

Notes de cours *Intégration web (version d'essai)*

Silvère Gangloff

27 mai 2020

1 L'intégration web et ses enjeux

Objectif (PPN) : *Connaître les standards (validation W3C, normes d'accessibilité), et savoir les appliquer.*

1.1 Définition

1.1.1 Développement "front-end" et "back-end"

La fonction de l'intégration dans le développement de sites web est réellement apparue en 2007, année marquée notamment par la commercialisation du premier *iPhone* et le développement des *smartphones* (téléphones intelligents), et donc de nouveaux enjeux pour la création de sites web (en particulier le "responsive web", c'est à dire l'adaptation à de multiples appareils), faisant apparaître une distinction entre développement "front-end" (tourné vers le client et l'aspect graphique, traité par le navigateur) et développement "back-end" (tourné vers la manipulation des données sur le serveur), correspondant à des spécialisations du métier de développeur.

1.1.2 Enjeux de l'intégration

L'intégration, tournée vers l'expérience de navigation de l'utilisateur d'un site web, consiste principalement en l'interconnection des divers contenus du site, développés de manière séparée, pour répondre à sa fonction. En pratique, cela implique notamment d'assurer la cohérence du site web et sa mise en forme, ainsi que l'évaluation de la pertinence du projet en tant qu'ensemble. L'évolution rapide des technologies du web implique également un besoin de robustesse du code aux changements associés, ainsi que la compatibilité avec les divers appareils et navigateurs, et l'omniprésence du web au niveau social implique d'assurer un référencement dans les moteurs de recherche, une bonne performance du site web à l'utilisation, ainsi que l'accessibilité du site à tout public.

1.1.3 Place dans la chaîne de production

Dans la chaîne de production d'un site web, l'intégration se situe entre le design (conception de l'aspect graphique du site, en pratique construction d'une maquette contenant l'information de la structure, de l'emplacement des divers

éléments (logo, menu, zones de contenu) et l'aspect visuel du site) et la construction du contenu (fonctions spécifiques). En pratique, l'intégrateur web assure ainsi la traduction de maquettes (produites par l'équipe graphique) en code, dans lequel il introduit le contenu produit par les programmeurs, en rend le site public.

1.1.4 Les langages utilisés

Le codage spécifique d'un site web se divise en deux parties : le codage de sa structure (disposition des divers éléments) et la forme dans laquelle apparaissent les éléments de cette structure. Depuis la quatrième version du HTML, la structure est codée en langage HTML, que l'on introduit dans la Section 2, la seconde en langage CSS, introduit dans la Section 3, dans des feuilles de style séparées, ainsi que JavaScript (aspect dynamique, non abordé dans le cadre de ce cours).

1.2 Standards

Depuis sa création par Tim Berners-Lee en 1991, le web a connu un essor fulgurant et le nombre de sites a explosé dans les années qui ont suivi. Alors que le langage HTML, créé également par Tim Berners-Lee, se transforme et connaît plusieurs versions successives, les balises ne sont pas créées de manière centralisée : en particulier, des balises sont fréquemment créées par les navigateurs Netscape et Microsoft Internet Explorer, donnent lieu à des bugs fréquents et des différences d'affichage selon les plateformes. De ce développement sauvage naît la nécessité de "surveiller" le web, et définir des standards, de manière à définir un langage utilisé de manière commune. C'est dans ce but que Tim Berners-Lee crée en 1994 le *World wide web consortium* (W3C), organisme à but non lucratif, aujourd'hui implanté dans de nombreux pays. Les enjeux de la standardisation sont multiples : en particulier permettre la compatibilité entre navigateurs et supports, accessibilité aux personnes handicapées, lisibilité du code, poids des pages, référencement plus efficace, pérennité des contenus. Aujourd'hui, il existe de nombreux autres organismes de standardisation, comme l'IETF (Internet Engineering Task Force), ou l'ISO (Organisation internationale de normalisation).

1.2.1 Recommendations

Le W3C agit pour la standardisation du web en produisant des recommandations, en particulier d'éléments de code HTML, CSS et Javascript. Si ces recommandations peuvent devenir des normes, elles ne sont pas imposées. Cependant il est en général dans l'intérêt du propriétaire du site web (surtout quand l'intérêt de ce site est de nature commerciale) de suivre ces recommandations (utilisation sur tout support et navigateur, accessibilité). De plus, un site web qui ne correspond pas aux standards peut être sujet à des effets négatifs, de distortion de la présentation, lenteur du chargement du site, etc.

En pratique, le site web du W3C fournit une liste d'éléments de code, avec leur description, ainsi que leur statut par rapport au standard (notamment en cours d'évaluation ou recommandé par le W3C). Avant d'utiliser un élément de code dans la conception d'un site web, il convient donc de consulter cette liste,

que ce soit pour sa fonction, les détails de son utilisation ou pour vérifier qu'il est en accord avec les standards.

Pour vérifier la compatibilité d'un document HTML avec le standard du W3C, au cours ou à la fin de sa conception, il existe un vérificateur fourni par le W3C que l'on peut trouver [ici](#). Il existe aussi un vérificateur pour les feuilles de style, que l'on peut trouver [ici](#).

1.2.2 Accessibilité

Un autre enjeu de la standardisation est l'accessibilité, en particulier pour les personnes handicapées, des divers contenus de sites web. Historiquement, la préoccupation pour l'accessibilité provient d'organisations telles que le W3C, qui crée en 1996 l'initiative WI (Web accessibility initiative), regroupant des organisations issues de l'industrie, des organismes pour personnes handicapées, organismes de recherche et gouvernementaux, pour proposer des solutions techniques pour rendre le web accessible aux personnes handicapées (que ce soit un handicap auditif, cognitif, neurologique, physique, verbal ou visuel), et de manière plus générale à toute personne pouvant se trouver en situation de handicap temporaire (bras cassé) ou situationnel (être en plein soleil ou dans un environnement où elles ne peuvent pas écouter l'audio), ou possédant une connexion lente, etc. Rendre accessible signifie plus précisément assurer que les personnes peuvent percevoir, comprendre, naviguer et interagir avec le web, et contribuer sur le web. En particulier, ces enjeux sont directement en lien avec la fonction de l'intégration web.

Dans le domaine public, plusieurs états et organismes étatiques (notamment les États unis et l'Union européenne) ont adopté des lois obligeant les instances publiques à respecter des règles d'accessibilité. En France, par exemple, l'article 47 de la loi du 11 février 2005 n°2005-102 impose la mise en conformité des sites internet du secteur public aux normes internationales d'accessibilité (cette loi a subi certaines évolutions depuis). Dans le domaine privé, aucune norme d'accessibilité n'est imposée. Certains organismes de standardisation, comme le W3C, fournissent des *référentiels* indiquant des listes de critères à respecter pour rendre un site web accessible (par exemple attacher à chaque image ou vidéo un texte alternatif qui donne une description de son contenu, permettant ainsi d'avoir accès à ce contenu par un autre moyen). Par exemple, on peut consulter les référentiels Accessiweb ([ici](#)), créés par l'association *BrailleNet*, association créée en 1997, et aujourd'hui intégrée au W3C. Accessiweb propose également un label qui mesure la conformité des sites web aux standards d'accessibilité du W3C/WAI.

2 Structure d'une page web : le langage HTML

Objectif (PPN) : *Savoir construire un document HTML et utiliser les commandes élémentaires (balises) de ce langage.*

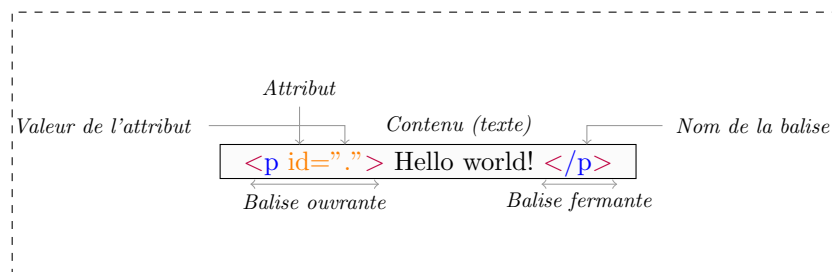
2.1 Un langage de description à balises

Le langage **HTML**, diminution de *Hypertext Markup Language*, est un langage d'enrichissement d'information textuelle, c'est à dire permettant l'ajout

d'information à un texte brut concernant la structure du texte tel qu'il apparaît dans un navigateur web (notamment l'organisation en sections, ou la disposition des divers éléments de contenu : images, menu, etc.). Le terme *markup* réfère au fait qu'un document écrit en HTML est composé du texte lui-même et de marques structurales, et le terme *Hypertext* à la possibilité de créer des liens (de référence par exemple) entre des parties du texte. C'est un langage de description, qui modifie la structure de présentation des données, par opposition à langage de programmation, qui transforme les données elles-mêmes.

Il est adapté à la structuration de textes par l'utilisation de balises, c'est à dire que les informations structurales sur le texte sont encodées dans des balises entourant la ou les parties du texte qui sont concernées par ces informations.

Les balises interviennent dans le texte la plupart du temps sous la forme suivante :



Il existe certaines exceptions à cette règle, comme la balise `
`, permettant de sauter une ligne, ou encore la DTD (voir Section 2.2). Ces balises sont écrites seules, et sont appelées balises **orphelines**.

Une balise peut avoir plusieurs attributs, qui sont simplement apposés et séparés d'un simple espace :

$$att_1 = "val_1" \quad att_2 = "val_2" \quad \dots \quad att_n = "val_n".$$

Les balises peuvent être emboîtées, par exemple

`<p> Texte </p>`,

mais pas croisées :

`<p> Texte </p> `.

On appelle indifféremment élément le nom de la balise, ainsi que la partie du document comprise entre une balise ouvrante et fermante correspondant (balises incluses).

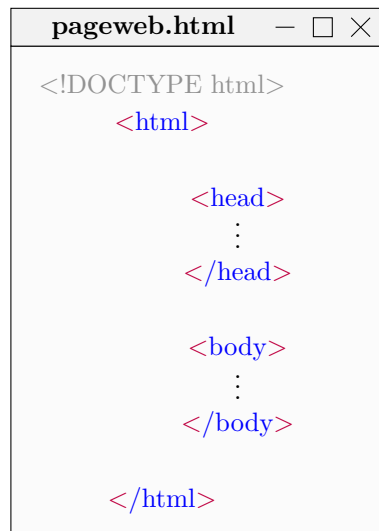
Remarque : il existe cinq versions du langage HTML, la dernière, nommée simplement HTML5, a été finalisée en 2014, et constitue aujourd'hui un standard. La suite de ce cours utilise cette dernière version.

2.2 Éléments *racines* d'un document HTML

Un site web est un ensemble de pages web reliées entre elles par des liens hypertexte (permettant de naviguer d'une page à une autre, mais aussi au sein d'une même page). La construction d'un site web commence donc par la construction des diverses pages web qui le composent.

Créer une page web : Une page web est l'interprétation d'un document texte écrit en langage HTML, dont l'extension est .html, ou .htm. Pour construire un tel document, il suffit de créer un fichier .txt puis de changer l'extension .txt par l'extension .html (par exemple pageweb.html). Ensuite ouvrir le fichier .html avec un logiciel de traitement de texte (par exemple bloc-notes sous Windows).

Éléments racines : On commence par écrire les éléments de structure suivants, dits éléments racines :



```
pageweb.html  -  □  ×
<!DOCTYPE html>
<html>

    <head>
        ⋮
    </head>

    <body>
        ⋮
    </body>

</html>
```

Analyse des éléments racines :

1. La balise `<!DOCTYPE html>` déclare le document comme écrit en HTML. La raison de la présence de cette balise est que le HTML est un cas particulier de langage SGML, diminution de *Standard Generalized Markup Language*. Ces langages respectent une norme commune, séparant :
 - (a) la **structure** du document, déterminée par un langage particulier (par exemple le HTML), correspondant à un type de document défini dans le document par une DTD, pour *Document type definition*, dont la commande apparaît en début de document.
 - (b) la **mise en forme**, traitée par des feuilles de style, écrites en CSS (voir Section 3).
 - (c) les **données**, encadrées par des balises.

2. Les balises `<html>` et `</html>` ouvrent et ferment le document. La balise ouvrante peut avoir l'attribut `lang` (par exemple `lang="fr"` pour le français) qui détermine la langue utilisée dans le document. Elle permet également d'assurer que les métadonnées (voir point suivant) sont énoncées correctement.
3. Les balises `<head>` et `</head>` encadrent des métadonnées, c'est à dire des informations portant sur le document lui-même, et d'autres informations qui ne portent pas sur le contenu du document. Les principaux éléments que l'on retrouve dans l'en-tête sont les suivants :
 - (a) **Le titre** (obligatoire) : il est encadré par les balises `<title>` et `</title>`. Il apparaît dans la barre de titres du navigateur web, et permet d'identifier le document (par exemple lorsque le navigateur affiche un accès rapide aux documents favoris).
 - (b) **Autres données** : par exemple l'auteur ou des mots-clés (pour le référencement), qui prennent la forme suivante :

`<meta name = "." content = ".">`,

où le nom est celui de la métadonnée, par exemple le nom de l'auteur (par exemple `name="author"`) et le contenu est la valeur de cette métadonnée.

On retrouve également des informations sur les feuilles de style CSS utilisées (voir Section 3).

4. Les balises `<body>` et `</body>` encadrent le contenu du document, c'est à dire les données et leur structure.

Dans toutes les parties des documents, on peut écrire un commentaire (qui ne s'affiche pas dans la page web), avec la convention suivante :

`<!-- Commentaire -->`.

Dans la suite de cette partie, on présente les balises élémentaires utiles pour la structuration du contenu : les balises proprement de structure (découpage en parties, listes, tableaux), les balises de style (taille de police, gras, italique, etc.), les hyperliens (vers des éléments du contenu de la page ou vers d'autres pages web), et les entités HTML (permettant d'écrire des symboles qui ne figurent pas sur le clavier).

2.3 Document object model

Le *Document object model* (DOM) d'un document HTML est une représentation de sa structure, standardisée par le W3C, qui permet notamment à un script d'agir sur cette structure. La définition de certains sélecteurs [Section ??] s'appuie sur cette représentation.

Étant donné un document HTML, son DOM est un arbre orienté dont les sommets sont les éléments, les attributs, et parties de texte contenues dans des éléments, reliés selon les règles suivantes :

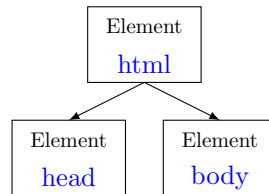
- Il existe une flèche d'un élément vers un autre lorsque le second est immédiatement contenu dans le premier (sans intermédiaire).

- Un attribut est relié à son élément par une arrête.
- Une partie de texte est reliée par une arrête à l'élément qui le contient immédiatement.

Étant donné un élément, les éléments vers lesquels pointe une flèche partant de celui-ci sont ses **éléments fils**. Ces éléments sont appelés **frères**. Pour deux éléments frères E_1, E_2 , on écrit $E_1 \geq E_2$ quand l'élément E_2 intervient après E_1 dans le document.

En pratique, l'arbre est représenté de manière descendante, en plaçant en haut de l'arbre le seul élément qui n'est le fils d'aucun autre élément, c'est à dire html, puis ses éléments fils dans la ligne suivante, puis leurs éléments fils dans la suivante, etc. Des éléments frères $E_1 \geq \dots \geq E_k$ sont représentés de E_1 à E_k de la gauche vers la droite. Pour tout j , le frère E_{j+1} de E_j est appelé son frère direct.

Par exemple, le DOM du document représenté sur la Figure 2.2 est le suivant :



2.4 Balises de structure

2.4.1 Le découpage en parties

Titres des parties : Les balises `<hi>`, pour $i = 1, 2, \dots$ ou 6 permettant de découper le document en parties et sous-parties, de manière hiérarchique. Pour un certain i , les balises `<hi>` et `</hi>` encadrent le titre d'une partie ou sous-partie. Une partie dont le titre est encadré par les balises `<h6>` et `</h6>` est incluse dans une partie dont le titre est encadré par les balises `<h5>` et `</h5>`, etc. Le contenu spécifique de chaque partie figure entre les titres encadrés par les balises.

Par exemple :

```
pageweb.html
<!DOCTYPE html>
<html>

  <head>
    <title> Poisson - Wikipedia </title>
  </head>

  <body>
    <h1> Définition et classification </h1>
    <h2> Étymologie </h2>
    <h2> Définition </h2>
    Le terme « poisson » est plus précisément
    employé pour désigner les crâniates non tétrapodes,
    c'est-à-dire des animaux ...
    <h2> Terminologie </h2>
  </body>
</html>
```

Découpage dans une partie : Dans une partie, le texte est usuellement découpé en paragraphes. L'élément `p` permet ainsi de former un paragraphe en groupant une partie du texte et en imposant un saut de ligne après cette partie. Un saut de ligne peut être aussi imposé avec la balise orpheline `
`. La balise orpheline `<hr>` permet de générer une ligne horizontale permettant de matérialiser une séparation thématique au sein d'une partie, entre deux paragraphes par exemple.

2.4.2 Agencements rigides

Pour agencer les données spatialement dans une partie, à part les paragraphes, il existe plusieurs types de formats préfixés, tels que les listes et les tableaux.

Listes : une balise de liste permet d'ordonner des parties de texte en colonne en identifiant chacune de ces parties. On peut créer ainsi une liste non-numérotée, pour laquelle les symboles identifiant ces parties sont uniformes, ou une liste numérotée, pour laquelle ces symboles sont des nombres ordonnés dans l'ordre naturel 1, 2, 3, Pour les listes non-numérotées, l'élément est `ul` (`un`ordered `list`). Cet élément contient des éléments `li` concaténés, qui sont les éléments de la liste. Par exemple :

Ils ont un rôle fondamental pour les humains : `` `` en tant que nourriture, les poissons sont partout dans le monde ; qu'ils soient pêchés dans la nature ou élevés en pisciculture `` `` ils sont aussi exploités à des fins récréatives, avec la pêche et l'aquariophilie, et sont parfois exposés dans de grands aquariums publics `` `` ...

Pour les listes numérotées, le principe est le même, excepté que l'élément `ul` est remplacé par `ol` (ordered list).

Tableaux : On peut créer un tableau en utilisant l'élément `table`. Cet élément contient les lignes du tableau, encadrées par les balises `<tr>` et `</tr>` (table row). A l'intérieur de ces balises, on concatène des cellules, encadrées par des balises `<td>` et `</td>` (table data), ou `<th>` et `</th>` (table head) pour des cellules d'en-tête. Le nombre de colonnes du tableau est alors le nombre maximal de cellules que l'on a inclut dans une ligne. Par exemple :

```
<table> <tr> <td> Texte 1 </td> <td> Texte 2 </td> </tr> <tr> <td>
    Texte 3 </td> <td> Texte 4 </td> </tr> </table>
```

Texte préformaté. Cette structure est équivalente au paragraphe, excepté que les distances entre les caractères tels qu'ils sont écrits entre les balises, ainsi que les sauts de ligne, sont préservés. Remplacer simplement l'élément `p` par `pre`.

2.5 Changer le style du texte

Police, taille des caractères, couleur, alignement : Pour changer la police ou la taille des caractères, on utilise un attribut de l'élément `p`, ou de l'élément `h1` si l'on veut modifier le titre d'une partie. Le nom de l'attribut est toujours `style`, et sa valeur dépend du changement que l'on veut opérer. Pour changer la police, la valeur sera `font-family :*`, où le symbole `*` est remplacé par le nom de la police, par exemple :

```
< p style="font-family :verdana" >.
```

Pour changer la taille de la police, la valeur de l'attribut est `font-size :*px`, où `*` est la taille, `px` signifiant "pixels", par exemple :

```
< p style="font-size :11px" >
```

On peut aussi changer la couleur du texte en utilisant la valeur `color :*`, où le symbole `*` est remplacé par le nom de la couleur, par exemple :

```
< p style="color :red" >.
```

On peut également aligner le texte avec la valeur `text-align :*`, et `*` égal à `right`, `center` ou `left`, changer la couleur de l'arrière plan avec la valeur `background-color :*`, où `*` est le nom d'une couleur.

Remarque : la valeur de l'attribut `style` est écrit en CSS, qui est utilisé pour l'écriture des feuilles de style (voir Section ??) et pour la mise en forme en général.

Styles préfixés : Il existe des balises permettant de changer le style du texte pour certains usages. L'élément `code` permettent d'afficher le texte qu'elles encadrent en fixant une police particulière à chasse fixe (c'est à dire telle que tous les caractères ont la même largeur), pour écrire un bout de code informatique.

Pour afficher une citation, on utilise de manière équivalente `cite`, et pour afficher un texte produit par un programme `samp`.

Mise en valeur : pour mettre en valeur le texte, on peut le mettre en gras, en encadrant avec les balises `` et `` (ou `` et ``), en italique avec `` et `` (ou `<i>` et `</i>`), ou le surligner avec `<u>` et `</u>`.

Remarque : la différence entre `b` et `strong`, ainsi qu'entre `i` et `em` est de nature sémantique : les premiers contiennent une information sur l'importance du texte encadré (qui peut être retranscrite aux personnes handicapées), mais non les seconds.

2.6 Liens hypertexte

Les balises : Les liens hypertexte sont des liens permettant de naviguer d'un endroit d'une page web vers un autre endroit de la même page, vers une autre page web, locale ou externe, ou alors vers un endroit d'une autre page. Pour créer un tel lien, on utilise l'élément `a` avec l'attribut `href`. Le texte encadré par les balises apparaît dans la page web comme un lien cliquable qui affiche l'endroit cible dans la page web ou alors ouvre la page web pointée par le lien. La valeur de l'attribut est alors l'adresse du lien, définie de manière spécifique pour chacun des cas évoqués ci-dessus :

1. **Vers une page web externe :** La balise ouvrante s'écrit sous la forme ``, où l'adresse URL commence par "http://www".
2. **Vers une page web locale :** Cette balise s'écrit sous la forme ``, où l'adresse URL est l'adresse locale, qui peut être l'adresse relative au dossier dans lequel se situe le fichier source (celui où est situé le document HTML que l'on écrit), ou l'adresse absolue. Dans le premier cas, si la page cible "page-cible.html" se trouve dans le sous-dossier dossier_n, qui se trouve dans le dossier_{n-1}, etc, l'adresse relative est "dossier₁ / ... /dossier_n/page-cible.html". Par exemple, si la page cible se nomme "poissons.html" et se trouve dans le sous-dossier "annexe", l'adresse relative est "annexe/poissons.html". Dans le second cas, l'adresse absolue est l'adresse locale complète de la page cible, qui s'écrit sous la forme "file:///dossier₁ / ... /dossier_n/ page-cible.html".
3. **Vers un endroit de la même page :** Pour créer un lien vers un endroit de la même page, on utilise une **ancree**, c'est à dire un attribut de la forme `id="ancree"`, qui peut se placer dans n'importe quelle balise. La balise ouvrante du lien est alors ``, et le lien mène à l'endroit où s'affiche le texte encadré par la balise.
4. **Vers un endroit d'une autre page :** Pour un lien vers un endroit d'une autre page, le procédé est similaire, excepté que la valeur de l'attribut href dans la balise ouvrante du lien est écrit sous la forme `adresse#ancree`, où `adresse` est l'adresse locale ou externe de la page web cible.

Fenêtre cible : Il est possible d'ajouter un attribut `target="."` dans la balise `<a>`, qui peut prendre deux valeurs non obsolètes en HTML5 : `"_blank"` et `"_self"`. La première permet d'ouvrir le lien dans une nouvelle fenêtre (ou onglet), la seconde dans la même fenêtre.

2.7 Entités HTML :

Il est possible d'écrire des caractères spéciaux en HTML (par exemple des symboles mathématiques $\sqrt{\quad}$ ou des lettres grecques α, β, \dots), qui ne sont notamment pas accessible au clavier.

On peut utiliser un élément `meta`, à placer dans l'en-tête, qui autorise l'utilisation d'une certaine catégorie de caractères, par exemple

```
<meta http-equiv="content-type" content="text/html; charset=ISO-8859-1">
```

pour les caractères des langues d'Europe de l'ouest (latin1), ou

```
<meta http-equiv="content-type" content="text/html; charset=UTF-8">
```

pour les caractères UNICODE.

On peut aussi utiliser des entités HTML, c'est à dire des séquences de caractères commençant par "&" et se terminant par ";", qui codent des caractères spéciaux. L'intérêt de ce code est qu'il est invariant par changement de DTD (document type definition). On peut trouver ces séquences dans un table fournie notamment par W3C [ici](#).

3 Mise en forme : feuilles de style CSS

Objectif (PPN) : *Savoir construire une feuille de style CSS et l'utiliser pour une ou plusieurs pages web.*

3.1 Structure d'une feuille de style

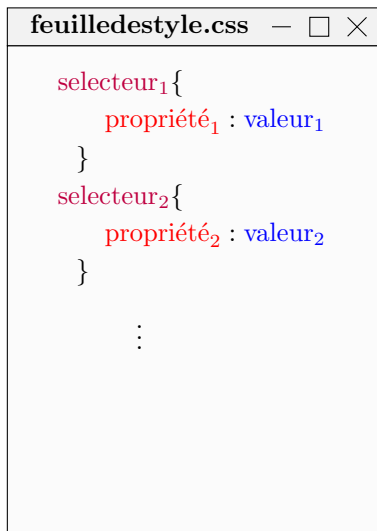
Créer une feuille de style et la relier à la page web : Alors que les balises du langage HTML déterminent la structure de la page web, la mise en forme (en particulier la couleur du texte, du fond, la police et la taille des caractères) est gérée de manière séparée par des feuilles de style, écrites en langage CSS. Ces feuilles de style, tout comme les documents html, sont écrites à l'aide d'un simple logiciel de traitement de texte. Pour créer une feuille de style, simplement créer un fichier ".txt", puis changer l'extension en ".css". Pour appliquer une feuille de style à une page web, on utilise la balise orpheline `<link>`, placée dans l'élément `head` avec les attributs `rel="stylesheet"`, `type="text/css"` `href="*.css"`, où le symbole * est remplacé par le nom du document .css. Par exemple :

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

Principe : Une feuille de style contient une liste de propriétés (par exemple police ou taille des caractères) auxquelles sont attribuées des valeurs (par exemple "New times roman" ou "10px") et une information sur le ou les endroits où s'appliquent ces propriétés dans le document .html. Toutes les attributions de ce type ont la structure suivante :

```
selecteur { propriété : valeur ; }
```

Dans le document .css, ces attributions sont simplement séparées par des sauts à la ligne :



Remarque : CSS est la diminution de *Cascading Style Sheet*. Le terme de cascade vient du système de gestion des conflits d’attributions de style (lorsque des propriétés différentes sont attribuées à un même élément) par des priorités (que l’on expose dans la Section 3.7).

Remarque : pour raccourcir le document .css, on peut rassembler les sélecteurs pour lesquels s’appliquent les mêmes propriétés avec les mêmes valeurs, selon le schéma général suivant :

```
selecteur1, selecteur2, ... { propriété1 : valeur11 valeur21 ...; propriété2 : valeur12  
valeur21 ...; ... }
```

3.2 Les propriétés et leurs valeurs

On peut trouver un index des propriétés CSS sur le site du W3C, [ici](#). La couleur de la propriété désigne son statut par rapport au standard du W3C. En cliquant sur le nom de la propriété, on peut avoir accès à certaines informations sur celle-ci, en particulier les valeurs possibles, et en cliquant sur le nom d’une valeur, à une description de l’effet de l’attribution de cette valeur. Autre information importante : le ou les endroits où une propriété peut être appliquée.

Exemples de propriétés élémentaires, recommandées par le W3C :

- **font-family** : pour le type de police. Le lien vers le site du W3C ci-dessus fournit une liste de familles génériques (par exemple Times New Roman, Bodoni, etc.)
- **font-size** : spécifie la taille des caractères, soit de manière absolue (par exemple small, medium, large, etc.), en nombre de pixels (exemple : 16px) ou en pourcentage (12pt) de la taille par défaut (ou de la taille de l’élément parent dans la hiérarchie des parties).

- **font-style** : normal, italic, oblique
- **font-weight** : largeur des caractères. Par exemple : normal, bold, lighter, etc.
- **border** : crée une bordure autour du contenu en spécifiant le style, la couleur, etc.
- **width** : largeur du contenu (par exemple un paragraphe), exprimée en pourcentage, pixels, ou centimètres (cm).
- **color** et **background-color** : couleurs du texte et de l'arrière-plan.

3.3 Sélecteurs

3.3.1 Sélecteurs élémentaires

Sélecteurs : Un sélecteur est une partie du contenu sur lequel s'appliquent certaines propriétés. Les sélecteurs élémentaires incluent les sélecteurs **E**, où **E** est un élément HTML, signifiant que les propriétés spécifiées s'appliquent à tout contenu qui se situe dans un élément **E**, et le sélecteur *****, qui signifie que les propriétés spécifiées s'appliquent partout. Dans la suite, le symbole ***** est utilisé comme un nom de balise qui signifie "n'importe quelle balise".

Groupements : Pour appliquer un style, on dispose d'éléments qui permettent de désigner des groupements de contenu sur lesquels s'appliquent des propriétés. Il s'agit des éléments **span**, ainsi que **div**. Les premières encadrent des parties de contenu qui apparaissent en ligne (comme une phrase), et les secondes pour des parties de contenu qui apparaissent en colonne (par bloc, comme plusieurs paragraphes). La différence entre **p** et **div** est que le paragraphe désigne un élément de structure de la page web, en interdisant notamment d'inclure des paragraphes dans un paragraphe, ce qui n'est pas vrai pour **div**.

Il existe plusieurs types de sélecteurs, s'appuyant sur différents éléments du corps du document .html : les sélecteurs par attribut, par classe, pseudo-éléments et pseudo-classes, que l'on présente dans les sections suivantes.

3.3.2 Sélection par attribut

Un sélecteur par attribut fait référence à un attribut d'une balise **<E>** pour définir une zone de contenu, de plusieurs manières possibles :

- Sélecteurs **E[att]** (par exemple **p[lang]**), où "att" est le nom de l'attribut : le sélecteur fait référence à toute partie de contenu située dans un élément **E** dont l'attribut "att" figure dans la balise ouvrante **<E>**.
- Sélecteurs **E[att=val]** : contenu situé dans un élément **E** dont l'attribut "att" a pour valeur "val".
- Sélecteurs **E[att~="val"]** : contenu situé dans un élément **E** dont la valeur de l'attribut "att" est une liste de mots séparés par des espaces dont l'un d'eux est "val".
- Sélecteurs **E[att≡"pref"]** : contenu situé dans un élément **E** dont la valeur de l'attribut "att" a pour prefixe "pref".
- Sélecteurs **E[att\$="suff"]** : contenu situé dans un élément **E** dont la valeur de l'attribut "att" a pour suffixe "suff".
- Sélecteurs **E[att*="val"]** : contenu situé dans un élément **E** dont la valeur de l'attribut "att" contient "val".

3.3.3 Sélection par classe et identification

Dans toute balise, on peut placer les attributs `class="x"` (que l'on peut attribuer à plusieurs balise) ou `id="x"` (que l'on ne peut attribuer qu'à une seule balise), où `x` peut être remplacé par n'importe quelle valeur. Ces attributs permettent de spécifier la partie du contenu qui correspond à cette balise comme appartenant à la classe `x` dans le premier cas ou comme étant identifiée comme `x` dans le second. Les sélecteurs `E.x` et `E#x` réfèrent à l'ensemble des parties de contenu situées respectivement entre des balises de la classe `x` et entre les seules balises identifiées comme `x`.

3.4 Combinaisons de sélecteurs

Il est possible de produire des sélecteurs à partir de sélecteurs élémentaires, en utilisant certaines opérations sur ces sélecteurs, par exemple :

- Le sélecteur `sélecteur1sélecteur2` (où les deux sélecteurs sont séparés par exactement un espace) désigne tout élément sélectionné par `sélecteur2` qui se trouve dans un élément sélectionné par `sélecteur1`.
- Le sélecteur `sélecteur1 > sélecteur2` désigne tout élément sélectionné par `sélecteur2` qui est fils d'un élément sélectionné par `sélecteur1`.
- Le sélecteur `sélecteur1 + sélecteur2` désigne tout élément sélectionné par `sélecteur2` qui est le frère direct d'un élément sélectionné par `sélecteur1`.
-
- Le sélecteur `sélecteur1 ~ sélecteur2` désigne tout élément sélectionné par `sélecteur2` qui suit (dans l'ordre \geq) frère d'un élément sélectionné par `sélecteur1`.

3.5 Pseudo-classes et pseudo-éléments :

Pseudo-classes : Il est possible d'attribuer des propriétés à un ou des sélecteurs selon l'état dans lequel ils sont lors de la navigation dans la page web, notamment si l'on passe la souris sur l'élément (pseudo-classe `hover`), ou lorsqu'un lien a été visité (pseudo-classe `visited`) ou non (pseudo-classe `link`). La syntaxe est la suivante (elle s'adapte à des sélecteurs et propriétés multiples) :

`selecteur :pseudo-classe propriété : valeur ;`

Pseudo-élément : Il est aussi possible d'attribuer des propriétés à une sous-partie du contenu situé dans un élément, par exemple la première ligne (pseudo-élément `first-line`) ou la première lettre (pseudo-élément `first-letter`). La syntaxe est alors (elle s'adapte à des sélecteurs et propriétés multiples) :

`selecteur :pseudo-élément propriété : valeur ;`

3.6 Requêtes media

Il est possible d'appliquer une attribution de style quand l'appareil par lequel on consulte la page web est d'un certain type (par exemple ordinateur portable, tablette, ou téléphone). Pour cela on utilise une requête media dont la syntaxe est la suivante :

```
@media connect1 mediatype and (mediafeature1 connect2 mediafeature2) {
    selecteur propriété : valeur ; }
```

Les parties `connect1` et `connect2 mediafeature` peuvent être présentes ou non et remplacées respectivement par `not` ou `only` et par `and`, `or` ou `not`. Le code CSS s'applique alors seulement quand l'appareil correspond (resp. ne correspond pas si `connect1 = not`, resp. seulement si l'appareil correspond si `connect1 = only`) au type `mediatype` et possède le ou les caractéristiques `mediafeature1` et/ou/non `mediafeature2`.

Exemple : Ceci permet notamment d'appliquer une attribution de style si la taille de l'écran ne dépasse pas une certaine taille, avec `mediatype=screen` et `mediafeature1 = max-width : *px`, où le symbole `*` correspond à la taille de l'écran, en pixels.

3.7 Gestion des priorités

Les feuilles de styles qui agissent sur un document HTML peuvent provenir de sources différentes : auteur de la page web, son utilisation, ou le navigateur web. Certaines attributions de style à des sélecteurs peuvent alors entrer en conflit. Les règles de priorité (en cascade) permettent de gérer ce problème, et s'organisent en trois étapes décrites ci-dessous, suivant le principe que si deux propriétés sont attribuées au même élément avec deux valeurs différentes, la seconde valeur écrase la première (seule la dernière valeur est appliquée).

1. **Tri par media :** ne sont appliquées que les attributions de style qui correspondent au media actuel, les autres ne sont pas prises en compte.
2. **Tri par origine :** les attributions de style sont prise en compte dans l'ordre d'origine suivant :
 - (a) Attributions de style provenant du navigateur,
 - (b) de l'utilisateur (sans marque " !important"),
 - (c) de l'auteur (sans marque " !important"),
 - (d) de l'utilisateur, avec marque " !important"
 - (e) de l'auteur, avec marque " !important".

La marque " !important" est écrite dans la syntaxe d'un code CSS entre la valeur de la propriété et le symbole " ;".

3. **Tri par calcul de priorité des sélecteurs :** pour chacune de ces origines, les attributions de style sont classés par ordre croissant de *priorité*, qui est un nombre à trois chiffres (usage) calculé en fonction du sélecteur. Lorsque deux attributions ont la même priorité, l'ordre de prise en compte est l'ordre d'apparition dans la feuille de style.

Définition 1 La *priorité* d'une attribution de style est *abc*, où *a* est le nombre de *id* dans le sélecteur, *b* est le nombre de classes, pseudo-classes et attributs dans le sélecteur, et *c* est le nombre d'éléments dans le sélecteur.

4 Éléments multimedia

Objectif (PPN) : *Savoir intégrer des éléments multimedia à une page web.*

4.1 Images

Pour ajouter une image, on utilise la balise `` qui est une balise orpheline (et correspond donc à un élément vide) qui suit en général le schema suivant :

``

La balise contient toujours l'attribut `src`, ayant pour valeur un chemin pointant vers le fichier image, qui peut être une URL relative au dossier contenant le document html, ou une URL absolue, de la même manière que pour l'attribut `href` de la balise de lien hypertexte `<a>`.

Il est possible d'ajouter l'attribut `alt`, qui associe à l'image un texte alternatif, qui s'affiche quand il n'est pas possible d'afficher l'image (par exemple quand la balise contient une erreur, en particulier si aucune image ne correspond au nom de fichier écrit). Cela peut être utilisé pour des raisons d'accessibilité : le texte contenu dans l'attribut `alt` permet de donner une description de l'image à un utilisateur qui ne peut la voir, ou simplement en cas d'erreur d'écriture, pour retrouver l'image que l'on souhaite apparaître.

On peut aussi régler la largeur et la hauteur d'une image dans le document HTML, que l'on peut exprimer en pixels (px), pourcentage, ou en centimètres (cm), à l'aide des attributs respectifs `width` et `height`. Ces propriétés peuvent également être attribuées dans une feuille de style. également dans une feuille de style, il est possible en utilisant la propriété float avec valeur `left` (resp. `right`) d'aligner une image à gauche (resp. à droite) avec du texte à droite (resp. à gauche) comme sur le schema suivant :



4.2 Video et audio

4.2.1 Balises et attributs simples

Pour afficher une vidéo, on utilise l'élément video selon le schema suivant :

`<video src="url" controls> <p> Texte </p> </video>`

En principe, on peut supprimer l'élément `p`, qui permet de positionner le texte en dessous de la vidéo, comme une légende. Le texte qu'il contient joue le même rôle que le texte alternatif pour une image. Comme pour un fichier image, l'attribut `src` spécifie un chemin d'accès au fichier video. L'attribut `src` s'écrit sans valeur : c'est un attribut booléen (valeurs `"true"` et `"false"`), et sa simple présence signifie qu'il a valeur `"true"`. Il permet à l'utilisateur un contrôle sur la lecture du fichier vidéo : il offre notamment la possibilité d'initier la lecture, de la mettre en pause, de régler le volume, et de déplacer le curseur de lecture.

Remarque : Bien que nous ne l'abordions pas dans ce cours, il est possible en utilisant JavaScript de modifier l'apparence du lecteur vidéo.

Comme pour les images, il est possible de régler la largeur et la hauteur avec les attributs respectifs `width` et `height`.

L'inclusion d'un fichier audio est similaire à celle d'un fichier vidéo, excepté que l'élément `video` est remplacé par l'élément `audio`.

4.2.2 Video : gestion des formats

Les principaux formats actuels de fichiers vidéos sont les formats MP4 et WebM. Ils sont appelés formats conteneurs, du fait qu'ils contiennent plusieurs parties : une piste vidéo (sans le son), une piste sonore, et éventuellement des sous-titres et métadonnées. De plus, le format de ces parties est spécifié par le format conteneur. Ils sont utilisés pour compresser les parties qui composent le fichier vidéo (celles-ci étant très volumineuses). Un navigateur contient des codecs permettant de lire seulement certains formats conteneurs, mais tous les navigateurs ne contiennent pas les mêmes codecs.

Pour permettre à l'utilisateur de pouvoir lire le fichier vidéo quel que soit le navigateur qu'il utilise, une solution est d'utiliser des éléments `source` dans l'élément `video`, qui pointent vers des versions du fichier dont les formats couvrent les principaux formats utilisés (MP4 et WebM), selon le schéma suivant :

```
<video controls> <source src="fichier1.mp4" type="video/mp4"> <source  
src="fichier2.webm" type="video/webm"> <p>texte</p> </video>
```

Le navigateur cherche et lit le premier de ces fichiers dont le format lui est compatible.

4.2.3 Attributs de lecture (audio et video)

Pour les deux éléments audio et vidéo, il existe des attributs qui permettent de spécifier comment est lu le fichier correspondant à la valeur de l'attribut `src`. Tout comme l'attribut `controls`, ces attributs ne prennent pas de valeur. Leur simple présence déclenche la propriété correspondante :

1. Attribut `loop` : le fichier est lu à nouveau quand la lecture se termine, indéfiniment.
2. Attribut `muted` : le son est éteint à la lecture du fichier.
3. Attribut `autoplay` : la lecture démarre automatiquement au chargement de la page web.

4.3 Intégration de pages web

Il est également possible d'intégrer une page web dans une autre, avec l'élément `iframe`, selon le schéma suivant :

```
<iframe src="."> </iframe>
```

La valeur de l'attribut `src` peut-être un fichier `.html` ou bien une adresse URL d'une page web, qui peut être en particulier réduite à une image, une vidéo ou un fichier pdf. La page web s'affiche alors dans un cadre dont la largeur et la hauteur sont par défaut respectivement 300 pixels et 200 pixels. Il est possible de les modifier avec les attributs `width` et `height`.

Problèmes de sécurité : l'utilisation de l'élément `iframe` peut être risquée, par exemple lorsqu'une page web intégrée de cette manière contient un code malveillant (notamment permettant à son auteur un accès à des informations importantes via un formulaire). Il convient donc d'être prudent, notamment en intégrant seulement des pages web qui utilisent HTTPS et/ou d'utiliser l'attribut `sandbox`, qui écrit sans valeur permet de désactiver un certain nombre de fonctionnalités de la page web intégrée qui sont potentiellement dangereuses (par exemple l'utilisation de Javascript ou des liens vers d'autres contenus). Les valeurs possibles de `sandbox` permettent d'autoriser spécifiquement certaines fonctionnalités (par exemple `sandbox="allow-popups"` permet l'apparition de fenêtres pop-up, et `sandbox="scripts"` permet le fonctionnement de scripts). Il convient également de respecter la propriété intellectuelle : l'utilisation d'une page web (même d'une image) sans l'autorisation de son auteur est très sévèrement punie.

Certains sites web, comme Google Maps, proposent directement un élément `iframe` qui permet d'intégrer une page web ou une partie d'une page web (comme une carte), que l'on obtient en cliquant sur "partager". Pour une carte Google maps, il suffit ensuite de cliquer sur "intégrer une carte" pour obtenir le code.

4.4 Images vectorielles

Une image vectorielle, contrairement à une image matricielle qui est définie pixel par pixel, est définie par un ensemble de formules mathématiques qui permettent, à partir d'une largeur et d'une hauteur, de produire une version de l'image qui a cette largeur et cette hauteur. Le principal intérêt est qu'un changement d'échelle n'affecte pas la qualité de l'image, car l'image est reproduite à partir de la donnée de la nouvelle largeur et hauteur. Ainsi l'image n'apparaît pas "pixelisée" après le changement d'échelle. Pour cette raison, les graphistes utilisent le plus souvent des images vectorielles plutôt que matricielles.

4.4.1 L'élément `svg`

On peut coder une image vectorielle en utilisant l'élément `svg`. La balise ouvrante peut prendre certains attributs comme `width` ou `height`, définissant respectivement la largeur et la hauteur de l'image. On ajoute alors entre ces deux balises des éléments le plus souvent sous la forme

$$<\text{element } \text{attribut}_1 = \text{"valeur}_1" \dots \text{attribut}_m = \text{"valeur}_n" / >.$$

Ces éléments correspondent à des parties de l'image, dont la définition repose sur un système de coordonnées défini par l'élément `svg`. L'origine (0,0) de ce système de coordonnées est le coin haut-gauche de l'image. La coordonnée horizontale x croît de la gauche vers la droite, et la coordonnée verticale y croît du haut vers le bas.

4.4.2 Éléments `svg` simples

Par exemples, les éléments `svg` suivants permettent de dessiner des figures géométriques simples (par défaut, les coordonnées sont exprimées en pixels, mais peuvent être exprimées en *em* par exemple) :

1. **line** : `<line x1="." y1="." x2="." y2="." / >` : dessine un segment de droite qui relie le point de coordonnées `x1` et `y1` et le point de coordonnée `x2` et `y2`.
2. **polyline** : `< polyline points=".,. ,. ,. ,." / >` : dessine des segments de droites reliant successivement les points dont les coordonnées sont spécifiées (dans l'ordre) par la valeur de l'attribut `points`.
3. **rect** : `< rect width="." height="." / >` : dessine un rectangle d'une certaine largeur et d'une certaine hauteur.
4. **circle** : `< circle cx="." cy="." r="." / >` : dessine un cercle dont le centre a pour coordonnées `cx` et `cy`, de rayon `r`.
5. **ellipse** : `< ellipse cx="." cy="." rx="." ry="." / >` : similaire au cercle, excepté que le rayon est remplacé par les deux rayons de l'ellipse, `rx` et `ry`.

À chacun de ces élément on peut ajouter les attributs `stroke`, `stroke-width`, qui spécifient respectivement la couleur du trait et son épaisseur. Lorsque la figure délimite une zone de l'image, l'attribut `fill` spécifie la couleur de remplissage de cette zone, et `fill-opacity` spécifie l'opacité du remplissage, dont la valeur est un nombre entre 0 (transparent) et 1 (complètement opaque).

4.4.3 Groupements et sous-images

De la même manière que les éléments `div` et `span` en HTML, l'élément `g` permet de grouper des éléments SVG de manière à leur appliquer un même attribut de même valeur, ou un style commun. Avec l'attribut `transform`, il est possible d'appliquer une certaine transformation à un élément `g`. Par exemple, lorsque `transform` a pour valeur `"translate(.,.)"` le contenu de l'élément est traduit d'un vecteur dont les coordonnées sont spécifiées par `translate (coordonnée x, coordonnée y)`. Lorsque l'attribut a pour valeur `"scale(d1,d2)"`, le groupement est dilaté par des facteurs donnés pour chacune des coordonnées : la largeur de chacun des éléments est multipliée par `d1` et la hauteur par `d2`. Il est possible d'inclure plusieurs transformations dans la valeur de `transform`, en les séparant par un simple espace. Dans ce cas, les transformations sont appliquées successivement de la droite vers la gauche.

Il est également possible d'inclure un élément `svg` dans un autre, les coordonnées des éléments inclus dans l'élément `svg` fils sont relatives au système de coordonnées introduit par celui-ci.

4.4.4 Chemins

L'élément `path` permet de construire des chemins plus généraux (dont le code est cependant plus complexe) que les rectangles, cercles, ellipses et polygones. L'écriture la plus générale de cet élément est complexe, et je voudrais simplement l'introduire dans le cas des courbes de Bézier quadratique et cubique. Ces courbes sont générées respectivement à partir de trois et quatre points, selon une formule mathématique. Le premier et le dernier point sont respectivement le point de départ et d'arrivée du chemin. Le ou les autres points informent sur la manière dont le chemin s'infléchit, du point de départ au point d'arrivée.

Pour tracer une courbe de Bézier cubique, on écrit la balise ouvrante `path` de la manière suivante :

```
<path d="M .. C .. .. ." / >
```

Les points entre **M** et **C** sont remplacés par les coordonnées du point de départ. Après le **C**, les deux premiers .. représentent les coordonnées des deux points d'inflexion, séparées par la virgule. Le dernier .. représente les coordonnées du point d'arrivée.

Pour visualiser la courbe de Bézier produite par un ensemble de points, on peut trouver un outil utile [ici](#).

En outre, il est possible d'écrire une ligne de texte selon un chemin dans l'image, à l'aide de l'élément `textpath`, pour un chemin construit à l'aide de `path` incluant l'attribut `id="mypath"`, comme suit :

```
<textpath href="#mypath"> Texte </textpath>.
```

4.4.5 Inclusion dans `img`

Ainsi présenté, l'élément `svg` permet d'inclure directement l'image construite dans la page web en incluant simplement l'élément `svg` dans le document html. Si cela permet d'appliquer des scripts (Javascript), notamment pour créer des animations, ou bien des pseudo-classes, il peut être préférable quand on ne souhaite en appliquer, pour alléger le document, d'écrire le code d'une image dans un document séparé. Pour cela, il suffit de créer un document d'extension `.txt`, d'écrire l'élément `svg` dans ce document, puis de changer l'extension en `.svg`. Pour inclure l'image dans la page web, on utilise alors l'élément `img`, dont l'attribut `src` a pour valeur l'URL du fichier `.svg`.