

## Cours HTML & CSS

## Table des matières

Cours HTML & CSS.....	1
Définition et utilisation du HTML et du CSS.....	4
HTML : le langage de structure.....	4
CSS : le langage de styles.....	5
Histoire et évolution de l'informatique et du web.....	5
Les environnements : local, préproduction et production.....	7
Choisir et installer un éditeur de texte.....	8
Préparer la structure d'un site local.....	8
Éléments, balises et attributs HTML.....	9
Structure minimale d'une page HTML.....	10
Indentation et lisibilité du code.....	11
Commentaires en HTML.....	11
Enregistrer et afficher une page HTML.....	12
Créer des liens hypertextes.....	13
Liens internes (ancres).....	13
Ajouter des images à une page.....	14
Comprendre les chemins relatifs et absolus.....	14
Les titres en HTML.....	15
Paragraphe et retours à la ligne.....	16
Séparer visuellement le contenu.....	16
Espaces et caractères spéciaux.....	17

Mise en forme du texte.....	18
Texte préformaté et citations.....	19
Les listes en HTML.....	20
Les listes de définitions.....	21
Structurer logiquement le contenu.....	21
Introduction au CSS.....	22
Les trois méthodes d'intégration du CSS.....	23
Feuille de style externe.....	23
c) Style en ligne.....	24
Comprendre la syntaxe CSS.....	24
Ajouter des commentaires dans une feuille de style.....	25

## Définition et utilisation du HTML et du CSS

Le HTML (HyperText Markup Language) et le CSS (Cascading Style Sheets) sont les deux piliers de toute page web. Le premier définit la structure et le contenu d'un document, tandis que le second en détermine l'apparence visuelle.

### HTML : le langage de structure

Le HTML est un langage de balisage créé en 1991. Il repose sur des éléments appelés balises, qui permettent d'indiquer au navigateur le rôle de chaque contenu : un titre, un paragraphe, une image, une liste, etc. Lorsqu'un utilisateur accède à une page web, son navigateur reçoit un fichier HTML depuis un serveur et l'interprète pour afficher le contenu à l'écran.

#### Bon à savoir

Un serveur est simplement un ordinateur connecté en permanence à Internet, chargé de stocker les fichiers du site et de les « servir » aux navigateurs lorsque ceux-ci en font la demande.

Voici un exemple minimaliste d'une page HTML complète et valide :

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Ma première page</title>
  </head>
  <body>
    <h1>Bienvenue sur mon site</h1>
    <p>Ceci est un paragraphe.</p>
```

```
</body>  
</html>
```

Les balises `<h1>` et `<p>` indiquent respectivement un titre principal et un paragraphe. Le navigateur sait alors comment interpréter et afficher ces contenus.

## CSS : le langage de styles

Le CSS a été créé en 1996 pour compléter le HTML. Là où le HTML structure la page, le CSS permet d'en personnaliser le rendu visuel : couleurs, tailles, marges, alignement, etc. Chaque règle CSS cible un ou plusieurs éléments HTML pour leur appliquer des propriétés de style.

```
h1 {  
  color: orange;  
  font-size: 24px;  
}  
  
p {  
  color: blue;  
  font-size: 16px;  
}
```

Grâce à ce code, le titre principal s'affiche en orange avec une taille de 24 pixels, et les paragraphes en bleu avec une taille de 16 pixels.

### ⚠ Erreur fréquente

Ne jamais utiliser le HTML pour mettre en forme le contenu. Le HTML sert à décrire le sens et la structure des informations, tandis que le CSS gère l'aspect visuel. Confondre les deux dégrade la qualité du code et le référencement du site.

## Histoire et évolution de l'informatique et du web

L'informatique est un domaine en constante évolution. Les progrès du matériel et des connexions Internet ont transformé le web en un environnement interactif et multimédia. Il y a 20 ans, les sites étaient simples, souvent composés uniquement de texte

et d'images légères. Aujourd'hui, les pages peuvent intégrer de la vidéo, de l'audio, des animations et des interactions complexes.

L'évolution du web est guidée par plusieurs organisations internationales qui définissent les standards et les bonnes pratiques des langages informatiques.

- W3C : World Wide Web Consortium – supervise HTML et CSS
- WHATWG : Web Hypertext Application Technology Working Group – promeut le HTML Living Standard
- ECMA : organisme responsable du langage JavaScript
- PHP Group et Python Software Foundation – maintiennent PHP et Python

Le W3C classe ses documents selon trois niveaux de maturité : 'Travail en cours', 'Candidat à la recommandation' et 'Recommandation officielle'. Seules les recommandations officielles sont considérées comme des standards pleinement reconnus.

#### Bon à savoir

Le WHATWG préconise une évolution continue du HTML appelée 'Living Standard'. Cela signifie que le langage s'améliore progressivement sans attendre une nouvelle version majeure.

Les versions HTML et CSS évoluent désormais par modules indépendants : chaque fonctionnalité (gestion des couleurs, typographie, disposition des éléments, etc.) possède sa propre spécification. Cette approche modulaire permet d'intégrer plus rapidement les nouveautés et d'éviter les retards liés à la validation d'une version complète du langage.

#### ⚠ Erreur fréquente

Certaines pratiques anciennes (balises obsolètes, attributs de mise en forme, anciens doctypes) ne sont plus reconnues par les

navigateurs modernes. Utiliser des éléments non standards peut provoquer des incohérences d'affichage.

Les développeurs doivent donc rester à jour sur les évolutions des langages, vérifier la compatibilité des nouvelles fonctionnalités avec les navigateurs, et se tenir informés des recommandations du W3C.

## **Les environnements : local, préproduction et production**

Dans le développement web, on distingue trois environnements de travail :

- Local : votre ordinateur personnel, où vous créez et testez vos fichiers HTML/CSS avant toute mise en ligne.
- Préproduction : un espace intermédiaire en ligne qui simule les conditions réelles du site, utile pour les tests collaboratifs.
- Production : le serveur public accessible aux internautes.
- 

### **Bon à savoir**

Travailler localement évite de modifier accidentellement le site en ligne et permet de tester rapidement les changements.

Les développeurs utilisent souvent un serveur local comme XAMPP, WampServer ou MAMP, qui permettent d'exécuter localement des sites dynamiques avec PHP et MySQL.

Une fois le site validé en local, il est transféré sur le serveur de préproduction, puis finalement sur le serveur de production via des outils comme FTP,SSH sur des plateformes d'hébergement (ex. OVH, Infomaniak, etc.).

## Choisir et installer un éditeur de texte

Le choix de l'éditeur de texte influence directement votre confort de travail. Un bon éditeur propose la coloration syntaxique, l'autocomplétion, la gestion des projets et la prévisualisation du code.

- Quelques éditeurs populaires :
- Visual Studio Code - léger, rapide, extensible via des extensions.
- Sublime Text - fluide et personnalisable, adapté aux projets de taille moyenne.
- Atom - interface agréable et intégration Git native.
- Brackets - pensé pour le web avec aperçu en direct.
- PHPSTORM / WEBSTORM - logiciels pro et payants mais vous avez un accès gratuitement grâce à votre adresse mail IFAPME

### Bon à savoir

Visual Studio Code est aujourd'hui le plus utilisé pour le développement web grâce à sa communauté active et ses extensions variées.

### ⚠ Erreur fréquente

Évitez les traitements de texte comme Word ou LibreOffice : ils ajoutent des caractères cachés et corrompent le code HTML.



## Préparer la structure d'un site local

Avant de commencer à coder, il est essentiel d'organiser les fichiers de votre site. Une bonne structure facilite la maintenance et le travail collaboratif.

Structure recommandée :

- Un dossier principal nommé selon votre projet (ex. mon-site/).
- Un sous-dossier images/ pour toutes les illustrations.
- Un sous-dossier css/ pour les feuilles de style.
- Un sous-dossier js/ pour les scripts JavaScript.
- Un fichier index.html à la racine du projet.

Illustration textuelle

Schéma typique d'arborescence :

```
mon-site/  
├── index.html  
├── css/  
│   └── style.css  
├── images/  
│   └── logo.png  
└── js/  
    └── script.js
```

## Éléments, balises et attributs HTML

Le HTML est composé d'éléments, chacun défini par une balise ouvrante et une balise fermante. Une balise est entourée de chevrons ``<`` ``>`` et indique la fonction du contenu qu'elle encadre.

```
<p>Ceci est un paragraphe.</p>
```

Ici, ``<p>`` ouvre un paragraphe et ``</p>`` le ferme. L'ensemble constitue un élément HTML.

#### Bon à savoir

Certaines balises, dites autofermantes, ne nécessitent pas de balise de fermeture. Par exemple : ````.

Les attributs ajoutent des informations supplémentaires à une balise. Ils se placent à l'intérieur de la balise ouvrante et suivent la syntaxe ``nom="valeur"``.

```
<a href="https://www.exemple.com">Visiter le site</a>
```

Dans cet exemple, l'attribut ``href`` indique la destination du lien.

#### ⚠ Erreur fréquente

Les attributs ne doivent jamais contenir d'espaces autour du signe égal et les valeurs doivent être entre guillemets.

## Structure minimale d'une page HTML

Une page HTML doit respecter une structure précise pour être valide. Voici le squelette de base :

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Titre du document</title>
  </head>
  <body>
    <h1>Titre principal</h1>
    <p>Mon premier paragraphe.</p>
```

```
</body>  
</html>
```

Le doctype `<!DOCTYPE html>` indique au navigateur que le document utilise la norme HTML5. L'élément `<head>` contient les métadonnées, et le `<body>` regroupe tout le contenu visible.

#### Bon à savoir

Les navigateurs modernes sont tolérants, mais un code mal structuré peut provoquer des incohérences d'affichage.

## Indentation et lisibilité du code

L'indentation consiste à décaler le code vers la droite selon la hiérarchie des balises. Cela ne change pas le rendu visuel de la page, mais facilite énormément la lecture du code.

```
<ul>  
  <li>HTML</li>  
  <li>CSS</li>  
  <li>JavaScript</li>  
</ul>
```

Chaque niveau de balise doit être indenté de deux ou quatre espaces. Les éditeurs de texte modernes effectuent cette mise en forme automatiquement.

### ⚠ Erreur fréquente

Une indentation incohérente rend le code difficile à maintenir et augmente les risques d'oublier une balise fermante.

## Commentaires en HTML

Les commentaires permettent d'ajouter des notes invisibles pour l'utilisateur. Ils sont très utiles pour documenter le code ou séparer les sections.

```
<!-- Ceci est un commentaire -->
```

Les commentaires ne s'affichent pas dans le navigateur mais restent visibles dans le code source.

#### Bon à savoir

Commentez régulièrement vos fichiers, surtout si plusieurs développeurs travaillent sur le même projet.

## Enregistrer et afficher une page HTML

Pour créer votre première page web, il suffit d'un éditeur de texte et d'un navigateur. Après avoir écrit le code HTML, vous devez enregistrer votre fichier avec l'extension `.html`.

#### Bon à savoir

Le nom du fichier ne doit pas contenir d'espaces ni de caractères spéciaux. Utilisez uniquement des lettres, chiffres et tirets. Exemple : `ma-page.html`.

#### Bon à savoir

Le nom du fichier n'est pas toujours visible sur Windows (et mac?) Il faut absolument afficher les extensions dans la configuration des dossiers

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Ma première page</title>
  </head>
  <body>
    <h1>Bonjour tout le monde !</h1>
  </body>
</html>
```

Une fois le fichier enregistré, double-cliquez dessus ou glissez-le dans un navigateur pour l'afficher. Vous verrez apparaître votre titre et le contenu du corps de la page.

## Créer des liens hypertextes

Les liens hypertextes permettent de naviguer entre différentes pages ou vers des sites externes. Ils sont définis par la balise `<a>` (pour 'anchor').

```
<a href="https://www.example.com">Visitez mon site</a>
```

L'attribut `href` précise la destination du lien. Lorsqu'on clique sur le texte entre les balises `<a>` et `</a>`, le navigateur charge la page correspondante.

### Bon à savoir

Pour créer un lien vers une autre page du même site, indiquez un chemin relatif :

```
<a href="contact.html">Page de contact</a>.
```

### ⚠ Erreur fréquente

Ne mettez pas d'espaces dans les noms de fichiers liés. Préférez `page-contact.html` à `page contact.html`.

## Liens internes (ancres)

Les ancres permettent de créer des liens vers une partie spécifique d'une même page. On définit d'abord une ancre avec un attribut `id`, puis on crée un lien vers cet identifiant.

```
<h2 id="section1">Section 1</h2>
<p>Contenu...</p>
<a href="#section1">Retour en haut</a>
```

Lorsque l'utilisateur clique sur le lien, la page défile automatiquement jusqu'à l'élément correspondant.

## Ajouter des images à une page

La balise `<img>` permet d'afficher une image dans une page web. Elle utilise au minimum l'attribut `src` pour indiquer le chemin du fichier image, et l'attribut `alt` pour fournir un texte alternatif.

```

```

L'attribut `alt` est essentiel pour l'accessibilité : il permet aux lecteurs d'écran de décrire l'image aux personnes malvoyantes.

### Bon à savoir

Le texte alternatif (`alt`) s'affiche aussi si l'image ne peut pas être chargée. Utilisez un texte concis et descriptif.

### ⚠ Erreur fréquente

Évitez d'utiliser des images trop lourdes. Optimisez-les pour le web afin d'accélérer le chargement de vos pages.

## Comprendre les chemins relatifs et absolus

Un chemin absolu indique la position complète d'un fichier sur Internet, tandis qu'un chemin relatif indique sa position par rapport au fichier HTML courant.

```
<!-- Chemin absolu -->  
  
  
<!-- Chemin relatif -->  

```

L'usage des chemins relatifs est recommandé pour la portabilité : vos liens fonctionneront même si vous déplacez le site sur un autre serveur.

## Les titres en HTML

Les titres servent à organiser le contenu d'une page web. Ils vont de ``<h1>`` à ``<h6>``, où ``<h1>`` représente le niveau le plus important et ``<h6>`` le moins important.

```
<h1>Titre principal</h1>  
<h2>Sous-titre</h2>  
<h3>Sous-section</h3>
```

Chaque page ne doit contenir qu'un seul ``<h1>``, généralement utilisé pour le titre principal. Les autres niveaux structurent les sous-parties.

### Bon à savoir

Les moteurs de recherche comme Google utilisent la hiérarchie des titres pour comprendre la structure de la page. Une bonne utilisation améliore le référencement naturel (SEO).





## Paragraphe et retours à la ligne

Le texte d'une page est principalement contenu dans des paragraphes, définis par la balise `<p>`. Chaque paragraphe doit être clairement séparé des autres.

```
<p>Ceci est un premier paragraphe.</p>
<p>Ceci est un second paragraphe.</p>
```

Pour forcer un simple retour à la ligne sans créer de nouveau paragraphe, on utilise la balise `<br>`, qui est auto-fermante.

```
<p>Ligne 1<br>Ligne 2<br>Ligne 3</p>
```

### ⚠ Erreur fréquente

N'utilisez pas plusieurs balises `<br>` à la suite pour créer des espaces. Utilisez plutôt le CSS pour gérer les marges entre les éléments.

## Séparer visuellement le contenu

La balise `<hr>` crée une ligne horizontale pour séparer les sections d'une page. Elle est utile pour aérer visuellement le contenu.

```
<h2>Introduction</h2>
<p>Texte d'introduction...</p>
<hr>
<h2>Conclusion</h2>
<p>Texte de conclusion...</p>
```

### Bon à savoir

La balise `<hr>` est purement visuelle. Pour structurer le contenu, utilisez toujours les titres (`<h1>` à `<h6>`) et les balises sémantiques appropriées.

## Espaces et caractères spéciaux

En HTML, plusieurs espaces consécutifs sont interprétés comme un seul. Pour insérer un espace insécable (qui empêche le retour à la ligne), on utilise le code `&nbsp;`.

```
<p>Prix : 10&nbsp;€</p>
```

De même, certains symboles nécessitent un code spécial, car ils sont réservés par le HTML.

```
&lt; pour <
&gt; pour >
&amp; pour &
&copy; pour ©
```

### Bon à savoir

Les entités HTML assurent que le navigateur affiche correctement les symboles réservés. Elles commencent toujours par `&` et se terminent par `;`.

## Mise en forme du texte

Le HTML permet de mettre en valeur certaines portions de texte à l'aide de balises spécifiques. Cependant, ces balises ne doivent pas être confondues avec la mise en forme visuelle, qui relève du CSS.

```
<p><strong>Texte important</strong></p>
<p><em>Texte en italique</em></p>
<p><mark>Texte surligné</mark></p>
<p><small>Texte plus petit</small></p>
```

Les balises ``<strong>`` et ``<em>`` ont une valeur sémantique : elles indiquent une importance ou une emphase. Le navigateur les affiche souvent en gras et en italique par défaut, mais leur signification dépasse le simple style.

### Bon à savoir

Les lecteurs d'écran utilisent la sémantique du HTML pour adapter la lecture du contenu. L'usage correct des balises améliore donc l'accessibilité de votre site.

## Texte préformaté et citations

Le HTML propose des balises pour afficher du texte tel qu'il est écrit dans le code, ainsi que pour insérer des citations.


```
<pre>
Ligne 1
  Ligne 2 (indentée)
Ligne 3
</pre>
```

La balise `<pre>` conserve les espaces et retours à la ligne du texte source, utile pour afficher du code ou des exemples de texte formaté.

```
<blockquote>
  Ceci est une citation plus longue extraite d'un texte.
</blockquote>

<p>Comme disait <q>Einstein</q> : tout est relatif.</p>
```

La balise `<blockquote>` crée un bloc de citation, tandis que `<q>` insère une citation courte en ligne.

 Erreur fréquente  
Ne combinez pas des balises de citation avec des balises de mise en forme pour simuler des styles visuels. Le CSS doit gérer l'apparence.

## Les listes en HTML

Les listes sont des éléments essentiels pour structurer l'information. Il existe deux types principaux : les listes non ordonnées et les listes ordonnées.

```
<ul>
  <li>HTML</li>
  <li>CSS</li>
  <li>JavaScript</li>
</ul>

<ol>
  <li>Étape 1</li>
  <li>Étape 2</li>
  <li>Étape 3</li>
</ol>
```

La balise ``<ul>`` crée une liste à puces, tandis que ``<ol>`` crée une liste numérotée. Chaque élément de liste est défini par la balise ``<li>``.

### Bon à savoir

Les listes peuvent être imbriquées pour représenter des hiérarchies d'informations. Veillez à conserver une indentation claire pour la lisibilité.

## Les listes de définitions

Les listes de définitions permettent d'associer des termes à leurs descriptions. Elles utilisent les balises `<dl>`, `<dt>` et `<dd>`.

```
<dl>
  <dt>HTML</dt>
  <dd>Langage de structure des pages web</dd>
  <dt>CSS</dt>
  <dd>Langage de mise en forme</dd>
</dl>
```

Ces listes sont souvent utilisées dans les glossaires, les FAQ ou les présentations de caractéristiques techniques.

### ⚠ Erreur fréquente

N'utilisez pas de listes de définitions pour faire de simples listes à puces. Elles ont une sémantique spécifique et doivent être réservées aux associations terme/définition.

## Structurer logiquement le contenu

L'utilisation des balises appropriées permet de rendre le code plus clair et accessible. Une page bien structurée sépare la logique du contenu de la présentation.

```
<header>
  <h1>Titre principal</h1>
</header>
<nav>
  <ul>
    <li><a href="#home">Accueil</a></li>
    <li><a href="#about">À propos</a></li>
    <li><a href="#contact">Contact</a></li>
  </ul>

</nav>
<main>
  <section>
    <h2>Introduction</h2>
    <p>Texte de présentation.</p>
  </section>
</main>
<footer>
  <p>&copy; 2025 Mon site web</p>
</footer>
```

Ces balises sémantiques modernes (HTML5) facilitent la lecture du code, l'optimisation pour les moteurs de recherche et l'accessibilité.

## Introduction au CSS

Le CSS (Cascading Style Sheets) permet de contrôler l'apparence d'une page HTML. Il agit sur la couleur, la taille, les marges, la position et d'autres propriétés visuelles des éléments HTML.

```
h1 {
  color: darkblue;
  font-size: 32px;
}
```



Dans cet exemple, la règle CSS s'applique à tous les titres `

# ` de la page, leur donnant une couleur bleue foncée et une taille de police de 32 pixels.

#### Bon à savoir

Le terme *\*cascading\** (cascade) fait référence à la manière dont les styles sont appliqués : si plusieurs règles ciblent le même élément, la priorité dépend de leur ordre et de leur spécificité.

## Les trois méthodes d'intégration du CSS

Il existe trois façons principales d'ajouter du CSS à une page HTML : interne, externe et en ligne.

### Feuille de style externe

```
<link rel="stylesheet" href="style.css">
```

La feuille de style externe est le moyen le plus recommandé. Elle permet de séparer totalement la structure (HTML) et la présentation (CSS). Les modifications de style s'appliquent à tout le site.

```
<style>
p {
  color: green;
  font-size: 18px;
}
</style>
```

Le style interne s'intègre directement dans le fichier HTML, à l'intérieur de la balise ``<head>``. Il est utile pour tester ou styliser rapidement une seule page.

### c) Style en ligne

```
<p style="color:red; font-size:16px;">Texte rouge</p>
```

Le style en ligne applique des règles CSS directement sur un élément HTML grâce à l'attribut ``style``. Cette méthode est déconseillée pour les sites complets, car elle rend le code difficile à maintenir.

#### ⚠ Erreur fréquente

Évitez de mélanger les styles en ligne, internes et externes dans une même page. Cela complique le débogage et provoque souvent des conflits de priorités.

## Comprendre la syntaxe CSS

Chaque règle CSS suit une structure claire composée de trois parties : le sélecteur, la propriété et la valeur.

```
sélecteur {  
  propriété: valeur;  
}
```

Le sélecteur désigne l'élément HTML ciblé, la propriété définit ce qui doit être modifié, et la valeur indique comment le modifier.

```
p {  
  color: blue;  
  text-align: center;  
}
```

#### Bon à savoir

Chaque déclaration CSS se termine par un point-virgule. Oublier ce symbole peut provoquer des erreurs d'interprétation.

## Ajouter des commentaires dans une feuille de style

Les commentaires en CSS servent à documenter le code ou à désactiver temporairement une règle. Ils sont ignorés par le navigateur.

```
/* Ceci est un commentaire CSS */  
h1 {  
  color: navy;  
}
```

#### ⚠ Erreur fréquente

Les commentaires CSS ne peuvent pas être imbriqués. Assurez-vous toujours de fermer le commentaire avec `\*/` avant d'en ouvrir un autre.