

Exercice 8. Les styles CSS

1. Une feuille de style est un fichier dont l'extension est `.css`. On définit dans ce fichier nos préférences concernant l'affichage du contenu de notre page HTML : les polices d'écriture, la taille des caractères pour les textes, les couleurs, la taille des images, arrière-plans, le positionnement des différentes sections, etc.

Nous avons vu (durant l'exercice 6) les trois façons de définir les styles d'une page HTML. L'avantage des styles externes, c'est qu'une modification dans cette feuille de style principale affectera toutes les pages qui lui sont liées.

2. Créez un fichier `mise-en-forme.html` dans le répertoire `/EnvSystemeReseau/tp-html`. Copiez-y le contenu du fichier `base.html`. Créez ensuite un nouveau fichier `mes-styles.css` dans le même répertoire.

3. Insérez, dans la balise `<head>` de votre page HTML, la balise suivante :

```
5 <link rel="stylesheet" type="text/css" href="mes-styles.css" >
```

Cette balise vous permet de créer un raccourci vers le fichier dans lequel vous définissez les styles d'affichage de votre page :

- La balise `<link>` indique au navigateur qu'il doit chercher un document situé à l'extérieur de la page HTML.
- L'attribut `rel="stylesheet"` précise que le document en question est une feuille de style.
- L'attribut `type="text/css"` précise le type de la feuille de style, du CSS en l'occurrence.
- Et L'attribut `href="URL"` donne l'URL de la feuille de style, c'est-à-dire son emplacement.

1. À présent, insérez dans le contenu de la page (dans la balise `<body>`) un titre `<h1>` dont le contenu est *Mise en forme par les styles CSS*. Insérez ensuite un titre `<h2>` dont le contenu est *Un premier petit exemple* suivi d'un paragraphe `<p>` que vous remplirez avec un texte quelconque de petit taille (à rechercher sur le Web). Insérez une deuxième fois un titre `<h2>` avec pour contenu *Un deuxième petit exemple* suivi d'un paragraphe que vous remplirez cette fois avec un texte de longueur moyenne ou grande.

Nous allons, par la suite, définir, dans notre fichier `mes-styles.css`, l'aspect global de la page `mise-en-forme.html` lors de son affichage. Pour cela, nous allons fournir au navigateur les **règles** d'affichage du contenu.

2. La syntaxe d'une règle CSS est de la forme suivante : **sélecteur {propriété1: valeur1; propriété2: valeur2; ...}**. Une règle est donc composée de deux parties : un **sélecteur** et une ou plusieurs **déclarations**. Une déclaration comprend elle-même deux parties : une **propriété** et une **valeur** associée à cette propriété. Ces deux parties sont séparées par le caractère deux-points (:) et la déclaration est terminée par un point-virgule (;). Dans l'exemple suivant, le sélecteur est `p` et la déclaration est `color: blue` ; dans la déclaration, la propriété est `color` et sa valeur est `#e8eae8`.

```
1 p { color: blue; }
```

- Tous les types d'éléments HTML (les noms des balises) peuvent être des sélecteurs.
- La liste des propriétés et leurs valeurs est définie dans la spécifications du w3c, que vous pourrez trouver à l'adresse suivante : <http://www.yoyodesign.org/doc/w3c/css1/index.html#css1-properties>.

1. **Les sélecteurs.** Un sélecteur peut contenir plusieurs noms de balises, séparés de différentes manières.

- Les **sélecteurs multiples** contiennent une liste de noms de balises HTML séparés par des virgules. Ils permettent d'appliquer le même bloc de règles de styles à plusieurs éléments de la page HTML. Les deux exemples suivants sont équivalents :

```
3 h1 { text-align: center; }
4 h2 { text-align: center; }
5 h3 { text-align: center; }
6 h4 { text-align: center; }
7 h5 { text-align: center; }
```

```
1 h1, h2, h3, h4, h5 { text-align: center; }
```

○ Les **sélecteurs contextuels** contiennent une liste de noms de balises HTML séparés par un espace. Ils permettent d'appliquer des styles à une balise lorsqu'elle se trouve dans un contexte donné. Dans le sélecteur, les balises sont listées dans leur ordre d'imbrication. L'exemple suivant définit la valeur de la propriété **list-style** des balises **** selon leur niveau dans une liste :

- si **** est imbriquée dans un seul ****, alors le navigateur utilise la valeur **upper-roman** de la propriété **list-style**
- si **** est imbriquée dans deux ****, alors il utilise la valeur **upper-alpha**
- si **** est imbriquée dans trois ****, alors il utilise la valeur **décimal**

```
10 ol li { list-style: upper-roman; }
11 ol ol li { list-style: upper-alpha; }
12 ol ol ol li { list-style: decimal; }
```

```
I. niveau 1
  A. niveau 2
    1. niveau 3
    2. niveau 3
  B. niveau 2
    1. niveau 3
    2. niveau 3
```

○ Le **sélecteur universel**, noté *****, applique les styles qui lui sont associés à tous les éléments du document. Avec la règle de l'exemple suivant, tous les textes de la page HTML seront affichés en violet et dans la police Times New Roman.

```
1 * { color: purple; font: "Times New Roman"; }
```

○ Les **sélecteurs adjacents** contiennent des noms de balises séparés par le caractère **+**. Ils permettent de sélectionner une balise lorsqu'elle est immédiatement suivie d'une autre. Dans l'exemple suivant, le navigateur ajoute de l'espace aux instances **<h1>** de la page lorsqu'elles sont immédiatement suivies d'un titre de niveau 2, **<h2>**.

```
1 h1 + h2 { margin-top: +10mm; }
```

○ Les **pseudo-éléments** contiennent un nom de balise suivi de **:** et du nom d'un pseudo-élément (**:first-line**, **:first-letter**, **:before**, **:after**). Ils permettent de mettre en forme des parties de contenu qui sont impossibles à baliser. Selon l'exemple suivant, la première ligne des éléments **<p>** doit s'afficher en italique et dans une taille de police deux fois plus grosse :

```
1 p:first-line { font-size: 200%; font-style: italic; }
```

1. **Les classes.** Le langage CSS permet d'appliquer plusieurs styles à un même élément en attribuant une classe à chacun de ces éléments. Pour attribuer une classe à un élément, on utilise l'attribut **class="nom"**.

○ Les **classes régulières.** On veut, par exemple, présenter un article contenant deux paragraphes **<p>**, l'un contenant le résumé de l'article et l'autre l'article lui-même. On attribue au premier paragraphe la classe *résumé*, ainsi : **<p class="resume">** et au deuxième paragraphe la classe *article*, ainsi : **<p class="article">**. On définit ensuite le style de chacun de ces paragraphes de la manière suivante (le nom de la balise et celui de sa classe sont séparés par un **.**). Le résumé sera alors affiché en italiques, dans une taille de police inférieure et avec une marge de 0.5cm. Le contenu de l'article sera centré dans la page.

```
1 p.resume { font-style: italic; font-size: 80%; margin-left: 0.5cm; }
2 p.article { text-align: center; }
```

○ Les **classes génériques**. On peut aussi définir une classe et l'associer à plusieurs balises différentes. Le sélecteur de la règle CSS est alors composé d'un `.` suivi du nom de la classe. Par exemple, pour afficher les titres `<h1>` et les paragraphes `<p>` en italiques, on définit une classe italique à laquelle on associe la règle permettant d'afficher du texte en italiques. Puis, dans le code HTML, on attribue le nom de cette classe aux `<h1>` et aux `<p>` : `<p class="italique">` et `<h1 class="italique">`.

```
1 .italique { font-style: italic; }
```

○ Les **classes ID** permettent d'attribuer un identifiant unique à une balise donnée. Le sélecteur de la règle CSS est composé d'un caractère `#` suivi du nom de l'identifiant. Par exemple, pour afficher le contenu d'un `<p>` donné en jaune, on peut lui attribuer l'identifiant *jaune* (`<p id="jaune">`) puis définir la règle CSS suivante :

```
1 #jaune { color: yellow; }
```

○ Les **pseudo-classes** sont des classes dont les noms sont prédéfinis et qui n'ont pas besoin d'être précisées par l'attribut `class`. Elles permettent de contrôler le style de certaines balises selon leur état.

Voici les états les plus usités pour les éléments de type `<a>` :

`a:link {color: orange;}` : le texte du lien est affiché en orange tant qu'il n'a pas été visité

`a:active {color: red;}` : le texte du lien est affiché en rouge lorsque l'internaute clique dessus

`a:visited {color: blue;}` : le texte du lien est affiché en bleu s'il a été visité au moins une fois

`a:hover {font-weight: bold;}` : le texte du lien est affiché en caractères gras lorsque l'utilisateur le survole avec la souris

La pseudo-classe `:hover` peut également être utilisée avec les autres balises du HTML.

○ Pour utiliser une pseudo-classe avec un élément auquel vous avez attribué une classe, utilisez la syntaxe suivante : `balise.classe:pseudo-classe`. Par exemple, pour attribuer un style à un lien visité auquel vous avez attribué la classe normal : `a.normal:visited { ... }`.

1. **Les propriétés des styles**. Le langage CSS propose six catégories de propriétés de style : **polices, couleurs et arrière-plans, texte, boîtes et mise en pages, listes et classement des balises**. On donne une valeur à une propriété en faisant suivre son nom par le caractère `:` puis un espace et, enfin, une ou plusieurs valeurs séparées par des virgules. Lorsqu'une valeur de propriété contient des espaces, il convient de l'entourer de guillemets doubles (`"`) afin d'explicitier le fait qu'il s'agit d'une seule des valeurs de la propriété. Par exemple, pour définir la police d'affichage d'un titre :

```
1 h1 { font-family: Copperplate, Palatino, Georgia, "Times New Roman", serif; }
```

Les 5 principales catégories de valeurs de propriétés sont :

○ **les mots-clés** : sont des noms de valeurs prédéfinis par le langage CSS. Par exemple, la propriété `font-size`, qui permet de spécifier la taille de la police d'un texte, peut prendre les valeurs `xx-large` (très très grand), `x-large` (très grand), `large` (grand), `medium` (moyen), `small` (petit), `x-small` (très petit) ou `xx-small` (très très petit).

○ **les longueurs** : déterminent la taille d'une propriété, à l'aide d'un nombre qui peut parfois être de valeur décimale. Pour indiquer s'il faut ajouter ou retrancher de la valeur actuelle de la propriété, on ajoute le signe `+` ou `-` avant la valeur (sans espace entre les deux). La valeur est directement suivie (sans espace) de l'abréviation en deux lettres de l'unité de mesure. On distingue trois types d'unités de mesure :

- les unités de mesure *relative* : `em`, qui correspond à la largeur de la lettre `m` en base de casse, et `x-height` (abrégiée `ex`), qui correspond à la hauteur de la lettre `x` en bas de casse.

- les unités de mesure en *pixels* : `px`, correspondant à la taille d'un pixel.

- et les unités de mesure *absolue* : `in` pour les pouces, `cm` pour les centimètres, `mm` pour les millimètres, `pt` pour les points et `pc` pour les picas.

- **les pourcentages** : % comportent un signe + ou - (ajouter ou retrancher) car ils décrivent une mesure proportionnelle à un autre contenu.
- **les URL** sont spécifiées selon la syntaxe suivante : **url (chemin)** (sans espace entre le mot-clé **url** et la parenthèse ouvrante). La valeur du chemin est relative non pas au chemin du fichier HTML mais au chemin de la feuille de style CSS.
- **les couleurs** : indiquent la couleur d'une propriété. Elles peuvent prendre pour valeur un mot-clé (**blue**, **yellow**, **orange**, ...), une valeur hexadécimale de la forme RGB (*red*, *green*, *blue*) ou une valeur décimale RGB. Par exemple, le blanc est **white** (mot-clé), **#000000**; en hexadécimal et **rgb(255, 255, 255)** ou **rgb(100%, 100%, 100%)** en décimal. Les valeurs **rgb** sont un nombre compris entre 0 et 255 ou un pourcentage.

Sur le Web, les tailles de caractères peuvent changer d'un ordinateur à l'autre et d'un navigateur à l'autre. Par conséquent, si vous souhaitez que l'aspect global que vous avez défini pour votre page soit respecté sur la plupart des navigateurs, vous utiliserez des tailles de caractères relatives : les **em** et les % ou, encore mieux, les descriptions prédéfinies citées précédemment (**xx-large** à **xx-small**). Avec ces dernières, le navigateur ne laissera pas la police d'écriture devenir illisible. Imaginez par exemple que vous choisissiez une taille de caractères relative (à celle par défaut de votre ordinateur), petite mais lisible sur votre PC. Si l'ordinateur d'un utilisateur a une taille de police par défaut plus petite que la vôtre, votre texte risque de devenir beaucoup trop petit.

1. Applications.

Nous allons, à présent, définir les règles d'affichage de la page HTML dans le fichier `mes-styles.css`.

- Attribuez à la page entière (**<body>**)
 - une marge intérieure de 50px
 - une couleur d'arrière-plan (**background-color**) de valeur **#e8eae8**;
 - et une police de couleur (**color**) **#5d665b**;

Affichez la page `mise-en-forme.html` dans votre navigateur pour observer le résultat.

- Définissez la police des textes (**font-family**) en **Verdana, Geneva, Arial, sans-serif**. Ainsi, si la police *Verdana* n'est pas présente sur votre ordinateur, le navigateur utilisera *Geneva*, mais si cette dernière n'est pas, elle non plus, présente sur votre ordinateur, il utilisera *Arial*. Si aucune des trois polices n'est présente, il utilisera la police sans-serif qu'il connaît. Actualisez l'affichage de votre page pour observer le résultat.

- Fixez la taille de la police de caractères (**font-size**) à **small** et la hauteur des lignes (**line-height**), c'est-à-dire la hauteur des caractères et de l'espace supplémentaire au-dessus, à 180% de la taille de la police de caractères. Observez le résultat.

- Attribuez au premier **<h2>** une taille de police de 75% et au premier **<p>** une taille de police de 70%, relativement à la taille de police spécifiée précédemment. Puis, attribuez au second **<p>** une indentation (**text-indent**) de 3em.

- Attribuez au texte du titre **<h1>** la couleur **#ffffdd**; . Puis attribuez au texte des titres **<h2>** la couleur de police **#966b72**; et la famille de polices **Georgia, "Times New Roman", Times, serif**. Observez le résultat.

- Afin d'aérer votre page, insérez un **<p>** vide entre chaque titre **<h2>** et le paragraphe qui le suite. Observez le résultat.

Insérez ensuite dans ces **<p>** vides un espace (au clavier) et observez le résultat.

Insérez maintenant dans ces **<p>** vides un espace insécable ** **; et observez le résultat.

Explication. Un paragraphe vide n'apparaît pas à l'écran, il doit obligatoirement avoir un contenu. Un espace ne fonctionne que s'il se trouve entre deux caractères.

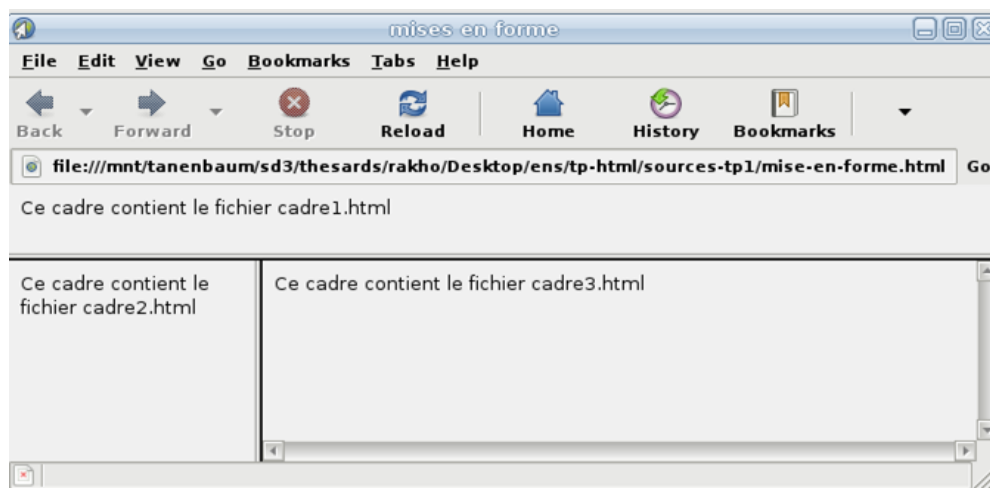
Astuce. On utilise l'espace insécable entre deux mots que l'on ne souhaite pas voir séparés lorsque la ligne se termine (des noms propres, par exemple).

- Encore quelques améliorations. Attribuez au `<body>` une marge supérieure (`margin-top`) et inférieure (`margin-bottom`) de 70px, une marge gauche (`margin-left`) de 120px et une marge droite (`margin-right`) de 50px.
- Amusez-vous avec les valeurs des styles de la page que vous venez de créer en modifiant les familles et tailles de polices, les marges, les interlignages, etc. Visionnez à chaque fois votre page dans différents navigateurs pour observer les différences de rendu. Notez les différences observées pour chacune des balises que contient votre page.

Exercice 8. Les cadres

1. Une page HTML peut être elle-même divisée en plusieurs pages HTML qu'on appelle **cadres**. Le contenu de chaque cadre est une page (un fichier) HTML distincte. Les balises permettant de créer des cadres sont `<frameset>` et `<frame>`. En voici un exemple et le résultat de l'affichage du fichier :

```
1 <html>
2   <head>
3     <title>mises en forme</title>
4     <meta http-equiv="content-type" content="text/html; charset=UTF-8" />
5     <link rel="stylesheet" type="text/css" href="mes-styles.css">
6   </head>
7   <frameset rows="25%,*">
8     <frame src="cadre1.html" name="cadre1">
9   </frame>
10  <frameset cols="25%,*">
11    <frame src="cadre2.html" name="cadre2">
12  </frame>
13    <frame src="cadre3.html" name="cadre3" scrolling="yes">
14  </frame>
15  </frameset>
16 </frameset>
17 </html>
```



2. Explications.

La balise `<frameset>`

- La balise `<frameset>` remplace la balise `<body>`. Elle définit un jeu de cadres et peut contenir des

cadres ou des jeux de cadres.

- Les attributs **border**, **bordercolor** et **frameborder** contrôlent les bordures des cadres ; l'attribut **framespacing** contrôle leur espacement.
- L'attribut **rows** définit le nombre et la taille des différents cadres **<frame>** ou jeux de cadres **<frameset>** que contient un **<frameset>**. Dans l'exemple précédent, le **<frameset>** principal contient un cadre dont la hauteur occupera 25% de la hauteur de la fenêtre et un jeu de cadres dont la hauteur occupera tout le reste (*) de la hauteur.
- L'attribut **cols** définit le nombre et la taille des cadres et jeux de cadres que contient un **<frameset>**. Ainsi, dans l'exemple précédent, le **<frameset>** de second niveau contient deux cadres dont l'un occupera 25% de la largeur du jeu de cadres et l'autre occupera tout le reste de la largeur.
- Les valeurs de **rows** et **cols** peuvent être des pourcentages ou des px. Au moins l'un des attributs **rows** et **cols** doit être spécifié pour chaque **<frameset>**.

La balise **<frame>**

- La balise **<frame>** ne peut être utilisée que dans le contexte d'un **<frameset>**. On l'utilise pour indiquer l'URL du document qui va s'afficher dans chaque cadre à l'aide de l'attribut **src**.
- L'attribut **name** de la balise **<frame>** permet de nommer un cadre de manière à l'utiliser comme cible d'un lien hypertexte **<a>**. Il suffit pour cela d'ajouter l'attribut **target** dans la balise **<a>** du lien source. Voici, par exemple, une ancre définie dans `cadre2.html` et qui permet d'afficher un document dans le cadre 3 :

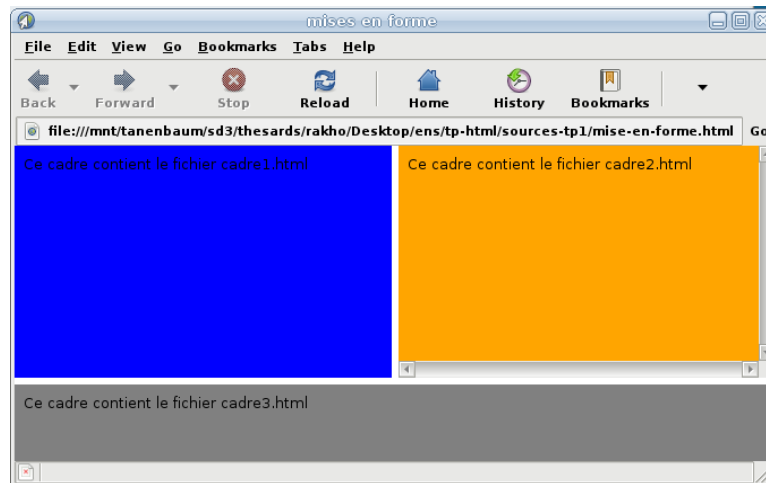
```
1 <a href="nouveau.html" target="cadre3">
```

Si l'internaute clique sur le lien, le document `nouveau.html` remplacera le contenu original du cadre 3.

- Afin d'empêcher le redimensionnement des différents cadres par l'utilisateur, on utilise l'attribut **noresize**.
- Par défaut, chaque cadre est doté de barres de défilement dès que leur contenu dépasse l'espace de la fenêtre. Afin de les supprimer, on utilise l'attribut **scrolling** avec pour valeur **no**.
- Les attributs **marginheight** et **marginwidth** permettent par ailleurs d'insérer une marge entre les bordures d'un cadre et son contenu. Une marge ne peut être inférieure à **1px** et ne peut avoir pour valeur une taille qui prend toute la place du contenu du cadre. Si les marges souhaitées ne conviennent pas au navigateur, ce dernier affiche le cadre du mieux possible, en utilisant ses styles par défaut.
- L'attribut **frameborder** permet de définir l'aspect de la bordure de chaque cadre. Ses valeurs possibles sont **yes** ou **1** pour encadrer le cadre et **no** ou **0** pour supprimer la bordure.
- Enfin, l'attribut **bordercolor** permet de modifier la couleur des bordures.

1. Applications.

Réalisez le **<frameset>** illustré par la copie d'écran ci-dessous.



- La zone 1 s'appelle (**name**) **cadre1**, la zone 2 est **cadre2** et la zone 3 est **cadre3**.
- La zone du bas occupe 60px de la hauteur de la page et la zone du haut occupe tout le reste. Sa hauteur ne doit pas être modifiable et son cadre ne doit pas être doté de barres de défilement.
- La zone 1 ne doit pas avoir de barres de défilement, la zone 2 peut en avoir.
- Les différents cadres doivent être séparés par un espace de 5px.
- Aucune des 3 zones ne doit avoir de bordures.
- Spécifiez les couleurs de fond des trois zones.
- Enfin, définissez deux liens dans la zone 1 qui permettront d'afficher des fichiers dans la zone 2.