R102 — Développement d'interfaces web

Introduction aux langages HTML et CSS

Clément Pagès
clement.pages@u-pec.fr

Université Paris-Est Créteil
BUT Informatique 1re année

2025-2026

Sommaire

Objectifs

- Apprendre à créer des sites Internet statiques.
- Respecter les standards du web.
- Langages étudiés : HTML (contenu des pages), CSS (design).
- S'initier à la gestion de projets informatiques.

Modalités d'évaluation

Écrit 65%	Un QCM en fin de ressource
Pratique 35%	Rendus de TP, mini-projet

Modalités d'évaluation

Écrit 65%	Un QCM en fin de ressource
Pratique 35%	Rendus de TP, mini-projet

Différence TD/TP

Dans cette ressource, on distingue TD et TP.

- TD : tutoriels, exercices guidés, non évalués. Aide de l'enseignant·e possible.
- TP : exercices peu guidés, à rendre pour évaluation. Aide de l'enseignant·e possible.

Qu'est-ce qu'Internet ?

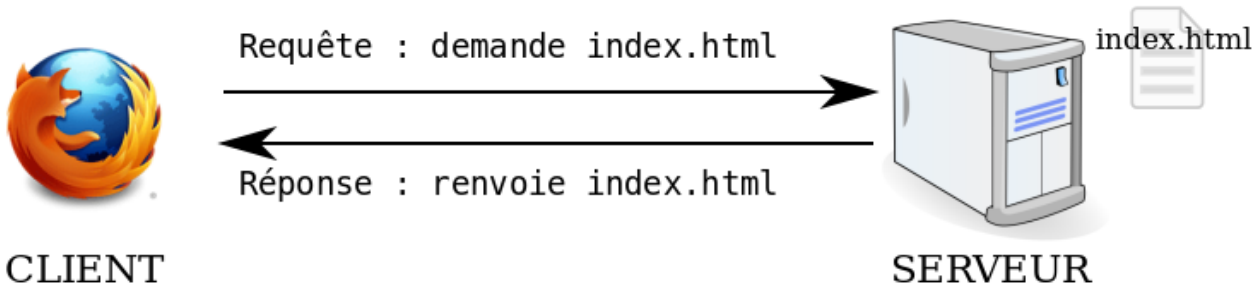
- Un réseau mondial de machines interconnectées, de manière décentralisée.
- Communication entre machines via des protocoles (TCP/IP).
- Applications multiples : mails, transferts de fichiers (FTP), messageries instantanées, peer-to-peer (P2P), World Wide Web (HTTP, HTTPS).
- Dans cette ressource, on va se concentrer sur le web.

Le web

- Inventé en 1989 par Tim Berners-Lee.
- Le web est constitué de pages reliées par des liens hypertexte.
- Basé sur les protocoles de communication HTTP et HTTPS.
- Les pages sont repérées par leur adresse URL (`https://www.u-pec.fr`).
- Les pages sont écrites avec des langages : HTML, CSS, Javascript, PHP, ...
- Elles sont lues par les navigateurs Internet (Mozilla Firefox, Edge, ...).
- Le World Wide Web Consortium (W3C) est l'organisme de référence pour normaliser les stantards du Web et promouvoir la compatibilité entre les diverses technologies.

Architecture client/serveur

- Les pages HTML et CSS sont enregistrées sur une machine distante appelée serveur.
- La machine qui souhaite accéder aux pages est la machine cliente.
- Le client envoie une requête au serveur pour lui demander une page, et le serveur répond à la requête en envoyant la page (ou une erreur...).



Source : Romain Breton

Sommaire

Concepts généraux

- Le contenu des pages est écrit dans des fichiers texte avec l'extension `.html`.
- HTML : *HyperText Markup Language* — Langages à balises pour hypertexte.
- Le document est structuré avec des balises, qui décrivent la sémantique de chaque élément, indépendamment de son apparence.
- La version actuelle la plus utilisée est HTML5.

Concepts généraux

- Le contenu des pages est écrit dans des fichiers texte avec l'extension `.html`.
- HTML : *HyperText Markup Language* — Langages à balises pour hypertexte.
- Le document est structuré avec des balises, qui décrivent la sémantique de chaque élément, indépendamment de son apparence.
- La version actuelle la plus utilisée est HTML5.

Quelques exemples de balises

HTML

```
<title>Titre du site</title>
<a href="autre_page.html">Lien vers une page</a>
<img src="img.png" alt="Desciption de l'image">
<!-- Ceci est un commentaire : ne sera pas affiché,
mais utile pour la ou le programmeur. -->
```

Structure minimale

HTML

```
<!DOCTYPE html> <!-- Type du document -->
<html> <!-- Début du code HTML -->
    <head> <!-- En-tête du document.
                Contiendra des informations de configuration.
            -->
        <title>Titre de la page</title>
        <meta charset="UTF-8"/> <!-- Encodage des caractères -->
    </head>

    <body> <!-- Corps du document -->
    <!-- On place ici le contenu de la page. -->
    <!-- C'est généralement la partie la plus longue -->
    </body>
</html>
```

Règles de syntaxe

Balises en paires

- Balises composées de deux parties. Une balise ouvrante de la forme `<balise>` et une balise fermante de la forme `</balise>`.
- Entre la balise ouvrante et la balise fermante, on écrit le contenu.
- Respecter les règles d'emboitement. Certaines combinaisons sont interdites (ex : `<h1>` à l'intérieur de `<p>`).

Règles de syntaxe

Balises en paires

- Balises composées de deux parties. Une balise ouvrante de la forme `<balise>` et une balise fermante de la forme `</balise>`.
- Entre la balise ouvrante et la balise fermante, on écrit le contenu.
- Respecter les règles d'emboitement. Certaines combinaisons sont interdites (ex : `<h1>` à l'intérieur de `<p>`).

Exemple

HTML

```
<p>Un court pragraphe, avec une partie du
texte <em>mis en valeur.</em></p>

<!-- Ordre des balises inversé. -->
<p>Mais ce deuxième paragraphe est </em>incorrect !</p>
car l'ordre dee fermeture des balises n'est pas bon !</em>
```

Règles de syntaxe

Balise en solo

- Balises composées d'un seul bloc, de la forme `<balise>` ou `<balise />`.

Règles de syntaxe

Balise en solo

- Balises composées d'un seul bloc, de la forme `<balise>` ou `<balise />`.
- Le W3C recommande la première forme...

Règles de syntaxe

Balise en solo

- Balises composées d'un seul bloc, de la forme `<balise>` ou `<balise />`.
- Le W3C recommande la première forme...
- ... sauf pour les balises vides (exemple : `<br />`).

Règles de syntaxe

Balise en solo

- Balises composées d'un seul bloc, de la forme `<balise>` ou `<balise />`.
- Le W3C recommande la première forme...
- ... sauf pour les balises vides (exemple : `<br />`).
- Exemple : `<img src="photo.png" alt="Texte descriptif">`

Règles de syntaxe

Balise en solo

- Balises composées d'un seul bloc, de la forme `<balise>` ou `<balise />`.
- Le W3C recommande la première forme...
- ... sauf pour les balises vides (exemple : `<br />`).
- Exemple : `<img src="photo.png" alt="Texte descriptif">`

Règles de syntaxe

Balise en solo

- Balises composées d'un seul bloc, de la forme <balise> ou <balise />.
- Le W3C recommande la première forme...
- ... sauf pour les balises vides (exemple :
).
- Exemple : <img src="photo.png" alt="Texte descriptif">

Exemple

HTML

```
<p>Un court pragraphe, avec une partie du texte
<em>mis en valeur.</em></p>

<hr> <!-- Ligne horizontale -->

<p>
    Un deuxième paragraphe, contenant un <a href="lien.html">
    lien</a> et une image : <img src="photo.png"
    alt="Description de la photo"><br> <!-- Saut de ligne -->
    Note : les sauts de ligne dans le code source sont ignorés.
</p>
```

Règles de syntaxe

Attributs

- Les attributs permettent de préciser ou de modifier le comportement de la plupart des balises.
- Syntaxe générale :
  `<balise attr1="val1" attr2="val2">…</balise>` et
  `<balise attr1="val1" attr2="val2">`
- Il en existe des dizaines (centaines ?), se référer à la documentation.

Validateur HTML

- Un document est valide s'il respecte toutes les règles de syntaxe.
- Le validateur, un outil indispensable : https://validator.w3.org/

La validité, c'est non négociable

Toutes vos pages HTML doivent être valides. Même si votre site contient plusieurs dizaines de page.

Sommaire

Cascading Style Sheets

Feuilles de style en cascade

- Principe général : séparer du fond et de la forme.

Cascading Style Sheets

Feuilles de style en cascade

- Principe général : séparer du fond et de la forme.
- La structure sémantique est décrite dans les fichiers HTML.

Cascading Style Sheets

Feuilles de style en cascade

- Principe général : séparer du fond et de la forme.
- La structure sémantique est décrite dans les fichiers HTML.
- L'apparence et la mise en forme sont gérées dans les fichiers CSS.

Cascading Style Sheets

Feuilles de style en cascade

- Principe général : séparer du fond et de la forme.
- La structure sémantique est décrite dans les fichiers HTML.
- L'apparence et la mise en forme sont gérées dans les fichiers CSS.
- Le CSS définit un ensemble de propriété qui modifient l'affichage des différents éléments :
  - Polices, tailles, couleurs, typographie, ...
  - Tailles de boites (largeur, marges, ...).
  - Position des boites.
  - etc.

Règle CSS

- Syntaxe générale :

```css
selecteur {
    propriete1: valeur1; /* Notez le point-virgule */
    propriete2: valeur2a, valeur2b, valeur2c;
    propriete3: valeur3a, valeur3b;
}
```

- Le sélecteur désigne les éléments HTML sur lesquels les règles s'appliquent.

Règle CSS

- Syntaxe générale :

```css
selecteur {
    propriete1: valeur1; /* Notez le point-virgule */
    propriete2: valeur2a, valeur2b, valeur2c;
    propriete3: valeur3a, valeur3b;
}
```

- Le sélecteur désigne les éléments HTML sur lesquels les règles s'appliquent.

Exemples

```css
h1 { /* Tous les éléments h1*/
    color: blue;
    font-size: 16pt;
}
h2, h3 { /* Éléments h2 et h3 */
    color: #a123b4; /* Couleur en hexadécimal */
    font-weight: bold;
}
```

Feuilles de styles

- On place les codes CSS dans des fichiers dont l'extension est `.css`
- On importe le fichier CSS dans le code HTML, entre les balises `<header>` et `</header>` :

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

- Un site Internet (même très gros) contient très peu de fichiers CSS. En général un ou deux, très rarement plus de quatre.

Feuilles de styles

- On place les codes CSS dans des fichiers dont l'extension est `.css`
- On importe le fichier CSS dans le code HTML, entre les balises `<header>` et `</header>` :

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

- Un site Internet (même très gros) contient très peu de fichiers CSS. En général un ou deux, très rarement plus de quatre.

La chose à ne pas faire : un CSS par page

On ne crée pas un fichier CSS par page. L'intérêt du CSS est de rendre uniforme la présentation sur l'ensemble des pages d'un site.
Un seul fichier CSS par projet (éventuellement deux pour les très gros projets ou les pages très particulières).

Les `id` et les `class`

Problème épineux

- On souhaite appliquer des styles différents à deux éléments de même type.
- Exemple : écrire un paragraphe en **gras** et un autre en *italique*.

Les `id` et les `class`

Problème épineux

- On souhaite appliquer des styles différents à deux éléments de même type.
- Exemple : écrire un paragraphe en **gras** et un autre en *italique*.
- Problème : comment distinguer deux paragraphes qui sont délimités par la même balise `<p>` ?

Les `id` et les `class`

Problème épineux

- On souhaite appliquer des styles différents à deux éléments de même type.
- Exemple : écrire un paragraphe en **gras** et un autre en *italique*.
- Problème : comment distinguer deux paragraphes qui sont délimités par la même balise `<p>` ?
- Solution : utiliser l'attribut html `class`.

Les `id` et les `class`

Problème épineux

- On souhaite appliquer des styles différents à deux éléments de même type.
- Exemple : écrire un paragraphe en **gras** et un autre en *italique*.
- Problème : comment distinguer deux paragraphes qui sont délimités par la même balise `<p>` ?
- Solution : utiliser l'attribut html `class`.

HTML
```
<p class="p_italique">Un paragraphe en italique.</p>
<p class="p_gras">Un paragraphe en gras.</p>
<p class="p_italique">Un autre paragraphe en italique.</p>
<p>Un paragraphe sans rien de particulier</p>
```

CSS
```
p.p_italique {
    font-style: italic;
}
p.p_gras {
    font-weight: bold;
}
```