Introduction

Ce document a pour objectif de présenter brièvement les tenants et aboutissants de la conception de pages Web.
Une bonne connaissance du HTML est requise.

Une page Web est une ressource

Une page Web diffuse une ou plusieurs informations sur le Net. Elles sont en général structurées.
Écrire une page Web peut sembler en apparence très simple. Les apparences sont parfois trompeuses.
Si l'on ne prend en compte que sa lisibilité depuis un navigateur précis par une personne « normale », cela peut être effectivement très simple. Mais cette ressource doit être aussi interopérable, indexable, imprimable... Il faut traduire la structure des informations puis la décrire, penser l'hypertextualité, mettre en forme...
Tout ce qui peut permettre à l'information d'être manipulée, transportée, re-diffusée sans perte devrait être mis œuvre dans la création d'une page Web.

Traiter les ressources

Souvent ces ressources durent et sont gérées par une ou plusieurs personnes. Le temps est un facteur important car les technologies évoluent et avec elles les méthodes de travail. Aussi pour faciliter le travail des uns envers les autres, permettre une maintenance et des mises jours raisonnables, des pratiques et outils existent.

Pour faciliter les relectures du code de la page :
Nommer les éléments de structuration de la page de manière explicite.
Appliquer et retrouver des styles, des scripts, etc. devrait être clair pour ceux qui n'ont pas écrit la page. Le nommage peut s'appuyer sur la fonctionnalité de l'élément.
Placer des commentaires pour guider et rendre compréhensible la démarche d'écriture.

L'engouement pour les outils de gestion de contenus en ligne n'est pas fortuite. Ils peuvent avoir quelques défauts mais s'appuient la plupart du temps sur une communauté d'utilisateurs et de développeurs qui rendent ces outils évolutifs et réactifs.
Ils ont l'avantage d'enregistrer les informations dans une base de données. L'information est indépendante de son support, ce qui peut permettre sa diffusion vers différents média. Les sauvegardes, maintenances et mises à jour sont facilitées : l'outil est commun à tous ceux qui participent à la vie de la ressource.

Rendre les ressources pérennes

La traduction de la structure des informations en HTML facilite son écriture en page Web.
Cette structuration est pérenne car a un sens. Les balises HTML ont été créées avec des fonctions spécifiques de description de contenu et non pour mettre en forme ce contenu.
Une page Web est en premier lieu un document texte, associé ou non à d'autres média. Pour qu'elle devienne un page Web, l'information écrite doit être déclarée comme telle. La version du langage HTML est à renseigner, le langage utilisé dans le document doit être déclaré. Des méta données seront aussi nécessaires, notamment celle de déclaration du type d'encodage. D'autres seront utiles à un grand nombre de « lecteurs » et d'outils de lecture.
Pour ce qui est de la ressource, à priori, elle a un titre et un seul. Elle peut être ensuite divisée en chapitres qui peuvent être titrés ou non. Ces chapitres peuvent être composés de paragraphes, de listes d'éléments, de citations, etc. Certains éléments, des mots ou groupes de mots, peuvent avoir une place importante dans un chapitre, ils seront alors mis en emphase.
Des listes complexes de données peuvent aussi être affichées. Les tableaux servent cet objectif.
On peut ajouter des divisions structurelles à la page Web : rarement seule, elle fait partie d'un ensemble de pages et les notions d'ergonomie et de navigation entrent en scène. Il faut « bien » alors les placer.
Ces divisions aideront aussi à mettre en forme.

Exemple de code d'une page « simple » :

```
1. <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
2. <html lang="fr">
3. <head>
     <title>Titre de la page</title>
4.   <meta http-equiv="Content-Type" content="text/html;
charset=utf-8" />
   </head>
   <body>
5.   <h1>Titre de la page / ressource</h1>
     <h2>Chapitre 1</h2>
     <p> Texte .... </p>
     <h3>Sous chapitre 1.1</h3>
     <p>... </p>
     <h2>Chapitre 2</h2>
     <p>... </p>
   </body>
6. </html>
```

1. La « grammaire » utilisée pour l'écriture de la page est définie ;
2. la langue est déclarée ;
3. les éléments non affichés dans la page et utiles à sa diffusion et présentation sont donnés et appelés ;
4. le type et l'encodage du document sont définis ;
5. la structure HTML des contenus reprend la hiérarchisation de l'information ;
6. la description de la ressource est terminée.

Exemple de code d'une page contenue dans un site :

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
<html lang="fr">
<head> ... </head>
<body>
     <div id="entete">
          <h1>Titre de la page / ressource</h1>
          <ul class="menu_navigation"> ... </ul>
     </div>
     <div id="contenus">
          <h2>Chapitre 1</h2>
          <p> Texte .... </p>
     </div>
</body>
</html>
```

Des divisions sont mises en place pour compartimenter les différents composants de la page : navigation, contenus, etc.

Contenu et contenant

L'habillage de la page, et plus généralement tout se qui se rapporte à de la mise en forme doit être séparé des contenus de la page. Les mises à jour graphiques sont facilitées et les contenus restent indépendants.

Les styles sont liés à la page et sont écrits dans un ficher tiers. La norme utilisée est CSS. Ils peuvent être appliqués par un ou des sélecteurs qui représentent des balises, classes ou identifiants et possèdent un grand nombre de propriétés de mises en forme, qu'elles soient d'apparence ou de positionnement.

Application d'un style par balise :

HTML
```
<h1>Titre de la page</h1>
```

CSS
```
h1 {
     font: bold 1.2em Arial, sans-serif ;
     text-align: center ;
}
```

Application d'un style par classe :

```HTML
<p class="notes">Texte...</p>
```

```CSS
.notes {
      margin : .2em 2em ;
      font-size : small ;
      padding-top : .5em ;
      border-top : 1px solid #CCC;
}
```

Application d'un style par identifiant :

```HTML
<div id="entete"> ... </div>
```

```CSS
#entete {
      text-align : center ;
      background : #F30;
      margin : 0 auto;
}
```

Héritage

Les propriétés d'apparence (et non de positionnement), définies pour un élément s'appliquent à tous les éléments contenus dans celui-ci. On parle de parents et enfants : **<body>** est l'enfant de l'élément **<html>** et le parent de tous les éléments qu'il contient. Appliquer un style de fonte (taille, police de caractères, graisse...) à **body** s'appliquera à tous ses éléments enfants et aux enfants de ces enfants. Cela évite des répétitions.

Il est aussi possible d'imbriquer les sélecteurs et leurs règles de mise en forme.
Cette classe ne sera appliquée qu'aux paragraphes attribués de la classe **notes** :

```CSS
p.notes {
      margin : .2em 2em ;
      font-size : small ;
      padding-top : .5em ;
      border-top : 1px solid #CCC;
}
```

Ici, seuls les éléments de liste contenus dans un élément ayant pour identifiant **entete** seront ciblés :

```CSS
#entete li {
      margin : .5em ;
      list-style-type : none ;
      display : inline ;
}
```

Des blocs, des boites et des lignes

Les balises HTML se divisent, en terme de positionnement (ou de rendu), en 2 types : les éléments de rendu « bloc » et ceux de rendu « en ligne ».
Par défaut, les éléments de rendu « blocs » prennent toute la largeur de la page, ce sont les balises de titre, de paragraphe, de listes... Les éléments « en ligne » s'insèrent dans des « blocs » pour les enrichir : lien, emphase, image...
Ces éléments, ou boites, se positionnent naturellement dans le « flux » : l'ordre dans lequel ils sont écrit dans le code détermine l'ordre d'affichage.
D'autres types de positionnements CSS permettent de changer l'ordre d'affichage du flux et les propriétés de rendu :

- ➔ positionnement relatif ;
- ➔ positionnement absolu ;
- ➔ positionnement fixe ;
- ➔ positionnement flottant.

Permettre l'accès aux ressources à tous

Il est nécessaire, pour tout *auteur* ou *éditeur* du Web, de tenter de se mettre à la place des *lecteurs*, les internautes. Ceux-ci, dans leur diversité n'ont pas tous les mêmes moyens d'accès aux informations présentes sur le Net. Que ce soit matériellement, physiquement ou même par simple connaissance. Quelques règles sont à respecter pour permettre à un maximum de « type de lecteurs » d'accéder aux informations des pages Web :

- **Images & animations.** Utiliser l'attribut `alt` pour décrire la fonction de chaque graphique.
  - image illustrative (sans information) : `alt=""` ;
  - image informative : `alt="information"` ;
  - image lien : `alt="fonction du lien"` ;
- **Images complexes, figures & diagrammes.** Les décrire dans la page ou avec l'attribut `longdesc` : `longdesc="description.html"`
- **Images cliquables.** Utiliser l'élément `map` et décrire les zones actives : attributs `alt` et `title` redondants dans les balises `area`
- **Multimédia.** Fournir légendes et transcriptions pour l'audio, et des descriptions pour les vidéos.
- **Liens hypertextes.** Utiliser des énoncés pertinents hors contexte. Par exemple, évitez « cliquer ici » ou « lire la suite ».
- **Organisation.** Utiliser des têtes de sections, des listes et une structure cohérente. Utiliser CSS si possible.
  - Structurer un document HTML avec des `h1`, `h2`, etc. de manière cohérente (à priori, une page ne doit contenir qu'un titre de niveau 1)
- **Scripts, applets & plug-ins.** Fournir une alternative quand le contenu actif est inaccessible ou non traité. Permettre la navigation au clavier.
- **Cadres.** Utiliser `noframes` et des intitulés utiles : attribut `name` pertinent de la balise `frame`.
- **Tableaux.** Faciliter la lecture ligne par ligne. Résumer :
  - attribut `summary=""` de la balise `table` s'il s'agit d'un tableau de mise en page ;
  - attribut `summary="description du tableau"` de la balise `table` s'il s'agit d'un tableau de données.
- **Vérifier votre travail.** Valider, utiliser les listes de contrôle, guides d'utilisation et les outils disponibles.