. Première lettre en majuscule : *capitalize*, texte en majuscules : *uppercase*, texte en minuscules : *lowercase*, normal :*none*, *inherit* pour héritage de son parent.
- *line-height* : règle l'interlignage : en pixels, em ou pourcentage, *inherit* ou *normal*.
- *white-space* : gère les espaces et les passages à la ligne. *nowrap* : pas de retour à la ligne, *pre* : texte tel qu'il a été écrit.
- *word-spacing* : Règle l'espace entre les mots : en pixels, em ou pourcentage, *inherit* ou *normal*.
- *letter-spacing* : Règle l'espace entre les lettres et les mots: en pixels, em ou pourcentage, *inherit* ou *normal*.
- *text-decoration* : Règle les soulignements de texte. *underline* : soulignement, *overline* : surlignement, *line-through* : texte barré, *blink* : clignotement (sauf IE), *none* : aucun

2. 2. La police de caractère ou font

On peut regrouper dans la propriété font ses différentes sous-propriétés :

- *font-family* : familles de caractères : *Verdana, Geneva, sans-serif, "Times New Roman", Times, serif/"Palatino Linotype", "Book Antiqua", Palatino, serif*...on écrit entre guillemets la police précise, puis ensuite des polices courantes de remplacement
- *font-size* : taille en pixels, en pourcentage, en em : *small,x-small, xx-smal,medium, large, larger, x-large, xx-large*.
- *font-stretch* : règle l'étirement de la police : *normal*, *condensed*, *expanded*...
- *font-style* : précise si la police est en *italic*, *oblique* ou *normal*.
- *font-variant* : avec cette propriété on peut obtenir des petites capitales: *small-caps*
- *font-weight* : règle l'épaisseur du trait, de *100* à *900* ou *bold* pour mettre en gras.

3. 3. Les couleurs et les fonds

Deux propriétés permettent d'ajouter des couleurs à votre site : *color* et *background*

- *color*

La propriété *color* donne la couleur au texte. Pour un titre coloré par exemple : h1 { color:#0033FF;}

- *background*

- *background-color :* donne une couleur au fond

- *background-image :* donne une image de fond, elle se rédige de cette façon :

élément *{background-image:url(images/monimage.jpg);}*

- *background-repeat :* précise le mode de répétition de l'image : *repeat*, *repeat-x, repeat-y, no-repeat;*

- *background-attachment* : effet de filigrane : *scroll* ou *fixed* (le fond reste fixe)

- *background-position :* si vous imposez avec *background-repeat* que l'image ne soit affichée qu'une fois, vous pouvez fixer la position d'arrière-plan exacte où le graphique doit être placé dans la page : vous précisez les coordonnées du point supérieur gauche.

- *background :* vous pouvez regrouper en une seule mention vos propriétés, par exemple :

élément {background: url(images/monimage.jpg) no-repeat #ccff42;}

4. 4. Les listes

Les listes créées à l'aide des balises *<ul>* ou *<ol>* peuvent être modifiées en CSS. En plus des polices, marges et autres propriétés qui peuvent leur une apparence différente, il existe trois propriétés qui sont spécialement conçues pour elles.

- *Propriété list-style-type*

La propriété list-style-type permet de définir le type de puce des éléments de la liste.

- *circle* : pour ul contour de cercle

- *disc* : pour ul cercle plein

- *square* : pour ul puce rectangulaire

- *none* : pas de puce

Cependant, même si on a défini la liste non ordonnée (à l'aide de la balise <ul>) on peut quand même lui donner l'apparence d'une liste ordonnée comme si elle était définie (avec la balise <ol>) grâce aux valeurs suivantes:

- *decimal* : pour les nombres décimaux commençant de 1.

- *decimal-leading-zero* : pour les nombres décimaux avec zéro initial commençant de 01.

- *upper-roman* : pour les nombres romains majuscules commençant de I.

- *lower-roman* : pour les nombres romains minuscules commençant de i.

- *lower-greek* : pour la numération grecque.

- *lower-alpha* : pour les lettres de l'alphabet minuscules.

- *upper-alpha* : pour les lettres de l'alphabet majuscules.

- *Propriété list-style-image*

Si on veut définir nos propres puces, autant le faire avec des images. La propriété *list-style-image*

la syntaxe est : *list-style-image:url("chemin_ de_ l_ image")*

- *Propriété list-style-position*

La propriété *list-style-position* permet de définir la position de la puce par rapport à la boite contenant les éléments de la liste. Elle peut avoir deux valeurs: *inside* et *outside.*

5. 5. Les bordures

La propriété *border* permet d'ajouter une bordure, on peut préciser son épaisseur et son style. Elle se rédige de cette façon : *élément {border:couleur style épaisseur;}*. Les différents styles sont les suivants:

- *solid* : trait continu

- *dotted* : trait pointillé

- *dashed* : petits tirets

- *double* : trait double et plein

- *groove* : trait en relief

- *ridge* : effet 3D

- *inset*: rentrante

- *outset* : sortante

- *none* : pas de bordure ou transparente

- *hidden* : pas de bordure ou transparente pour les tableaux

L'épaisseur peut être *thin* : fine, *medium* : moyenne, *thick* : large ou de l'épaisseur en pixel que l'on souhaite. La bordure peut s'appliquer au contour général mais également à un seul côté, deux ou trois, selon votre convenant.

On agit alors sur la *border-left* (bordure gauche), *border-top* (bordure haute), *border-right* (bordure droite), *b*order-*bottom* (bordure inférieure).

6. Exercice

[Solution n°4 p 26]

Exercice

1) Dans le code CSS suivant : bloc2 {border:#E6E6EE2 solid 2px;}, quelle est le style de la bordure du bloc2 ?

- ◯ Un trait plein
- ◯ Des pointillées
- ◯ Un trait en 3D

Exercice

2) Quelle propriété ou balise CSS puis-je utiliser pour justifier un texte ?

- ◯ text-trasform
- ◯ text-align
- ◯ text-iddent

Exercice

*3) La balise <ul> a le même rôle et effet que la balise *

- ◯ faux par ce que <ul> définit une liste non ordonnée et <ol> une liste ordonnées
- ◯ vrai par ce que <ul> définit une liste non ordonnée et <ol> une liste ordonnées
- ◯ faux par ce que <ol> définit une liste non ordonnée et <ul> une liste ordonnées
- ◯ vrai par ce que <ol> définit une liste non ordonnée et <ul> une liste ordonnées

Exercice

4) Laquelle des valeurs suivante n'est pas une valeur de l'élément font-style ?

- ◯ italic
- ◯ normal
- ◯ bold

Exercice

5) Quelle propriété permet de définir explicitement la couleur de fond d'une page ?

- ◯ color
- ◯ background
- ◯ background-color

Section 5 : les blocs et leur positionnement

V

1. 1. Marges externes, marges internes et dimensions

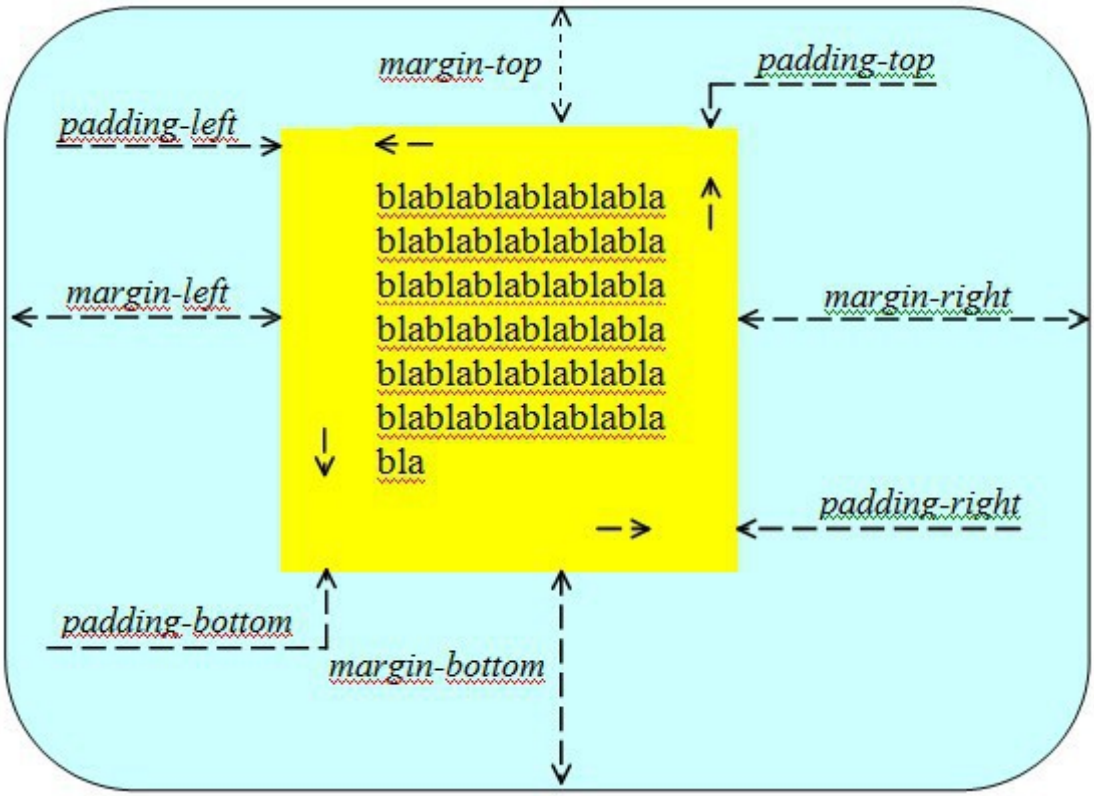
Un bloc se place par défaut en haut à gauche de son conteneur (le body, un autre bloc,...) et occupe toute la largeur de ce conteneur. Ses frères se placeront en-dessous.

Les marges permettent de positionner un bloc au sein de son parent :

- Marges externes définies par la propriété *margin*.
- Marges internes définies par la propriété *padding*.

Les définitions des marges externes et internes sont similaires :

- *margin-top* et *padding-top* pour la marge supérieure
- *margin-right* et *padding-right* pour la marge droite
- *margin-bottom* et *padding-bottom* pour la marge inférieure
- *margin-left* et *padding-left* pour la marge gauche



On peut affecter les valeurs des quatre marges de deux façons équivalentes en écrivant

```
/* notation classique */

#comment{
   margin-top : 20px ;
   margin-right : 30px ;
   margin-bottom : 30px ;
   margin-left : 20px ;
}
```

ou bien

```
/* notation globalee */

#comment{
   margin: 20px 30px 30px 20px;
}
/* les 4 valeurs sont toujours
dans l'ordre top, right, bottom,
left */
```

En notation globale :

- *margin : 20px* ; affecte la même dimension aux 4 marges soit 20px.
- *margin : 20px 30px ;* affecte 20px en haut et en bas et 30px à droite et à gauche.
- *margin : 20px 30px 10px ;* affecte 20px en haut, 30px à droite et à gauche et 10px en bas.

Les termes width et height désignent les dimensions d'un bloc

- *width* pour la largeur du bloc
- *height* pour la hauteur du bloc

Exemples d'application

a. Positionner un bloc a l'intérieur d'un autre

On désire placer un bloc jaune de 100px X 100px à 80px du haut du conteneur et à 35px de la gauche du conteneur.

Code HTML :

```html
<div class="conteneur">
    <div class="bloc">bloc positionné </div>
</div>
```

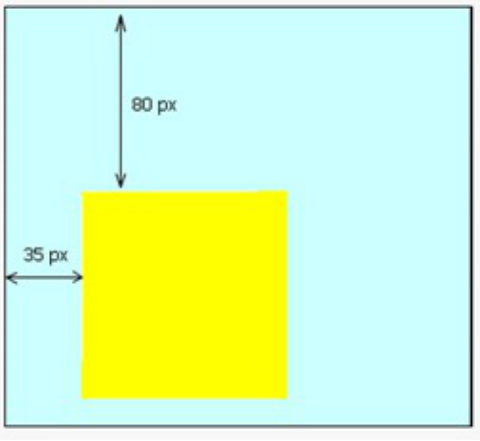
Partie CSS :

```css
.conteneur{
width: 300px;
height: 250px;
padding-top: 80px;
padding-left: 35px;
border: 1 px solid #000000;
backgroung-color:blue;
}
.bloc{
width: 100px;
height: 100px;
background-color: yellow;
}

```

Le résultat ressemble a cela :

Nota : On emploie ici un *padding-top* et un p*adding-left* pour définir le positionnement du bloc à l'intérieur du conteneur (marge interne). Cela signifie que la zone située autour du bloc jaune ne pourra pas être remplie par du texte.



b. Deux blocs l'un au-dessous de l'autre avec un espace de séparation :

Partie HTML

```
<div class="bloc1">bloc1</div>
<div class="bloc2">bloc2</div>
```

CSS correspondant

```
.bloc1 {
  background-color: aqua;
  height: 50px;}
.bloc2 {
  background-color: lime;
  height: 50px;
  margin-top: 20px;
}
```

c. Trois blocs côte à côte séparés par un espace

Partie HTML

```
<div class="bloc1">bloc1</div>
<div class="bloc2">bloc2</div>
<div class="bloc3">bloc3</div>
```

CSS correspondant

```
.bloc1 {
  background-color: aqua;
  height: 50px;
  width: 100px;
  float: left;
}
.bloc2 {
  background-color: lime;
  height: 50px;
  width: 100px;
  float: left;
  margin-left: 20px;
}
.bloc3 {
```

```
15  background-color: red;
16  height: 50px;
17  width: 100px;
18  float: left;
19  margin-left: 20px;}
```

La propriété *float:* fera l'objet de la section suivante.

2. 2. Positionnement flottant

La propriété *float* permet de positionner un bloc à gauche (*float : left;*) ou à droite (*float : right;*)à l'intérieur du bloc parent (et non-plus l'un en-dessous de l'autre). Le reste du conteneur parent occupera alors l'espace laissé libre, à côté puis éventuellement au- dessous.

Exemple : Une image insérée à droite d'un texte.

Partie HTML :

```
1 <div class="conteneur">
2   <img class="image" src="..." alt="" />
3   <p>bla bla bla bla bla bla
4   bla bla bla bla bla bla
5   bla bla bla bla bla...
6   </p>
7 </div>
```

le CSS Correspondant est le suivant :

```
1 .conteneur {
2   width: 40%;
3   background-color: yellow;
4 }
5 .image {
6   float: right;
7 }
```

NB : Dans le cas de plusieurs éléments flottants, il est souvent préférable de les placer dans un parent commun. Il est possible de cumuler les propriétés *float* pour obtenir plusieurs blocs côte à côte.

3. 3. Positions absolues, fixes et relatives

- *Position "absolute", position "fixed"*

L'élément est placé à l'aide des propriétés "*top*" et "*left*" par rapport au coin supérieur gauche :

- de son parent si ce dernier est positionné, ou alors du dernier ancêtre positionné.

- de la page entière (balise *html*) si aucun Ancêtre n'est positionné.

- *Conséquences :*

La position de l'élément est indépendante de l'emplacement de sa balise à l'intérieur du code source du Parent. L'élément positionné en absolu ne décale pas et ne perturbe pas les autres données, auxquelles il peut cependant se superposer.

Ce positionnement rend difficile l'adaptation du site aux différentes résolutions d'écrans des visiteurs.

- *Différence entre "absolute" et "fixed":*

- "*absolute*" autorise le défilement des données positionnées

- "*fixed*" interdit le défilement des données positionnées

- *Position "relative":*

La position "*relative*" d'un élément est dépendante de la position de l'élément qui le précède dans le code HTML. Le code CSS donne la position finale du bloc à l'aide des propriétés "*top*", "*right*","

bottom" et "*left*".

4. Exercice

[Solution n°5 p 27]

Exercice

1) Quel attribut permet de spécifier une hauteur minimum ?

- ◯ mini-height
- ◯ min-height
- ◯ height-min

Exercice

2) La règle suivante

```
cadenas {
width:200px;
height:200px;
border:1px solid gray;
}
```

Signifie (sans tenir compte de margin et de padding) que :

- ◯ tous les blocs cadenas auront une taille de 200x200 pixels
- ◯ tous les blocs cadenas auront une taille de 200x0 pixels
- ◯ le premier cadenas de son conteneur aura une taille de 200x200 pixels

Exercice

3) La règle suivante

```
div {
float:left;
width:100px;
height:100px;
border:1px solid gray;
}
```

définit tous les div

- ◯ en mode absolu, à gauche
- ◯ comme flottants, à gauche
- ◯ dans le flux et à gauche

Exercice

4) En mode absolu, pour placer un bloc div (de classe contenu) à 100 px du bord gauche du conteneur, j'utilise la déclaration suivante :

- ◯ right:100px;
- ◯ top:100px;
- ◯ left:100px;

Exercice

5) La règle suivante :

div2 {

max-width:450px;

height:200px;

}

Définit un bloc div dont la largeur :

- ◯ ne s'adaptera pas en fonction de la taille de l'écran
- ◯ s'adaptera en fonction de la taille de l'écran sans dépasser 450px ;
- ◯ sera de 450 px, quelle que soit la taille de l'écran

Travaux Pratiques N° 10

VI

1. Les feuilles de style (CSS)

Durée de la vidéo : 12 minutes

Temps nécessaire pour le T.P : 25 minutes

La vidéo est également disponible sur YouTube : *https://youtu.be/JbIwi72PtVY*

Complément : Autres sources d'apprentissage et de T.P sur les CSS

Vous pouvez vous exercer chez vous avec d'autres TP sur les CSS et les formulaires :

- Ce TP de la Mozilla Developpers Networks (développeurs de Firefox) vous guide dans la création d'un formulaire avec utilisation du CSS

https://developer.mozilla.org/fr/docs/Web/Guide/HTML/Formulaires/Mon_premier_formulaire_H

- Ce autre T.P de PIERRE GUIRAUD vous guide dans la création d'une menu horizontal avec CSS qui vous sera très pratique :

http://pierre-giraud.fr/creation-dun-menu-horizontal-simple-en-html-en-css/

Solutions des exercices

> Solution n°1

Exercice p. 6

Exercice

- (●) Cascading Style Sheets
- ( ) Computer Style Sheets
- ( ) Colorfull Style Sheets

Exercice

- ( ) `<stylesheet>mon_style</stylesheet>`
- (●) `<link rel="stylesheet" type="text/css" href="mon_style.css">`
- ( ) `<stylesheet>mon_style.css</stylesheet>`

Exercice

- ( ) <css>... </css>
- ( ) <script>... </script>
- (●) <style>... </style>

Exercice

- ( ) Au début du document
- ( ) A la fin du document
- ( ) Entre les balises <body>et </body>
- (●) Entre les balises <head> et </head>

Exercice

- ( ) peux utiliser Microsoft Word.
- (●) peux utiliser un éditeur de texte quelconque tel que bloc-note.
- ( ) dois utiliser Microsoft Powerpoint.

Exercice p. 8

> Solution n°2

Exercice

- ○ // un commentaire //
- ○ */ un commentaire /*
- ◉ /* un commentaire */

Exercice

- ○ Un sélecteur
- ◉ Une propriété
- ○ Une déclaration

Exercice

- ○ une règle
- ○ une propriété
- ◉ un sélecteur
- ○ rien du tout

Exercice

- ○ h1 seulement
- ○ h1 ou .main ou .plug
- ◉ h1, .main et .plug
- ○ rien, cette écriture est fausse

Exercice

- ○ {p color : blue;}
- ◉ p {color : blue;}
- ○ p : color : blue

> Solution n°3

Exercice p. 12

Exercice

- ◉ On doit donner une valeur unique à un attribut *id* mais pas forcément à des attributs *class*
- ○ On doit donner une valeur unique à un attribut *class* mais pas forcément à des attributs *id*
- ○ On doit donner une valeur unique à chaque attribut *class* et *id* dans une page

Exercice

- ○ L'élément *span* est de *type block*, l'élément *div* est de *type inline*
- ◉ L'élément *div* est de *type block*, l'élément *span* est de *type inline*
- ○ L'élément div sert de conteneur, au contraire de l'élément *span*

Exercice

- ○ occupe toute la largeur disponible dans la page
- ○ n'occupe que la largeur nécessaire à son contenu
- ◉ occupe toute la largeur disponible dans son élément parent

Exercice

- ○ .toto
- ○ {toto}
- ◉ #toto

Exercice

- ○ <titi>
- ◉ .titi
- ○ [titi]

> Solution n°4

Exercice p. 16

Exercice

- ◉ Un trait plein
- ○ Des pointillées
- ○ Un trait en 3D

le mot *solid* désigne un trait plein

Exercice

- ○ text-trasform
- ◉ text-align
- ○ text-iddent

Exercice

- ○ faux par ce que <ul> définit une liste non ordonnée et <ol> une liste ordonnées
- ◉ vrai par ce que <ul> définit une liste non ordonnée et <ol> une liste ordonnées
- ○ faux par ce que <ol> définit une liste non ordonnée et <ul> une liste ordonnées

- ○ vrai par ce que <ol> définit une liste non ordonnée et <ul> une liste ordonnées

Exercice

- ○ italic
- ○ normal
- ◉ bold

bold est une valeur de l'élément *font-weight*

Exercice

- ○ color
- ○ background
- ◉ background-color

background permet aussi de définir la couleur de font mais pas de manière explicite.

> Solution n°5

Exercice p. 21

Exercice

- ○ mini-height
- ◉ min-height
- ○ height-min

Exercice

- ◉ tous les blocs cadenas auront une taille de 200x200 pixels
- ○ tous les blocs cadenas auront une taille de 200x0 pixels
- ○ le premier cadenas de son conteneur aura une taille de 200x200 pixels

Exercice

- ○ en mode absolu, à gauche
- ◉ comme flottants, à gauche
- ○ dans le flux et à gauche

Exercice

- ○ right:100px;
- ○ top:100px;
- ◉ left:100px;

Exercice

- ○ ne s'adaptera pas en fonction de la taille de l'écran

- ◉ s'adaptera en fonction de la taille de l'écran sans dépasser 450px ;
- ○ sera de 450 px, quelle que soit la taille de l'écran