TD1 — Premiers pas en HTML

1 Environnement de travail

Commencez par vous placer dans votre dossier `Documents▸BUT_INFO`, puis créez un sous-dossier `R102_DIW`. Placez-vous dans ce dossier. À l'intérieur, créer un sous-dossier `TD1`.

Pour créer et faire fonctionner vos sites Web, il vous faudra deux outils à installer sur vos machines.

Un éditeur de texte dans lequel vous écrirez les codes HTML (et plus tard d'autres langages) de vos pages. Attention, un éditeur de texte *n'est pas* un logiciel de traitement de texte. Il ne gère pas la mise en forme du document.

Un serveur qui aura pour rôle de simuler la machine, en principe distante, qui fournit les pages.

Il existe de très nombreux éditeurs de textes et de serveurs. Pour l'éditeur de texte, nous allons utiliser `Notepad++` et pour le serveur, nous utiliserons `QuickPHP`.

1.1 Éditeur de texte `Notepad++`

Dans cette ressource, nous utiliserons Notepad++. Rendez-vous sur le site https://notepad-plus-plus.org/, puis à la page `Download`. Cliquez sur le lien correspondant à la version la plus récente (v8.8.1 au moment où ces lignes sont écrites). Téléchargez l'exécutable (excension `.exe`) et ouvrez-le. Puis suivez les instructions pour l'installation.

1.2 Serveur `QuickPHP`

Remarque. En principe, cette partie a déjà été réalisée lors des premières séances de MRU, lorsque vos machines ont été distribuées.

Commencez par ouvrir votre dossier `Documents`, puis creéez à l'intérieur un sous-dossier `QuickPHP`. Allez sur EPREL, dans la ressource R1.02 et cliquez sur la vignette `QuickPHP`. Cela va télécharger l'archive ZIP `quickphp_webserver.zip`. L'archive est composée de quatre fichiers :

- `License and readme.txt`, qui contient les informations légales mais n'a pas d'utilité technique
- `QuickPHP.exe` contient le programme principal du serveur.
- `php5ts.dll` est un fichier de configuration indispensable au fonctionnement du serveur.
- `php_license.txt` contient d'autres informartions légales.

Copiez et collez les quatre fichiers dans le répertoire `Documents▸QuickPHP` que vous venez de créer.

Attention

Les fichiers `QuickPHP.exe` et `php5ts.dll` doivent absolument être sauvegardés dans le même répertoire pour fonctionner correctement.

Si vous tentez d'exécuter directement `QuickPHP.exe` à l'intérieur de l'archive ZIP, l'exécution plantera. Il faut penser à extraire le contenu de l'archive.

2 Vos premières pages HTML

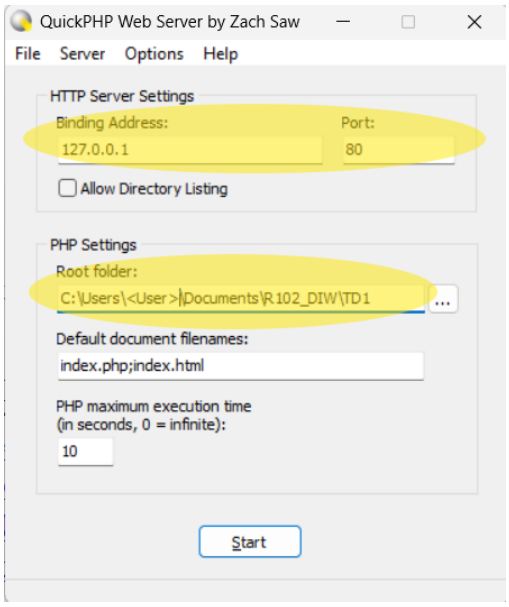
Pour l'ensemble du TD (et en fait de la ressource), effectuez les tests de vos pages en mode *navigation privée*.

Exercice 1 (Un premier exemple)**.** On considère le code suivant :

```HTML
<!DOCTYPE html>
<html>
    <body>
        <h1>niveau 1</h1>
        <p>Un paragraphe.</p>
        <h2>niveau 2</h2>
        <p>Un autre paragraphe
        pour faire joli.</p>
    </body>
</html>
```

1) Ouvrez le serveur `QuickPHP` et renseignez les champs suivants :

   Binding adress `127.0.0.1`
   Port `80`
   Root folder Choisissez le répertoire `Documents▸BUT_INFO▸R102_DIW▸TD1`.



Cliquez sur Start et ouvrez votre navigateur Internet à l'URL `http://127.0.0.1`. Que remarquez-vous ?

2) Récupérez le code et enregistrez-le dans `Documents ▸ BUT_INFO ▸ R102_DIW ▸ TD1` avec le nom `exo1.html`. Dans votre navigateur Intenet, accédez à l'URL `127.0.0.1/exo1.html`. Que remarquez-vous ?

3) À quoi les balises `<h1>`, `<h2>` et `<p>` servent-elles ?

Rappel sur la sémantique

Les balises HTML n'ont pas pour objectif de donner des instructions sur l'aspect du contenu, mais de le structurer, d'appporter du sens sur l'élément balisé. C'est pourquoi on parle de *balisage sémantique*.

La manière dont le titre est présenté n'est pas importante (pour l'instant), ce qui compte est qu'il s'agisse d'un titre. Le navigateur Internet utilise un style par défaut sur les balises HTML. Avec le CSS, on pourra associer à chaque balise l'apprence que l'on souhaite.

4) Testez votre code dans le validateur du W3C, via l'onglet Direct input. Que constatez-vous ?

5) Ajouter les lignes suivantes entre les balises `<html>` et `<body>` :

```HTML
<head>
    <meta charset="UTF-8">
    <title>Titre de la page</title>
</head>
```

6) Actualisez la page dans votre navigateur Internet. Que remarquez-vous ? Passez maintenant la page au validateur. Quel est le résultat obtenu ?

7) Précisons maintenant la langue, en remplaçant la balise `<html>` par `<html lang="fr">`. Cela ajoute un *attribut* à la balise.

8) Passez de nouveau votre code au validateur. L'exercice n'est considéré terminé *que si votre code est valide*.

Exercice 2 (Images). Nous allons ajouter une image à notre page. Le code de base pour insérer une image est le suivant :

```HTML
<img src="smiley.png" alt="Smiley face" heigth="42" width="42">
```

1) Les navigateurs Internet utilisent uniquement trois formats : `GIF`, `PNG` et `JPEG`. Cherchez et expliquez les différences entre ces trois formats.

2) Utilisez une image de votre choix ajoutez-la à votre page. Que pensez-vous du résultat ? Comment améliorer la qualité du rendu ?

3) À quoi les attributs `src` et `alt` servent-ils ? Mettez un nom d'image incorrect dans l'attribut `src` et comparez le résultat.

4) Vérifiez la validité de votre code. Retirez l'attribut `alt` et vérifiez à nouveau. Que remarquez-vous ?

Accessibilité des pages

L'attribut `alt` est par exemple indispensable aux personnes malvoyantes pour avoir une description de l'image, ou si le serveur n'est pas en mesure de fournir l'image.
C'est pourquoi il faut fournir un texte alternatif suffisamment précis. Le code `<img alt="image" src="pingouin.jpg">` est valide, mais n'est pas de bonne qualité. On préfèrera `<img alt="Un manchot sur une plage." src="pingouin.jpg">`.

5) Nous allons placer les images de notre site dans un dossier spécifique. Dans le dossier `R102_DIW`, créez un sous-dossier `img` et déplacer votre image à l'intérieur. Adaptez votre code HTML pour afficher l'image.

Exercice 3 (Liens). Les liens, ou hyperliens, sont la principale originalité initiale des documents hypertextes, d'où ils tirent leur nom. Un lien est décrit par une balise en paire `<a>…</a>` et l'attribut `href` vous permet de fournir l'url de la page liée. Ils permettent la redirection ou l'ouverture d'une nouvelle fenêtre au clic, via du texte ou une image, ou le téléchargement d'une ressource :

```HTML
<a href="https://developer.mozilla.org/fr/docs/Web/HTML">Documentation HTML
de Mozilla</a>
```

1) Liens vers une page.

a. Placer un lien vers le site de l'IUT.

b. Faire en sorte que le lien soit le logo de l'IUT.

On peut utiliser des *adresses relatives* au document de départ. Si le document ciblé (par exemple `musique.mp3` se trouve dans le même répertoire que celui où l'on se trouve, le lien peut s'écrire `<a href="musique.mp3">…</a>`.
Si la cible est dans un sous-dossier, disons `audio ▸ musique.m3`, le lien est remprend le chemin d'accès : `<a href="audio/musique.mp3">…</a>`.
Si le document ciblé, par exemple `index.html` se trouve dans le répertoire parent, le lien est `<a href="../index.html">…</a>`.

2) Lien vers une ancre — Une ancre est un repère placé dans une balise pour localiser un endroit précis à l'intérieur d'une page. Cela permet de créer un lien vers cet endroit précis, appelé *point d'angrage*. Le point d'ancrage doit être identifié de manière unique. On utilise pour cela l'attribut `id`. Par exemple :

```HTML
<h1 id="intro">Introduction</h1>
```

Pour faire un lien vers ce titre, la syntaxe est alors `<a href="#intro">vers l'introduction</a>`.
On peut aussi faire un lien vers une ancre située dans une autre page.

a. Ajouter la ligne de code suivante à votre page, et expliquez ce qu'elle fait.

```HTML
<a href="https://developer.mozilla.org/fr/docs/Web/HTML/Reference/
Elements/a#href">En savoir plus sur les ancres</a>
```

b. Ajoutez des titres de plusieurs niveaux à votre page. Tout en bas de page, ajoutez un lien vers une ancre sur le titre de premier niveau.

3) Cherchez comment intégrer un lien vers une adresse mail. Ajoutez sur votre page un lien de contact avec une adresse mail fictive de la forme `butinfo@test.fr`.

4) Vérifiez la validité de votre code source avec le validateur W3C.

Exercice 4 (Listes). Placez le code source suivant à l'intérieur d'un paragraphe, et testez-le :

```HTML
<ol>
    <li>Café</li>
    <li>Thé</li>
    <li>Lait</li>
</ol>
<ul>
    <li>Café</li>
    <li>Thé</li>
    <li>Lait</li>
</ul>
```

En utilisant uniquement des listes ordonnées et non ordonnées, obtenez le rendu suivant :

- Lundi
    1. Cours de 8 h à 16 h.
    2. Faire du vélo.
    3. Faire les courses.
- Mardi
    1. Finaliser l'inscription administrative.
    2. Cours toute la journée.
    3. Écouter le dernier podcast « Voyage sans ailes.
    4. Sieste !!
- Mercredi
    1. Aller à la fac à vélo.
    2. Commencer la lecture de *Betty*, Tiffany McDaniel.
    3. Terminer le TD de HTML...

3 Le mot de la fin

Quelques bonnes pratiques

- La page d'accueil d'un site Internet est toujours nommée `index.html`.
- Les fichiers annexes (images, vidéos, fichiers texte, PDF, etc.) sont organisés dans des sous-dossiers du projet. On ne met pas tout en vrac à la racine !
- Les pages HTML doivent être valides.
- Les balises utilisées respectent la sémantique.
- Les noms de fichiers et de dossiers ne contiennent ni espace ni caractères spéciaux. Ils doivent être clairs et explicites.