Chapitre II

I. Rappel

Site sans CSS:

CSS Zen Garden

The Beauty of CSS Design

A demonstration of what can be accomplished through CSS-based design. Select any style sheet from the list to load it into this page.

Download the example html file and css file

The Road to Enlightenment

Littering a dark and dreary road lay the past relics of browser-specific tags, incompatible DOMs, broken CSS support, and abandoned browsers.

We must clear the mind of the past. Web enlightenment has been achieved thanks to the tireless efforts of folk like the W3C, WaSP, and the major browser creators.

The CSS Zen Garden invites you to relax and meditate on the important lessons of the masters. Begin to see with clarity. Learn to use the time-honored techniques in new and invigorating fashion. Become one with the web.

So What is This About?

There is a continuing need to show the power of CSS. The Zen Garden aims to excite, inspire, and encourage participation. To begin, view some of the existing designs in the list. Clicking on any one will load the style sheet into this very page. The HTML remains the same, the only thing that has changed is the external CSS file. Yes, really.

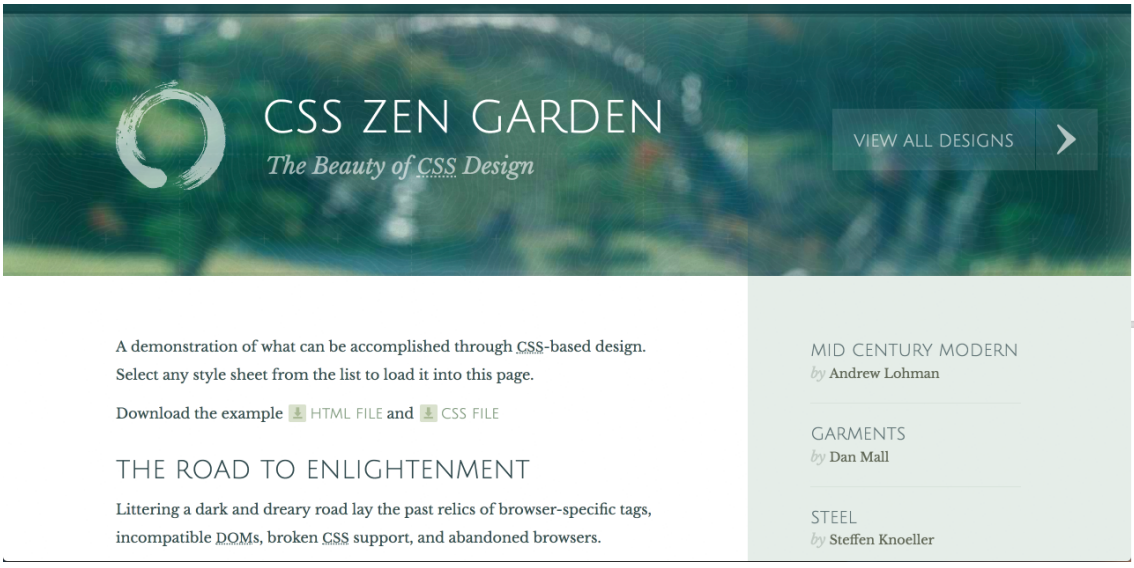
CSS allows complete and total control over the style of a hypertext document. The only way this can be illustrated in a way that gets people excited is by demonstrating what it can truly be, once the reins are placed in the hands of those able to create beauty from structure. Designers and coders alike have contributed to the beauty of the web; we can always push it further.

Participation

Strong visual design has always been our focus. You are modifying this page, so strong CSS skills are necessary too, but the example files are commented well enough that even CSS novices can use them as starting points. Please see the CSS Resource Guide for advanced tutorials and tips on working with CSS.

You may modify the style sheet in any way you wish, but not the HTML. This may seem daunting at first if you've never worked this way before, but follow the listed links to learn more, and use the sample files as a guide.

Site avec CSS :



II. Capacités du CSS

Ce que permet le CSS :

- Positionnement des éléments HTML
- Modifier le style des éléments HTML :
    - Police de texte
    - Couleur de texte, de fond
    - Ajouter des bordures
- Animations

Ce que ne permet pas le CSS :

- Algo
- Calcule
- Gestion de variables

III. Fonctionnement du CSS

Pour ajouter du CSS à une page HTML vous avez plusieurs solutions. Soit directement utiliser l'attribut « style » d'un élément HTML, soit utiliser une balise « style » ou encore créer une feuille de style et la lier à votre page HTML.

Modifier l'attribut style :

```
<balise style="option: valeur; option2: valeur;"></balise>
```

La syntaxe est la suivante :

- « style='' » : Ajout de l'attribut sur l'élément HTML
- « option » : L'option de stylisation que l'on veut modifier
- « valeur » : La valeur que l'on veut donner à l'option de stylisation que l'on veut modifier
- « option : valeur ; » : Vous pouvez directement ajouter plusieurs options de style sur un même élément html en les séparant par un point-virgule.

Exemple :

```
<p style="color: red; background-color: black;">Texte avec style modifié</p>
```

Utiliser une balise <style>:

```
<style>
        selecteur {
            option: valeur;
            option2: valeur;
            option3: valeur;
        }
</style>
```

La syntaxe est la suivante :

- « selecteur { } » : Identification l'élément que l'on veut modifier, par exemple par son nom de balise.
- « option » : L'option de stylisation que l'on veut modifier
- « valeur » : La valeur que l'on veut donner à l'option de stylisation que l'on veut modifier
- « option : valeur ; » : Vous pouvez directement ajouter plusieurs options de style sur un même identifiant en les séparant par un point-virgule.

Exemple :

```
    <style>
        p {
            color: red;
            background-color: black;
        }
    </style>
```

Créer une feuille de style:

index.html :

```
<head>
    <link rel="stylesheet" type="text/css" href="chemin/vers/fichier.css">
</head>
```

style.css :

```
selecteur {

    option: valeur;
    option2: valeur;
    option3: valeur;

}
```

Exemple:

index.html :

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

style.css :

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

Conclusion :

Dans cette partie nous avons vu comment appliquer du CSS à une page HTML. Nous avons vu qu'il y avait plusieurs manières de le faire, mais quelle est la meilleure pratique à adopter ?

Et bien généralement on utilise toujours une feuille de style séparée de notre page HTML. Tout simplement par propreté et lisibilité du code. La mise en forme à un endroit et la mise en page à un autre.

L'utilisation de l'attribut style peut être pratique lorsque l'on veut modifier peu d'options sur un unique élément HTML.

Nous avons vu que pour appliquer du CSS via une feuille de style il faut identifier l'élément HTML que l'on veut modifier. Mais comment identifier un élément ?

IV. Sélection des éléments HTML

a) Intérêt de la sélection

Pour expliquer l'intérêt de la sélection d'éléments HTML nous allons prendre l'exemple suivant :

Admettons que nous ayons une page HTML avec plusieurs balises <p> et plusieurs balises <h1>. Nous voulons rendre notre page plus corporate et utiliser les couleurs de la charte graphique de notre entreprise

Avec attribut style :

```
<h1 style="color: blueviolet;">Nom de notre super
entreprise!</h1>

<h1 style="color: blueviolet;"> Un autre titre </h1>

<p style="color: skyblue;"> Description de notre entreprise.
</p>

<p style="color: skyblue;"> Encore plus d'informations </p>
```

Nous avons utilisé l'attribut « style » sur chacun de nos éléments pour modifier leur style. Mais l'on remarque que c'est redondant. Tous nos paragraphes ont la même couleur et nos titres ont aussi la même couleur. C'est à ce moment-là l'intérêt de l'identification est évident.

Avec selection:

- index.html :

```
<h1>Nom de notre super entreprise!</h1>

<h1> Un autre titre </h1>

<p> Description de notre entreprise. </p>

<p> Encore plus d'informations </p>
```

- style.css :

```
h1 {
    color: blueviolet;
}

p {
    color: skyblue;
}
```

En utilisant la selection nous avons pu éviter de redonder notre code et ainsi d'écrire une modification de style une seule fois.

b) Les différentes sélections

Dans cette partie nous allons voir qu'il existe plusieurs manières de sélectionner un élément HTML. Jusqu'ici nous avons utilisé la sélection la plus simple qui est de sélectionner une balise par son nom.

Le nom de la balise :

Utiliser le nom d'une balise pour modifier la mise en forme d'un élément HTML peut être intéressant seulement lorsque le style que l'on veut appliquer est valable pour tous les éléments HTML portant ce nom.

```
balise {

}
```

Une « class » :

Utiliser une classe pour mettre en forme un élément HTML est la méthode la plus utilisée. Car elle permet de modifier le style de plusieurs éléments HTML à la fois sans pour autant forcément modifier tous les éléments portant le même nom.

- style.css:

```
.nom_class {

}
```

- index.html:

```
<balise class="nom_class"></balise>
```

Tous les éléments HTML ayant l'attribut « class » avec une valeur « nom_class » auront la mise en forme implémenter dans le fiche de style pour la class « nom_class ».

Un identifiant unique :

L'utilisation d'un identifiant unique pour sélectionner un élément HTML dans notre fiche de style est généralement utilisé lorsque l'on veut changer la mise en forme d'un seul élément HTML tout en modifiant un nombre important d'options qui nous empêchent d'utiliser directement l'attribut « style » sur l'élément correspondant.

Il faut faire attention à la valeur que l'on donne à un identifiant unique. Comme son nom l'indique il est important que la valeur de celui-ci soit unique. Ce qui veut dire que l'identifiant unique de deux éléments HTML ne doivent pas avoir la même valeur.

- index.html :

```
<balise id="identifiant_unique"></balise>
```

- style.css :

```
#identifiant_unique {

}
```

Exemple :

- index.html :

```
<h1 id="titre_entreprise">nom entreprise</h1>

<p class="texte_bleu"> texte de couleur bleu </p>

<p class="texte_bleu"> texte de couleur bleu </p>

<p class="texte_bleu"> texte de couleur bleu </p>
```

- style.css :

```
#titre_entreprise {
    color: violet;
}

.texte_bleu {
    color: blue;
}

p {
    font-size: 1.2em;
}
```

c) Autres sélections

Liste :

On peut utiliser une liste de sélecteurs si l'on veut appliquer du style à plusieurs « class », balises ou identifiants à la fois. Cette selection fonctionne comme un OU logique.

```
selecteur, selecteur2, selecteur3 {

}
```

Exemple :

On veut que tous les titres soient de couleur rouge.

```
h1, h2, h3, h4, h5, h6 {
    color: red;
}
```

Balise contenue dans une autre :

On peut utiliser cette sélection lorsque l'on veut appliquer du style sur des balises qui sont seulement à l'intérieur d'une autre balise.

```
selecteur selecteur2 {

}
```

Exemple :

On veut modifier seulement la balise <h1> se trouvant dans le <header> de notre page.

- index.html :

```
    <header>
        <h1>
            Titre dans un header
        </h1>
    </header>
```

- style.css :

```
header h1 {
    color: violet;
}
```

Balise placée juste après :

On peut utiliser cette sélection lorsque l'on veut modifier des éléments HTML qui sont placés juste après une sélection d'éléments HTML dans la hiérarchie de notre page.

```
selecteur + selecteur2 {

}
```

Exemple :

- index.html :

```
<h1>
    Titre dans un header
</h1>
<p>Sous titre </p>
```

- style.css :

```
header + p {
    opacity: 0.8;
}
```

Tous les éléments :

Cette sélection est à utiliser lorsque l'on veut appliquer de la mise en forme sur tous les éléments HTML.

```
* {

}
```

Il existe bien d'autres manières de sélectionner des éléments HTML en CSS mais nous avons vu ici les principales, les plus utilisées.

Tous les sélecteurs :

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

V. Positionner des éléments

Dans cette partie nous allons voir les options CSS qui nous permettent de positionner nos éléments HTML au sein de notre page.

Il existe plusieurs de manières de positionner des éléments HTML en CSS, on peut les positionner par rapport aux éléments voisins, par rapport à la fenêtre, par rapport au scroll etc...

a) La propriété « position »

La propriété position spécifie le type de méthode de positionnement utilisé pour un élément.

```
selecteur {
    position: static;
}
```

Options :

- static :
    - Valeur par défaut
    - Pas de position particulière, respecte le fonctionnement normal de la hiérarchie HTML
- relative :
    - Positionne l'élément par rapport à sa position normale
    - Donner une valeur aux propriétés top, left, bottom, right d'un élément avec une « position : relative ; » lui fera changer sa position par rapport à sa position normale en fonction de la valeur donnée à ses propriétés
- fixed :
    - Positionne l'élément par rapport à la fenêtre, ce qui signifie qu'il restera toujours à la même position même si la page est « scrollée ».
    - Les propriétés top, left, bottom, right sont utilisées pour positionner l'élément
- absolute :
    - Positionne l'élément par rapport à son parent.
    - Si un élément avec une « position : absolute ; » n'a pas de parent alors il aura la même comportement que s'il avait une « position : fixed ; »

Exemple :

Admettons que l'on veut un en tête qui est toujours visible même lorsque l'utilisateur scroll, plusieurs paragraphes et une image avec du texte positionner en haut à droite.

- index.html :

```
    <header>
        <h1>Mon super header</h1>
    </header>
    <p>Lorem</p>
    <div class="container">
        <img src="image" alt="super image" />
        <p class="topleft">Texte en haut à gauche</p>
    </div>
    <p>Lorem</p>
    <p>Lorem</p>
```

- style.css :

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

.container {
    position: relative;
}

.topleft {
    position: absolute;
    top: 0;
    left: 0;
}
```

b) Les propriétés top, bottom, left, right

- Top: positionne l'élément par rapport au haut de l'élément parent.
- Bottom : positionne l'élément par rapport au bas de l'élément parent.
- Left : positionne l'élément par rapport à la gauche de l'élément parent.
- Right : positionne l'élément par rapport à la droite de l'élément parent.

    - Prend en valeurs : auto, initial,inherit ou une valeur numérique.

Utile lorsque l'élément a une « position : absolute ; ».

c) Aligner des éléments

Lorsque construit notre page en HTML les éléments ont tendance à s'aligner de manière verticale. En CSS nous avons accès à la propriété « display » qui nous permet d'aligner des éléments HTML comme on le souhaite.

```
p {
    display: block;
}
```

Options (principales):

- inline : Affiche un élément en ligne (valeur par défaut pour)
- block : Affiche un élément en block (valeur par défaut pour , <p>, etc…)
- inline-block : Affiche un élément comme un conteneur de blocs en ligne. Lui-même est positionner en ligne, ses enfants sont afficher en block.
- none : L'élément n'est pas affiché
- inherit : L'élement a la même valeur de display que son parent

Exemple :

Admettons que l'on veut une page avec un élément caché, deux conteneur côte à côte avec du texte à l'intérieur qui sont alignés verticalement.

- index.html :

```
    <h1 style="display: none;">Titre invisible</h1>

    <div class="container">
        <span> Lorem</span>

        <span> Lorem</span>

        <span>Lorem</span>

        <span> Lorem</span>
    </div>

    <div class="container">
        <span> Lorem</span>

        <span> Lorem</span>

        <span>Lorem</span>

        <span> Lorem</span>
    </div>
```

- style.css :

```
.container {
    display: inline-block;
}

span {
    display: block;
}
```

VI. Changer la forme

a. Changer la taille d'un élément

Pour changer la taille d'un élément HTML on peut modifier les propriétés « width », « height », « max-height » ou « max-width ».

Toutes ces propriétés peuvent prendre les valeurs suivantes :

- Valeur numérique + px : Définit la taille en pixels.
- Valeur numérique + % : Définit la taille en fonction du ratio de la taille du parent.
- auto : Taille définit automatiquement en fonction du contenu de l'élément.

Exemple:

- index.html

```
    <div class="container">
        <span class="texte"> Lorem</span>
    </div>
    <div class="container">
        <span class="texte"> Lorem</span>
    </div>
```

- style.css

```
.container {
    display: inline-block;
    width: 49%;
    height: auto;
}

.texte {
    width: 50px;
    height: auto;
}
```

b. Margin et Padding

Les propriétés CSS « margin » et « padding » permettent de rajouter de l'espace entre des éléments HTML. Ces propriétés utilisent les mêmes unités que les propriétés « width » et « height » (px, %, auto).

La seule différence entre « margin » et « padding » est la zone sur laquelle l'espace est exercé. Les « margin » affecte la zone hors des bordures de l'élément alors que le « padding » affecte la zone à l'intérieur des bordures de l'élément.

Il existe deux manières d'utiliser les « margin » et « padding » :

```
selecteur {
    (margin|padding)-(top|bottom|left|right): valeur;
}
```

ou

```
selecteur {
    margin|padding: valeur(top) valeur(right) valeur(bottom) valeur(left);
}
```

Exemple :

```
p {
    margin-top: 1px;
    margin-bottom: 1px;
    margin-left: 1px;
    margin-right: 1px;
}
```

ou

```
p {
    margin: 1px 1px 1px 1px;
}
```

Petit tuyau :

Pour centrer un élément vous pouvez utiliser margin comme suit :

```
.centrer {
    margin: 0 auto;
}
```

c. Bordures

Pour styliser votre page HTML vous pouvez aussi rajouter des bordures à vos éléments. Vous pouvez ajouter une bordure sur un seul ou plusieurs côtés de votre élément, avec une certaine épaisseur, un certain style, couleur et un niveau d'arrondi.

Elément avec bordure sur tous les côtés :

```
selecteur {
    border-width: 2px;
    border-style: solid;
    border-radius: 5px;
    border-color: red;
}
```

ou

```
selecteur {
    border: 2px solid red;
    border-radius: 5px;
}
```

Élément avec bordure différente sur chaque côté :

```
selecteur {
    border-width: 2px 1px 2px 3px;
    border-style: solid solid solid solid;
    border-radius: 5px 10px 3px 2px;
    border-color: red green yellow black;
}
```

ou

```
selecteur {
    border-top-width: 2px;
    border-bottom-width: 1px;
    border-right-width: 4px;
    border-left-width: 5px;
}
```

ou

```
selecteur {
    border-left: 2px solid red;
    border-radius: 5px;
}
```

Plus d'infos : https://www.w3schools.com/css/css_border.asp

VII. Changer la couleur

a. La couleur du texte

```
selecteur {
    color: red;
}
```

b. La couleur de fond

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

VIII. Pratiquer

Sujet : Mettre en forme le travail fait en HTML

Mes infos :

- Mail : aniel.jeanbaptiste@gmail.com
- Linkedin : https://www.linkedin.com/in/jean-baptiste-aniel-185abb97/
- Twitter: https://twitter.com/_rhanb
- Github: https://github.com/rhanb/

Ressources :

- Styliser du texte: https://www.w3schools.com/css/css_text.asp
- Tutoriel : https://www.w3schools.com/css/