HTML COURSE

1. GETTING STARTED WITH HTML
   1. Hello World
      1. introduction

HTML : Hyper Text Mark up Language : utilise un système de balisage composé d’éléments qui représentent des contenus.

Markup : signifie qu’avec HTML vous déclarez ce qui est représenté à l’utilisateur mais pas comment il est représenté

La représentation visuelle est définie par le CSS (Cascading Style Sheets) et réalisée par le navigateur.

HTML est de fois appelé un langage de programmation mais ça n’a pas de logique donc c’est un langage de balisage ou un langage de description.

Les balises HTML fournissent une signification sémantique et une lisibilité au contenu de la page.

Un élément est composé généralement d’une balise ouvrante <element\_name>, et d’une balise fermante </element\_name>, qui contiennent le nom de l’élément entouré de crochets angulaires et le contenu entre : <element\_name> … contenu <element\_name>

Certains éléments HTML n’ont pas de balise de fermeture ni de contenu. Ceux sont appelés des élements vides ou des balises orphelines : <img>, <meta>, <link>, and <input>

Les noms des éléments peuvent être considérés comme des mots clés descriptifs pour le contenu qu’ils contiennent, tels que : video, audio, table, footer, etc …

Une page HTML peut contenir potentiellement des centaines d’éléments qui sont ensuite lus par un navigateur web et interpretés et rendu dans un contenu lisible ou audible par l’homme sur l’écran

Nous allons distinguer un élément HTML et une balise HTML

Element: video, audio, table, footer

Balises ou Tags : <video>, <audio>, <table>, <footer>,</html>, </body>

* + 1. Elements Insight: Apercu des elements

<p> : la balise représente un paragraphe

Les éléments ont généralement une balise ouvrante et une balise fermante. La balise d’ouverture contient le nom de l’élément entre crochet (<p>).

La balise de fermeture est identique à la balise d’ouverture avec l’ajout d’une barre oblique (/) entre la parenthèse ouvrante et le nom de l’élément (</p>)

Le contenu peut etre placé entre les deux balises : <p> This a paragraph </p>

* + 1. Creating a simple page

La page HTML est créée par un éditeur de texte. Et le fichier doit avoir une extension .html ou .htm pour etre reconnu comme page HTML

<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <title>Hello World</title>

  </head>

  <body>

    <h1>Hello World</h1>

    <p>This is a simple paragraph</p>

  </body>

</html>

Explication de cette page

* <! DOCTYPE html > : définie la version de la page HTML utilisée dans le document. Dans ce cas il s’agit de HTML5.
* <Html> : ouvre la page HTML donc ça marque le début de la page HTML, aucune balise ne doit venir apres la fermeture </html>. L’attribut **lang** déclare la langue principale de la page en utilisant les codes de lange ISO (en : anglais, fr : français)
* <head> : ouvre la section d’en tête, qui n’apparait pas dans la fenêtre principale du navigateur mais contient principalement informations sur le document HTML, appelées **métadonnées.** Il peut également contenir des importations en provenance de l’extérieure comme par exemple de style sheet ; des scripts. La balise de fermeture est </head>
* <meta > : cette balise donne au navigateur des métadonnées à propos du document ou de la page HTML. L’attribut charset déclare le caractère d’encodage de la page. HTML moderne utilise UTF-8, cette balise ne demande pas une balise fermante.
* <title> : Le titre de la page. Le texte écrit entre cette balise ouvrante et la balise fermante (</title>) sera affiché sur l'onglet de la page ou dans la barre de titre du navigateur.
* <body> : ouvre la partie du document à afficher aux utilisateurs ; c’est-à-dire tout le contenu visible ou audible d’une page. Pas de contenu après la balise de fermeture </body>

1. DOCTYPES

Doctypes c’est une abréviation de **Document Type** – ça aide juste le navigateur à comprendre la version HTML utilisée par le document ou la page pour une meilleure interopérabilité.

Les déclarations doctypes ne sont pas de balises HTML et apparaissent juste tout en haut du document.

* 1. Adding the Doctype

La déclaration <! DOCTYPE> doit venir en début de la page HTML avant même l’élément html ou la balise <html>

En version 5 le doctype est définie : < ! DOCTYPE html> : juste pour dire au navigateur que nous utilisons une version 5 de html

* 1. HTML5 DOCTYPE

HTML5 n’est pas basé sur SGML (Standard Generalized Markup Language), et ne nécessite donc pas de référence à un DTD (Document Type Définition)

C’est quoi SGML : est une spécification ISO pour définir des langages de balisage génériques pour de document. Html1.0 – html4 étaient basé sur SGML mais avec sa version 5, html n’est plus basé sur SGML et possède ses propres règles

* + 1. Case insensitivity: insensible a la casse

Insensible à la casse CAD les majuscules et les minuscule sont traités de la même façon donc <!docType hTml> == < !DOCTYPE html>

Sensible a la casse les majuscules et les minuscules sont traités d’une manière différente

1. HEADINGS : LES TITRES

HTML ne fournit pas seulement des balises de paragraphes mais aussi 6 niveaux de balise de titres différente de tailles et d’épaisseur.

En se basant de 1 à 6 ; le titre 1 a le texte le plus long et plus épais pendant que le titre 6 est plus petit et plus fin jusqu’au niveau du paragraphe.

<!-- LES TITRES -->

    <h1>Hello World</h1>

    <h2>Hello World</h2>

    <h3>Hello World</h3>

    <h4>Hello World</h4>

    <h5>Hello World</h5>

    <h6>Hello World</h6>

* 1. La structure correcte

Les moteurs de recherche et autres agents utilisateurs indexent généralement les pages en fonction des éléments de titre. Il est donc important d’utiliser la bonne structure pour les titres.

**Notes :**

* L’information de titre peuvent être utiliser par le navigateur pour construire automatiquement la page de matière de ta page
* Il faut éviter de sauter le niveau de titre sur une page, on commence toujours par h1, suivi de h2, etc,
* On essayera d’avoir au moins un titre de niveau 1 sur au moins chaque page du site
* Avec html5 on peut maintenant avoir plusieurs titre h1 sur une page, en ayant au moins un en début de chaque section

1. LES PARAGRAPHES

<p> : définie un paragraphe

<b> : insère un seul saut de ligne

<pre> : défini un texte reformaté

* 1. HTML PARAGRAPHS

1. <!-- LES PARAGRAPHES -->
2. <p>This is a paragraph</p>
3. <p>This is an other paragraph</p>

Affichages:

* On ne peut pas être sûr de la façon dont HTML sera affiché
* Des écrans grands ou petits et des fenêtres redimensionnées créeront des résultats différents
* Avec HTML vous ne pouvez pas modifier la sortie en ajoutant des espaces ou des lignes supplémentaires dans votre code HTML
* <p>This      is a      paragaph     to      display</p>

Le navigateur va supprimer ces espaces vide lorsque la page sera affichée

**Notes : Aucune.**

1. TEXT FORMATTING : LES FORMATAGES DU TEXT

La plupart de balise HTML sont utilisées pour créer des éléments, HTML fournit également des balises de formatage dans le texte pour appliquer des styles liés à des portions de texte.

* 1. HIGHLIGHTING: <mark>

L’élément mark ou balise <mark> est nouveau dans html5 et utilisé pour marquer ou surligner du texte dans un document en raison de sa pertinence dans un autre contexte

Cas d’utilisation : l’exemple le plus courant serait dans les résultats d’une recherché ou l’utilisateur a saisi une requête de recherché et les résultats sont affichés en mettant en évidence la requête souhaitée

 <p>

      Here is some content from an article that contains the

      <mark>searched query</mark> that we are looking for. Highlighting the text

      will make it easier for the user to find what they are looking for.

    </p>

Par défaut le texte marqué par la balise <mark> est de couleur jaune en background

**Notes :**

* <mark> ne doit pas etre utiliser pour la coloration syntaxique, c’est plutôt <span> qui doit etre utilisé
* <mark> ne doit pas etre confondu avec <strong> : ce dernier indique que le texte est important tandis que <mark> indique que le texte est pertinent.
  1. Bod, Italic, underline
     1. Bold

<strong> contenu </strong>:

 <p>

      Here is some content from an article that contains the

      <strong>searched query</strong> that we are looking for.

    </p>

<b> contenu </b>:

<p>

      Here is some content from an article that contains the

      <b>searched query</b> that we are looking for.

    </p>

En apparence <strong> et <b> donne le même résultat, mais la différence se trouve du point de vu sémantique, l’élément strong est utilisé pour indiquer que le texte est fondamentalement ou sémantiquement important et est mis en gras tandis que l’élément b indique l’élément n’est pas important mais doit être seulement en gras.

Si on utilise un programme de synthèse vocale, le mot entouré par <b> ne sera pas lu différemment des autres mots qui l’entoure car ça va justement permettre d’attirer simplement l’attention sans ajouter une importance supplémentaire, mais avec <strong> le meme programme va dire ce mot avec un ton de voix diffèrent pour transmettre que le texte est important d’une certaine manière.

* + 1. Italic : <em> et <i>

<em> :

 <p>

      Lorem ipsum dolor sit amet <em>voluptatum numquam</em> consequatur

      voluptatem facere eos perferendis.

    </p>

<i>

<p>

      Lorem ipsum dolor sit amet <i>voluptatum numquam</i> consequatur

      voluptatem facere eos perferendis.

    </p>

Du point de vu apparence on voit que l’affichage est la même, mais du point de vu sémantique ; la balise <em> est utilisé que le texte doit avoir une emphase(amplication) supplémentaire qui doit être accentué ou soit juste un texte sur lequel on insiste tandis que <i> représente que le texte doit être séparé du texte normal qui l’entoure et par défaut ça prend l’italic.

Si vous voulez accentuer un accent sur une action, on utilise : <em> :: souvent utiliser pour insister surtout à certaines actions

Ex : Souhaitez-vous déjà <em>soumettre</em> la modification ?

Si vous voulez juste mettre un texte en italic ou soit mettre en prose ou separe un texte, un livre etc: <i>

* + 1. Underline : <u>

Cet élément a été obsolète dans html4 mais ré introduit dans 5 avec une signification sémantique alternative, dans html5 pour représenter une annotation (critique) non articulée et non textuelle.

On peut utiliser un tel rendu pour indiquer que le texte est mal orthographié sur la page ou pour une marque de nom propre chinois

<p>

      Lorem ipsum dolor sit amet <u>voluptatum numquam</u> consequatur

      voluptatem facere eos perferendis.

    </p>

* + 1. Abbreviation: <abbr>

Est utiliser pour montre que l’expression est une abréviation et présente un attribut title pour décrire l’abréviation.

    <p>

      i like write <abbr title="Hyper Text Makeup Language">HTML</abbr> because

      is not have some logic

    </p>

L’attribut title : donne une full description de l’abréviation

* + 1. Inserted, Deleted, Stricken
* Inserted : <ins> : représente un fragment du texte qui a été ajouté au document
* Deleted : <del> : représente un fragment du texte qui a été supprimer du document
* Stricken : <s> : permet d’afficher le texte qui est barré car il n’est plus pertinent ou il ne doit pas être employé pour indiquer des éditions sinon on utilise <ins> et <del>

Note : la plupart d’assistance n’annonce pas la présence des éléments <ins> , <del> et <s> on peut les annoncer par les pseudos ::before et :: after et avec la propriété *content*

<p>

      Lorem ipsum dolor sit amet <ins>Teddy</ins> adipisicing elit. Nulla odit

      libero soluta quos iusto adipisci! Neque autem <del>impedit</del> laborum

      officia numquam dicta <s>facere</s> reprehenderit nam iusto consequuntur,

      deleniti sunt ipsum.

    </p>

<ins>

      <p>Morceau inserer dans le document</p>

    </ins>

    <del>

      <p>Morceau du texte supprimer du docuement</p>

    </del>

* + 1. Superscript and subscript
* Superscript : <sub> : pour afficher le texte en exposant
* Subscript : <sup> : pour afficher le texte souscrit en indice ou généralement plus bas

    <!-- LE SUBSCRIPT AND SUPERSCRIPT -->

    <p>

      C

      <sub>8</sub>

      H

      <sub>9</sub>

    </p>

    <p>

      Voici la fonction exponentielle : e

      <sup>x</sup>

    </p>

Note : l’utilisation pour de formule mathématique : x 2 , mais pour des cas un peu plus complexe il est nécessaire d’utiliser **MathML (a exploiter)**, affichage de lettre plus supérieure pour respecter certaines conventions typographiques ( ex M lle), représentation de nombre ordinaux 7e art.

1. ANCHORS and HYPERLINKS : LES ANCRES ET LES HYPERS LIENS

Les liens ont des paramètres

* **href** : spécifie l’adresse de destination, il peut s’agir d’une URL absolue ou relative ou d’un nom d’une ancre. Une url absolue c’est une url complète d’un site web (<https://facebook.com>), une URL relative pointe vers un autre répertoire, ou un document a l’intérieur du même site web (/about-us/ : qui signifie pointe vers le répertoire about-us DNS le répertoire racine / , lorsque vous pointez dans un répertoire sans spécifier explicitement le document, les serveurs web renvoie généralement le document index.html de ce répertoire)
* **hreflang** : spécifie la langue de la ressource liée par l’attribut Href ;
* **rel** : spécifie la relation entre le document actuel et le document lié (liste des attributs : [4.8 Links — HTML5 (w3.org)](https://www.w3.org/TR/2014/REC-html5-20141028/links.html#linkTypes)), https://developer.mozilla.org/fr/docs/Web/HTML/Link\_types

|  |  |  |
| --- | --- | --- |
| Link type | Effect on | description |

* **target** : spécifie ou ouvrir le lien, ex ( dans un nouvel onglet, ou une nouvelle fenêtre), les valeurs possible sont : \_blank, \_self, \_parent, top, force un tel comportement n’est pas recommandé car il viole le contrôle de l’utilisateur
* **title** : spécifie les informations supplémentaires sur un lien. Les informations sont souvent afficher sous forme de texte info bulle lorsque le curseur se déplace dessus, cet attribut n’est pas limité au lien, il peut être appliqué sur presque toutes les balises HTML.
* **Download** : spécifie que la cible sera téléchargée lorsqu’un user va cliquer sur le leien. La valeur de l’attribut sera le nom du fichier téléchargé.

Les balises d’ancrage sont couramment utilisées pour lier des pages web distinctes, mais elles peuvent être aussi utiliser pour créer des liens entre différentes places dans une seule page

* 1. Link to another site

C’est le cas d’utilisation basic de l’élément d’ancrage

 <a href="http://facebook.com" rel="external"

      >Cliquer ici pour aller sur facebook</a

    >

Ça crée un lien hyper texte vers le site de Facebook comme spécifie l’attribut href,

Pour indiquer que le lien mène vers un site externe alors nous ajoutons rel = “external”

<a href="ftp://example.com">Vers un site FTP</a>

Vers un site ftp

* 1. Link to an Anchor : lien vers une ancre

Les ancres peuvent être utilisée pour accéder à des balises spécifiques sur une page HTML. La balise <a> peut pointer vers n’importe quel élément qui a un identifiant attribut.

 <a href="#Topic1">Click to jump to the first topic</a>

    <a href="#Topic2">Click to jump to the second topic</a>

Les sections

 <h2 id="Topic1">First content</h2>

    <p>Content about first topic</p>

    <h2 id="Topic2">Second content</h2>

    <p>Content About second topix</p>

* 1. Link to page on the same site: Lien sur la page du meme site

Nous devons utiliser le chemin relatif, ou une url relative

<a href="page1.html">Go to page 1</a>

* 1. Link to dials a number

Si la valeur d’attribue href commence par tel, votre appareil composera le numéro de téléphone lorsque vous cliquer dessus,

Cela fonctionne sur mobile, sur ordinateur ou tablette exécutants des logiciels comme skype qui peuvent passer des appels téléphoniques.

<a href="tel:+243972194053">Call Me</a>

* 1. Open a link in new tab / Window

<a href="https://facebook.com" target="\_blank">Open in new tab</a>

En précisant l’attribut target on demande au navigateur de l’ouvrir dans un nouvel onglet ou nouvelle fenêtre,

Security vulnerability

En utilisant la balise target =’\_blank’, on donne accès au site ouvert l’accès a l’objet window.opener de notre page via javascript, et de cela le site destination peut modifier l’objet window.opener et peut facilement créer des redirections pour de phishing.

Pour pallier nous devons ajouter le paramètre ou l’attribue rel = “noopener noreferrer” pour un effect maximal de préventions que l’object window.opener soit envoyer à la page destination.

    <a href="https://facebook.com" target="\_blank" rel="noopener noreferrer"

      >Open in new tab</a

    >

* 1. Link that runs javascript

On peut créer un lien qui appelle une fonction javascript

<a href="javascript:myFunction()">Run Code</a>

To prevent an href = “#” to scrolling to top, nous devons après la function retourner un false

<a href="#" onclick="myFunction(); return false">Run Code Again</a>

On peut encore interdire le scrolling to top en ajout un ! après le #

<a href="#!"> Scroll on top</a>

Ceci prévient le scrolling on top parce que ça recherché un élément avec id =” !” qui ne sera pas retrouver et puis ça va scroller null part

**Notes :** cette solution n’est pas envisageable, il faudra mieux utiliser du javascript pure pour ce genre de cas.

* 1. LINK TO RUNS EMAIL CLIENT

Basic usage

Si la valeur d’un href commence par mailto, ça va essayer d’ouvrier un logiciel de messagerie pour envoyer à un client.

<a href="mailto:teddywalterjob@gmail.com">Send Email</a>

Ceci va mettre l’adresse email dans la destination

Cc (Copie carbone) et Bcc ; cci (Copie Carbonne invisible)

CC : cad le destinateur verra cette liste ou ces emails qui sont en copie et Bcc il ne verra pas les mails qui sont en copie

<a

      href="mailto:teddywalterjob@gmail.com ?cc=john@gmail.com&bcc=test@gmail.com"

    ></a>

Subject and body

<a

      href="mailto:teddywalter@gmail.com?subject=Example+Subject& body=Message+Text"

    >

      Send Email Subkec</a

    >

1. LES LISTES

Nous avons 3 sortes des listes, les listes ordonnées, les non ordonnées, les listes de description.

Par défaut les listes ordonnées utilisent une séquence de nombre pour indiquer l’ordre ; une liste non ordonnée utilise par exemple de puces pour lister les éléments sans ordre défini et les listes de description utilisent des retraits pour répertorier les éléments avec leur enfant.

* 1. Liste ordonnée

Une liste ordonnée est créée par la balise <ol> et les éléments sont créer par <li>

    <ol>

      <li>Item</li>

      <li>Another Item</li>

      <li>Yet Another Item</li>

    </ol>

On peut jouer avec les nombres qui apparaissent sur les éléments de la liste dans une liste ordonnée. La première consiste à définir un numéro de départ avec start comme attribut dans <ol> et on verra liste commencé par le nombre

    <ol start="4">

      <li>Item</li>

      <li>Another Item</li>

      <li>Yet Another Item</li>

    </ol>

La notion de renverse list introduit en html5 permet de renverser les numero de la list

    <ol reversed>

      <li>Teddy</li>

      <li>Thibert</li>

      <li>Titien</li>

    </ol>

Affichage



Changing list type numérotation

Dans la balise <ol> on peut ajouter un attribut type qui peut prendre différente valeur : type= 1|a|A|i|I

1 : numéro décimal : 1,2,3,4,5

a : numérotation alphabétique en minuscule : a,b,c,d,e

B : numérotation alphabétique en majuscule : A,B,C,D’

i : chiffre romain en minuscule : i,ii,iii,iv

I : chiffre romain en majuscule : I, II, III, IV

    <ol type="I">

      <li>Item 01</li>

      <li>Item 02</li>

      <li>Item 03</li>

      <li>Item 04</li>

      <li>Item 05</li>

    </ol>

* 1. List non ordonnées

Ces genres de liste sont créée par la balise <ul> et les éléments sont créés par <li>

Note : l’attribut type est obsolète dans les <ul>

<ul>

      <li>Item 01</li>

      <li>Item 02</li>

      <li>Item 03</li>

      <li>Item 04</li>

    </ul>

* 1. Nested List

C’est juste une création de list imbrique

La liste imbrique doit être l’enfant de <li>

On peut aussi imbriquer différente type de liste : les ul dans les ol et vice versa

 <ul>

      <li>Item</li>

      <li>Item</li>

      <li>

        Item

        <ul>

          <li>Item</li>

          <li>

            Item

            <ul>

              <li>Item</li>

              <li>Item</li>

            </ul>

          </li>

        </ul>

      </li>

      <li>item</li>

    </ul>

* 1. Liste de description

Ces listes sont créées par <dl>, et on les noms de valeur sont donné par <dl> et les valeurs en question avec <dd>

 <dl>

      <dt>Nom</dt>

      <dt>Prenom</dt>

      <dd>Teddy Walter</dd>

      <dt>Prenom</dt>

      <dd>Walter</dd>

      <dd>Nkubonage</dd>

    </dl>

1. LES TABLEAUX
   1. Simple table

<table>

      <tr>

        <th>Nom</th>

        <th>Prenom</th>

        <th>Age</th>

      </tr>

      <tr>

        <td>TEDDY</td>

        <td>walter</td>

        <td>24</td>

      </tr>

      <tr>

        <td>TEDDY</td>

        <td>walter</td>

        <td>24</td>

      </tr>

    </table>

La balise <table> va afficher un tableau composé de 3 lignes de <tr> : le premier <tr> est composé des éléments <th> : tabular header pour l’entête du tableau, et les 2 autres par de <td> : tabular data, pour les contenus des données.

* 1. SPANNING COLUMNS AND ROWS

Pour être les colonnes on utilise l’attribut colspan dans l’élément <tr> et pour les lignes on utilise l’attribut rowspan dans le <tr>

<table border="1">

      <tr>

        <th>Nom</th>

        <th>Prenom</th>

        <th>Sexe</th>

      </tr>

      <tr>

        <td colspan="3">Test</td>

      </tr>

      <tr>

        <td rowspan="2">Teddy</td>

        <td>Teddy</td>

        <td>12</td>

      </tr>

      <tr>

        <td>Teddy</td>

        <td>12</td>

      </tr>

    </table>

Note : Il n’est pas recommandé de concevoir un tableau avec les lignes et les colonnes se chevauchent car il s’agit d’un code HTML invalide et le résultat est géré différemment par différents navigateurs.

* 1. COLUMN GROUPS

De fois on souhaite appliquer du style à un groupe de colonnes ou à des fins de sémantique ; on peut regrouper les colonnes ; pour se faire on utilise la balise <colgroup> et <col>

<table>

      <!-- colgroup -->

      <colgroup id="MyColumn">

        <col id="MyCellue" />

        <col />

        <col />

      </colgroup>

      <!-- content -->

      <tr>

        <th>Nom</th>

        <th>Prenom</th>

        <th>Age</th>

      </tr>

      <tr>

        <td>Teddy</td>

        <td>Walter</td>

        <td>29</td>

      </tr>

      <tr>

        <td>Teddy</td>

        <td>Walter</td>

        <td>29</td>

      </tr>

      <tr>

        <td>Teddy</td>

        <td>Walter</td>

        <td>29</td>

      </tr>

    </table>

* 1. TABLE WITH thead, tbody, tfoot and caption

HTML fournit encore d’autre éléments de table avec <thead>, <tbody>, <tfoot> et <caption>. Ces éléments additionnels fournissent une valeur sémantique a ton tableau

<table border="1">

      <caption>

        Table title

      </caption>

      <thead>

        <tr>

          <th>Nom</th>

          <th>Prenom</th>

          <th>Age</th>

        </tr>

      </thead>

      <tbody>

        <tr>

          <td>Teddy</td>

          <td>Walter</td>

          <td>10</td>

        </tr>

        <tr>

          <td>Teddy</td>

          <td>Walter</td>

          <td>20</td>

        </tr>

      </tbody>

      <tfoot>

        <tr>

          <td>Footer</td>

          <td>Footer</td>

          <td>Footer</td>

        </tr>

      </tfoot>

    </table>

* 1. HEADING SCOPE

<th> élément est très souvent utilisé pour indiquer le titre de tableau pour les colonnes et les lignes.

|  |  |  |
| --- | --- | --- |
|  | **Nom** | **Prénom** |
| **Heure** |  |  |
| **Date** |  |  |

<table border="1">

      <thead>

        <tr>

          <td></td>

          <th scope="col">Nom</th>

          <th scope="col">Prenom</th>

        </tr>

      </thead>

      <tbody>

        <tr>

          <th scope="row">Heure</th>

          <td>-</td>

          <td>-</td>

        </tr>

        <tr>

          <th scope="row">Date</th>

          <td>-</td>

          <td>-</td>

        </tr>

      </tbody>

    </table>

1. COMMENTAIRE

Les commentaires fournissent à d’autre développeur une information spécifique a tes codes sans affecter l’interface utilisateur.

* 1. Créer un commentaire

< !-- -->

1. CLASSES AND IDs

Class : indique la classe d’un élément et n’est pas unique

Id : indique l’ID d’un élément et est unique dans le même contexte

Les attributs class et id facilitent le référencement des éléments HTML à partir de script et de feuilles de styles.

* 1. Giving an element a class

Les class sont des identifiant pour les éléments auxquels elles sont affectées. Utilisez l’attribut class pour affecter une classe à un élément

 <div class="example-class"></div>

Assigner une multiple class a un élément, les classes doivent être séparés par les espaces.

<div class="classe1 classe2"></div>

**Utilisez les classes en CSS**

Les classes peuvent être utilisées pour stylisé certain élément sans changer tous les éléments de ce type. Deux span peuvent avoir deux style diffèrent en utilisant les classes.

<span></span>

    <span class="special"></span>

Une même classe peut être attribué à plusieurs éléments et ces derniers recevront le même style associé à cette classe. Ceci sera toujours vrai sauf si vous spécifiez l’élément dans le css.

    <div class="higlight"></div>

    <span class="higlight"></span>

Si dans notre css, nous avons **.higlight { color : green }** alors la couleur verte sera appliquée au ces deux contenus

Mais si on sélectionne div ayant une classe **higlight** alors c’est le texte du div qui pourra avoir seulement la couleur verte. **div.highlight { color : green }**

**Note : néanmoins en css il est recommandé d’utiliser généralement les classes seules (.highlight) plutôt de les utiliser les éléments avec des classes (div.highlight)**

On peut aussi chainer le sélecteur de classe pour sélectionner uniquement les éléments qui ont une combinaison de plusieurs classes.

<div class="special left menu"></div>

Ceci peut être chainer de façon

***.special.left.menu { color : pink }***

* 1. Giving an element an ID

L’attribut id d’un élément doit être uniquement sur toute la page ou sur tout le document, même si les attributs sont attachés à plusieurs à deux types d’éléments différents.

Exemple ce code ici dessous est incorrect :

    <div id="special-id"></div>

    <span id="special-id"></span>

Pour référencer un élément avec id en css, on utilise #

**#special-id { color : green }**

* 1. Acceptable values
     1. For ID

Les seules restrictions sur la valeur d’un identifiant sont :

* Doit être unique sur tout le document
* Il ne doit pas contenir aucun espace
* Il doit contenir au moins un caractère

<div class="container"></div>

    <div class="999"></div>

    <div class="#%LV-||"></div>

    <div id="\_\_\_\_V">...</div>

    <div id="⌘⌥">...</div>

    <div id="♥">...</div>

    <div id="{}">...</div>

    <div id="©">...</div>

    <div id="♤₩¤☆€~¥">...</div>

Toutes ces valeurs ci haut sont valides

<div class=" "></div>

Mais celui-ci haut est invalide

Quand dans un seul document nous retrouvons deux IDs du même nom c’est aussi invalide

* + 1. Pour les classes

Les règles pour les classes sont essentiellement les mêmes que pour un identifiant. La différence est que les valeurs de classe n’ont pas besoin d’être unique sur tout le document.

En se référant aux exemples ci-dessus, bien que cela ne soit pas valable dans le même document :

 <div id="results"></div>

    <div id="results"></div>

Le code ci haut n’est pas valide mais avec les classes ci bas, c’est bien valide.

    <div class="results"></div>

    <div class="results"></div>

*Remarque importante : comment les valeurs d’ID et classes sont traitées en dehors de HTML ?*

*Gardez à l’esprit que* les règles et les exemples ci-dessus, s’appliquent dans le contexte du HTML. L’utilisation de chiffres, de ponctuation ou de caractères spéciaux dans la valeur d’un identifiant ou d’une classe peut causer des problèmes dans d’autres contextes, tels que CSS, JavaScript et les expressions régulières.

<div id="9lions"></div>

Bien que cet identifiant soit valide dans HTML5 mais ça reste invalide dans css

Note : en css les identifiants (les IDs et les classes) ne peuvent contenir que les caractères [a-zA-z0-9] et ISO 10646 caractères U+00A0 et supérieur, plus le trait d’union et le trait de soulignement (\_) ; mais ils ne peuvent pas commencer par un chiffre, deux traits d’union ou un trait d’union suivi d’un chiffre

* 1. Problèmes lié à la duplication des IDs

Avoir plus d’un élément avec le même id dans un seul document est un problème difficile à résoudre. L’analyseur HTML essayera de rentre la page dans tous les cas. Aucune erreur ne va s’affiche mais le rythme pourrait aboutir à une page web qui se comporte très mal.

    <div id="aDiv">a</div>

    <div id="aDiv">b</div>

En css si on sélectionne l’élément div en utilisant l’id aDiv ça va bien marcher le style

#aDiv {

    color: red;

  }

Donc ça va appliquer la couleur rouge sur tous les éléments

Mais en javascript on aura du mal à sélectionner tous les éléments et par défaut ça juste sélectionner le premier élément

<script>

  var html = document.getElementById("aDiv").innerHTML;

  console.log(html);

</script>

Par défaut la console va afficher la valeur du premier élément qui est **a**

1. Data attributes
   1. Prise en charge des anciens navigateurs

Les attributs de données ont été introduit dans HTML5 qui est pris en charge par tous les navigateurs modernes mais les navigateurs plus anciens avant HTML5 ne reconnaissent pas les attributs de données.

* 1. Utilisations des attributs de données

Les attributs HTML5 data-\* offrent un moyen pratique de stocker des données dans des éléments HTML. Les données stockées peuvent être lues où modifier à l’aide du javascript.

<div data-submitted="yes" class="user-profile">....some content</div>

* La structure de l’attribut de données est **data-\*** cad que le nom de l’attribut de données vient apres la partie data-. En utlisqnt ce nom on peut avoir accès a l’attribut depuis le javascript.
* Les données au format chaine (y compris du json) peuvent etre stockée à l’aide de l’attribut data-\*
* Le nom de l’attribut ne peut pas commencer par xml,
* Le nom ne peut pas contenir de point-virgule
* Le nom ne doit pas contenir une lettre majuscule de l’alphabet latin (A-Z)

1. LINKING RESSOURCES : RESSOURCES LIEES

* charset : spécifie l’encodage de caractère du document lié
* crossorigin : spécifie comment l’élément gère les demandes d’origine croisée
* href : spécifie la location ou l’emplacement du document lié
* hreflang : spécifie la langue du texte dans le document lié
* media : spécifie sur quel appareil le document sera affiché, souvent utilisé avec la sélection de feuille de style en fonction de l’appareil en question.
* rel : c’est obligatoire et spécifie la relation entre le document actuel et le document lié
* rev : spécifie la relation entre le document lié et le document actuel
* sizes : spécifie la taille de la ressource liée. Uniquement lorsque rel=icon
* target : indique ou le document lié doit être chargé
* type : spécifie le type de support du document lié
* integrity : spécifie un hachage encodé en base64 (sha256, sha384 ou sha512) de la ressource liée permettant au navigateur de vérifier sa légitimité.

Alors que de nombreux scripts, icones et feuilles de style peuvent être écrits directement dans les balises HTML, c’est la meilleure pratique et plus encore efficace de les mettre dans leur propre dossier et de lier à votre document.

* 1. Javascript
     1. Synchronous : synchrone

La pratique la plus courant consiste à place les balises javascript <script> juste avant la fermeture de </body>. Charger le script en dernier permet au visuel de votre site de s’afficher plus rapidement et décourage votre javascript d’essayer d’interagir avec les éléments qui n’ont pas encore été chargés.

<script src="js/script.js"></script>

* + 1. Asynchronous: asynchrone

Ce chargement est nécessaire si le chargement encours n’est pas nécessaire à l’initialisation de la page. Exemple si nous avons des fonctions qui vont s’exécuter quand on exécute une action spécifique sur la page mais lors du chargement.

Le chargement asynchrone consiste à charger le fichier javascript en parallèle et une fois il est entièrement téléchargé, le navigateur va interrompre le chargement de la page HTML afin d’analyser d’abord le fichier javascript.

<script src="js/script.js" async></script>

* + 1. Deferred: chargement differé

Le chargement différé est comme le chargement asynchrone à l’exception que l’analyse du javascript ne sera qu’une fois le code HTML entièrement chargé. Les scripts différés sont garantis d’être chargé dans l’ordre de déclaration, de la même manière que les scripts synchrones.

<script src="js/script.js" defer></script>

* + 1. <noscript>

L’element noscript definit le contenu à afficher si l’utilsateur a des scripts desactivés ou si le navigateur ne le fait pas.

La balise <noscript> est de fois placé dans le <header> ou dans le <body> du document.

<noscript>Javascript desabled</noscript>

* 1. External CSS stylesheet

La pratique standard consiste à placer les balises <link> a l’intérieur de <head> en haut de votre code HTML. De cette façon le CSS sera chargé en premier et s’appliquera lors de son chargement plutôt que d’afficher du code HTML sans style jusqu’à ce que le CSS soit chargé.

L’attribut type n’est pas nécessaire dans HTML5 car HTML5 prend généralement en charge CSS.

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

Donc on peut encore

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

Une autre pratique bien sûr moins courante consiste à utiliser @import dans les balises <styles>

<style type="text/css">

    @import url("css/style.css");

  </style>

* 1. Favicon

<link rel="shortcut icon" href="favicon.ico" type="image/x-icon" />

Utilisez le type mime/ image/png pour les fichiers PNG et image/x-icon pour les fichiers d’icônes (\*.ico).

Avec cet exemple un fichier nommé favicon.ico à la racine du site web sera généralement chargé et appliqué automatiquement sans le besoin d’une balise <link>. Si jamais ce fichier change, les navigateurs peuvent être lent et têtus à mettre à jour leur cache.

<link rel="icon" type="image/png" href="/favicon.png" />

<link rel="shortcut icon" href="favicon.ico" type="image/x-icon" />

* 1. Media

<link rel="stylesheet" href="print.css" media="print" />

L’attribut media permet de précise rune feuille de style sur un media spécifique sélectionné ;

Celui spécifie que print.css sera utilisé lorsque nous voulons imprimer la page

 <link

      rel="stylesheet"

      href="css/mobile.css"

      media="screen and (max-width: 600px)"

    />

Ceci spécifie que le fichier mobile.css sera utilisé lorsque nous sommes sur appareil avec la taille max est de 600px

* 1. Web Feed **(A faire des recherches dessus : Atom xml et rss xml)**

<link

      rel="alternate"

      href="atom.xml"

      type="application/atom+xml"

      title="Atom"

    />

<link

      rel="alternate"

      href="rss.xml"

      type="application/rss+xml"

      title="RSS"

    />

1. Include Javascript Code in HTML
2. Using HTML with Css
3. LES IMAGES

* src : spécifie l’URL ou le chemin (absolu ou relatif) de l’image
* srcset : spécifie l’image à utiliser dans une situation (ex : high résolution displays, small monitors, etc)
* sizes : taille des images selon le breakpoints ( CAD les tailles des écrans)
* alt : le texte alternative lorsque il y a des problèmes de chargements de l’image
* width : spécifie la largeur de l’image (optional)
* height : spécifie la hauteur de l’image (optional)
  1. Creating an image

Pour ajouter une image sur une page, on utilise la balise <img />

La balise img n’a pas de balise de fermeture et Les deux principaux attributs sont **src** pour définir la source de l’image et **alt** qui affiche un texte alternatif qui décrit l’image.

<img src="images/profile.jpeg" alt="profile de Teddy" />

La source peut être une url relative, ou une url absolue (une url depuis internet)

<img src="https://i.stack.imgur.com/ALgZi.jpg?s=48&g=1" alt="stack over" />

Note : les images ne sont pas techniquement insérées dans la page HTML ; les images sont liées à des pages HTML. La balise <img /> crée un espace de maintien pour l’image référencée. Conseil : pour lier l’image a un autre document, imbriquez seulement la base <img> dans les balises <a>

* 1. Choisir un texte alternatif : alt

Le texte alternatif est utilisé par des lecteurs d’écran pour les utilisateurs malvoyants et par les moteurs de recherche. Il est donc important d’écrire un bon texte alternatif pour vos images.

<img src="images/anonymous.jpeg" alt="anonymous user avatar" />

Lien qui utilise les images

    <a href="https://google.com"><img src="edit.png" alt="Edit icon" /></a>

    <a href="https://google.com"><img src="delete.png" alt="Edit icon" /></a>

1. INPUT CONTROL ELEMENT

* Class : indique la classe de l’input
* Id : indique-l’ID de l’input
* Type : identifie le type de contrôle d’entrée à afficher, les valeurs acceptables sont : hidden, text, tel, url, email, password, date, time, number, range, color, checkbox, radio, file, submit, image,reset and button.
* Name : indique le nom de l’input
* Disabled : c’est une valeur booleenne indiquant que l’entrée doit être désactivée. Les contrôles désactivés ne peuvent pas être modifiés et leur valeur n’est même pas envoyé lors de la soumission du formulaire et ne peut pas recevoir le focus.
* Checked : lorsque la valeur de l’attribut type est radio ou checkbox, la présence de cet attribut booleen indique que le champ est sélectionné par défaut
* Multiple : introduit en HTML5, indique que plusieurs fichiers ou valeurs peuvent être passés. Ceci s’applique uniquement aux entrées de type file et email
* Placeholder : HTML5, une indication à l’utilisateur de ce qui peut être dans le contrôle. Ce texte ne doit pas contenir de retour a ligne ou des sauts de ligne
* Autocomplete : HTML5, indique si la valeur du champ peut être complétée automatiquement par le navigateur
* Readonly : indique que cette valeur n’est pas modifiable. Les contrôles en lecture seule sont toujours envoyés sur le formulaire de soumission, mais ne recevra pas le focus HTML5.
* Required : HTML5, indique qu’une valeur doit être présente ou que l’élément doit être coché pour que le formulaire soit soumis.
* Autofocus : l’élément <input> devrait avoir le focus lors du chargement de la page
* Value : spécifie la valeur de l’input
  + 1. **Text**

Le type texte définit un champ de texte sur une seule ligne avec les sauts de ligne automatiquement supprimés de la valeur d’entrée.

<Input> sont utilisé dans l’élément <form>

Syntaxe

    <input type="text" />

    <!-- OR -->

    <input />

Un input n’autorise qu’une seule ligne de texte. Si vous avez besoin d’une saisie de texte multiligne pour une quantité importante de texte, utiliser un élément <textarea> à la place.

* + 1. **Chekbox and Radio**

Checkbox et radio sont écrits avec la balise <input>, et leur comportement est défini dans le HTML

La case à cocher ou le bouton radio le plus simple est un élément input avec un attribut type radio ou checkbox.

    <input type="checkbox" />

    <input type="radio" />

Un seul élément case à cocher autonome est utilisé pour une seule option binaire telle qu’une question oui ou non.

Les checkboxs sont indépendant cela signifie que l’utilisateur peut sélectionner autant de choix qu’il le souhaite dans un groupe de case à cocher. En d’autre terme la case à cocher ou checkbox ne décoche pas les autres checkbox du groupe de checkbox.

Le bouton radio viennent généralement en groupe, et identifié en utilisant le même attribut name sur tous les boutons de ce groupe. La sélection d’un bouton radio s’exclue mutuellement ce qui signifie que l’utilisateur ne peut sélectionner qu’un seul choix parmi un groupe de bouton radio. Lorsqu’un bouton radio est sélectionné tous les autres portant les mêmes noms deviennent décoché.

    <input type="radio" name="p" id="red" value="#FFF" />

    <input type="radio" name="p" id="green" value="#FFF99" />

    <input type="radio" name="p" id="blue" value="#FFF877" />

*Les attributs*

Les checkbox et radio ont un certain nombre d’attributs pour Controller leur comportement

* Value : cet attribut spécifie la valeur associée au bouton lors de la soumission du formulaire. Ces boutons ont une particularité, lorsque la valeur est omise, par défaut ils envoient ON au lieu d’envoyer une valeur vide.
* Checked : l’attribut checked spécifie l’état initial d’un checkbox ou d’un bouton radio, c’est un attribut booléen et peut être omis.

Toutes ces valeurs sont valides

    <input type="radio" checked />

    <input type="radio" checked="" />

    <input type="radio" checked="checked" />

    <input type="radio" checked="ChEcKeD" />

*L’accessibilité*

* Labels : pour donner un contexte aux boutons ou input et montrer aux utilisateurs à quoi sert chaque bouton ; chacun doit avoir une étiquette (label). Cela peut être fait en utilisant un élément <label> pour envelopper le bouton. En outre cela rend l’étiquette cliquable.

<label > <input type="radio" name="p" id="red" value="#FFF" /> Red </label>

    <label>

      <input type="radio" name="p" id="blue" value="#FF00F" /> Blue

    </label>

Ça peut que l’élément label est séparé de l’élément input, alors on utilise l’attribut for pour le lié

* + 1. Input validation

La validation des inputs se fait automatiquement par le navigateur en fonction des attributs que l’on a renseigné dans l’input. Ces validations peuvent partiellement ou complètement remplacer la validation des inputs par JavaScript.

Ce genre de validation peuvent être contourné via des requêtes http spécialement conçues, de sorte que cette validation ne doit pas remplacer la validation coté serveur.

La validation se produit uniquement lors de la tentative de soumission du formulaire ; donc tous les inputs doivent être à l’intérieur d’un formulaire <form> pour que la validation soit effectuée sauf si on utilise JavaScript.

Les inputs disabled and readonly ne sont pas soumis dans le processus de validation. Certains types d’input comme (email, url, tel, date, et bien d’autre) sont automatiquement validés et ne nécessitent pas vos propres contraintes de validation.

* Required : utilizer l’attribut required indique qu’un champ doit être rempli afin de réussir la validation

<input type="text" name="" id="" required />

* Minimum / Maximum length : utiliser les attributs minlength et maxlength pour indiquer les exigences de longueur. La plus part de navigateur empêcheront l’utilisateur de taper plus que le nombre maximum de caractères dans l’input.

<input type="text" name="" id="" maxlength="10" minlength="5" />

* Specifying a range

Utiliser les attributs min et max pour restreindre la plage de nombre qu’un utilisateur peut saisir dans un input de type number or range

      <input type="number" name="" id="" min="3" max="10" />

      <input type="range" name="" id="" min="3" max="10" />

* Accept File type

Pour les champs de saisie de type fichier, il est possible de n’accepter que certains types de fichiers, telles que des vidéos, des images, des audois, des extensions de fichier spécifiques ou certains types media.

      <input

        type="file"

        name=""

        id=""

        accept="image/\*, .rar, application/zip"

        title="Seulement images are allowed"

      />

NB : l’ajout de l’élément novalidate dans l’élément de formulaire <form> ou de l’attribut fromnovalidate au bouton submit, empêche la validation sur les éléments de formulaire.

    <form action="#" novalidate>

      <input type="text" name="" id="" required />

      <input type="text" name="" id="" required />

      <input type="text" name="" id="" required />

      <input type="text" name="" id="" required />

      <input type="submit" value="Send" formnovalidate />

    </form>

* + 1. Color

Dans le navigateur supporté un élément input du type color ; crée un contrôle de couleur dont la valeur est l’hexadécimal de la couleur sélectionnée

<input type="color" name="" id="" value="#ff0000" />

* + 1. Password

L’input avec le type password crée un champs texte sur une seule ligne comme celui du type text sauf que le texte n’est pas affiché lors que l’utilisateur le saisit.

<input type="password" name="password" id="" placeholder="Mot de passe" />

* + 1. File

Les input type file permettent aux utilisateurs de sélectionner un fichier à partir de leur système de fichier local pour l’utiliser dans la page en cours. C’est utiliser dans un formulaire, probablement ça va nous aider à télécharger le fichier sur le serveur.

L’exemple permet aux utilisateurs de sélectionner un fichier dans leur system de fichiers et de l’envoyer sur le serveur en faisant le traitement sur le fichier upload\_file.php

    <form action="uplad.php" method="post" enctype="multipart/form-data">

      Select file to upload

      <input type="file" name="fileSubmission" id="fileSubmission" />

      <input type="submit" value="Telecharger" />

    </form>

* Multiple file

Pour télécharger plusieurs fichiers au même moment, on utilise l’attribut multiple qui va permettre à l’utilisateur de sélectionner multiple fichiers

    <form action="uplad.php" method="post" enctype="multipart/form-data">

      Select file to upload

      <input type="file" name="fileSubmission" id="fileSubmission" multiple />

      <input type="submit" value="Telecharger" />

    </form>

* Accept Files

L’attribut accept spécifie les types de fichiers que l’utilisateur peut selectionner.

Cfr validation

* + 1. Buttons

Les boutons peuvent être utilise pour déclencher des actions sur la page, sans soumettre le formulaire. On peut aussi utiliser l’élément <button> si on a besoin d’un bouton qui peut etre stylé plus facilement ou qui peut contenir d’autre élément.

<button type="button">CLICK ME</button>

Le bouton est typiquement utilisé avec un évènement “onClick”

    <button type="button" onclick="alert('My first hello')">SAY HELLO</button

    ><br />

    OU <br />

    <input type="button" value="Say Again" onclick="alert('Good')" />

[Attributs]

* Name : le nom du bouton qui sera envoyé dqns les données du formulaire
* Type : le type du bouton

[type ]

* Submit : c’est la valeur par défaut si l’attribut type n’est spécifié
* Reset : permet de réinitialiser tous les inputs à leur valeur initiale
* Button : le bouton n’a pas de comportement par défaut, il peut avoir de script coté client associés aux évènement de l’élément qui se déclenche lorsque l’évènement se produit.
* Menu :

[value]

C’est la valeur initiale du bouton

* + 1. submit

un input ou un bouton de type submit crée un bouton qui soumet le formulaire dans lequel il se trouve lorsqu’il est cliqué.

 <input type="submit" value="Send" />

On peut aussi utiliser un élément button à la place, de l’input si on veut bien styliser le bouton d’envoie peut être avec une image.

      <button type="submit">

        <img

          src="images/profile.jpeg"

          alt="submit button"

          width="20"

          height="20"

        />

      </button>

* + 1. Reset

Un input ou un bouton du type reset, permet de créer un bouton qui va réinitialiser toutes les valeurs des inputs par défaut dans un formulaire.

<input type="reset" name="reset" id="reset" value="Reinitiailiser" />

* Le texte d’un input type text sera réinitialisé à vide ou à sa valeur par défaut spécifié a l’aide de l’attribut value
* Tous les checkbox et le radio seront désélectionné à moins qu’ils n’aient par défaut l’attribut checked

Note : un bouton ou un input du type reset doit être à l’intérieur ou attaché à un formulaire afin d’avoir effet. Le bouton va réinitialiser uniquement les éléments du formulaire.

* + 1. Hidden

Un input avec type hidden ne sera pas visible par l’utilisateur mais sa valeur sera envoyée au serveur lors de la soumission du formulaire

<input type="hidden" name="hidd" id="hidd" />

* + 1. Tel

L’input avec type tel, représente un input en texte d’une ligne pour la saisie d’un numéro de téléphone

<input type="tel" name="numero" placeholder="Numero de telephone" />

* + 1. Email

Cet input avec type email, doit être utiliser pour le champ de saisie qui contiennent une adresse email

<input type="email" name="email" id="email" placeholder="email" />

* + 1. Number

L’élément input avec type number représente un contrôle précis qui doit prendre pour valeur un nombre et non le texte.

<input type="number" name="numero" id="numero" value="0" />

* + 1. Range

Un input pour saisir un nombre dont la valeur exacte n’a pas d’importance.

<input type="range" min="0" max="100" step="2" />

Min : valeur minimum pour l’intervalle

Max : valeur maximum pour l’intervalle

Step : montant à augmenter à chaque incrémentation

* + 1. Search

L’input du type search est utilisé pour la recherche textuelle. Il ajoutera un symbole de loupe à côté de l’espace pour le texte sur la plupart des navigateurs.

<input type="search" name="search" id="search" />

* + 1. Image

      <input

        type="image"

        src="images/profile.jpeg"

        alt="profile image"

        width="50"

        height="50"

      />

* + 1. Week

En fonction de la prise en charge du navigateur, un contrôle pour entrer un numéro semaine-année et un numéro de semaine sans fuseau horaire.

<input type="week" name="semaine" id="semaine" placeholder="the week" />

* + 1. Url

Ceci est utilisé pour les champs de saisie qui doivent contenir une adresse URL. Selon la prise en charge du navigateur, le champs URL peut être automatiquement validé lors de la soumission.

Certains smartphones reconnaissent le type d’URL et ajoutent .com au clavier pour correspondre à la saisie de l’URL.

<input type="url" name="url" id="url" placeholder="Tapez ton url" />

* + 1. DateTime-Local

En fonction de navigateur, un sélecteur de date et d’heure apparaitra à l’écran pour vous permettre de choisir une date et une heure.

<input type="datetime-local" name="datetime" id="datetime" />

* + 1. Month

En fonction de navigateur, un contrôle s’affichera pour choisir le mois

<input type="month" name="mois" id="mois" />

* + 1. Time

<input type="time" name="temps" id="temps" />

* + 1. Date

Un sélecteur de date apparaitre à l’écran pour vous permettre de choisir une date, ceci n’est pas pris en charge dans FireFox ou Internet.

<input type="date" name="date" id="date" />

1. LES FORMULAIRES

* accept-charset : spécifie l’encodage de caractère à utiliser
* Action : spécifie où envoyer les données de formulaire lors de la soumission
* autocomplete : spécifie si un formulaire doit avoir la saisie semi-automatique activé ou désactivé
* enctype : spécifie comment les données du formulaire doivent être encodées lors de leur soumission au serveur (uniquement pour les méthodes POST)
* method : spécifie la méthode http à utiliser lors de l’envoie de données de formulaire (POST ou GET)
* name : spécifie le nom du formulaire
* novalidate : spécifie que le formulaire ne doit pas être validé lors de la soumission
* target : indique ou afficher la réponse reçue après la soumission du formulaire

Afin de pouvoir regrouper les éléments d’input et de soumettre les données, HTML utilise l’élément form pour encapsuler l’entrée et la soumission des données. Ces formulaires gèrent l’envoi des données dans une méthode spécifique à une page gérée par le serveur.

* 1. Submitting
* **Attribut action**

L’attribut action définit l’action à effectuer lors de la soumission du formulaire, ce qui conduit généralement à un script qui recueille les informations soumises et les exploiter. Si vous laissez vide alors il l’enverra dans le même fichier.

<form action="action.php"></form>

* Attribute method

L’attribut method est utilisé pour définir la méthode http du formulaire qui est soit Get ou POST

      <form action="action.php" method="get"></form>

      <form action="action.php" method="post"></form>

La méthode GET est principalement utilisée pour obtenir les données, par exemple pour recevoir un message par son identifiant ou son nom ou soumettre une requête recherche.

La méthode GET ajoutera les données du formulaire à l’url spécifié dans l’attribut action.

[www.example.com/action.php](http://www.example.com/action.php) ? firstname=Mickey & lastname = Mouse

La méthode POST est utilisée lors de la soumission de données à un script. La méthode POST n’ajoute pas les données du formulaire à l’URL de l’action mais envoie en utilisant le corps de la requête.

Pour soumettre correctement les données du formulaire, un nom d’attribut de name doit être spécifié.

<input type="text" name="lastname" value="Mouse" />

* More attribute

<form

        action="action.php"

        method="POST"

        target="\_blank"

        accept-charset="UTF-8"

        enctype="multipart/form-data"

        autocomplete="off"

        novalidate>

</form>

* 1. Attribute target

L’attribut submit spécifie là où la réponse du formulaire sera affichée après avoir soumis le formulaire.

<form action="action.php" target="\_blank"></form>

* 1. Uploading Files

Les images et les fichiers peuvent être envoyés au serveur en définissant l’attribut enctype = multipart/form-data.

L’enctype spécifies juste comment les données du formulaire seront encodées lors de la soumission au serveur.

    <form action="action.php" enctype="multipart/form-data">

      <input type="file" name="pic" id="pic" />

      <input type="submit" value="Upload" />

    </form>

* 1. Grouper les champs inputs

Lors de la conception d;un formulaire, vous aimeriez peut être regrouper quelques inputs pour aider à organiser ou à ameliorer la mise en page du formulaire. Cela peut être fait en utilisant la balise fielset and legend pour le titre du groupe.

    <form action="action.php">

      <fieldset>

        <legend>Informations personnelles</legend>

        <label for="nom">NOM</label>

        <input type="text" name="nom" id="nom" /> <br />

        <label for="pnom">PRENOM</label>

        <input type="text" name="pnom" id="pnom" />

      </fieldset>

      <fieldset>

        <legend>Informations personnelles</legend>

        <label for="nom">NOM</label>

        <input type="text" name="nom" id="nom" /> <br />

        <label for="pnom">PRENOM</label>

        <input type="text" name="pnom" id="pnom" />

      </fieldset>

      <input type="submit" value="ENREGISTRER" />

    </form>

1. **Div** Element

L’élément Div en HTML est un élément conteneur qui encapsule d’autre éléments et peut être utilisé pour regrouper et séparer certaines parties d’une page web.

Un Div lui-même ne représente rien mais est un outil ou un élément très puissant dans la conception web.

* 1. Utilisation Basic

L’élément div en soit n’a pas de signification sémantique, ça représente tout simplement une division et est généralement utilisé pour regrouper et encapsuler d’autres éléments dans la page HTML et les séparer des autres groupes de contenu.

En tant que tel, chaque <div> est mieux décrit par son contenu.

    <div>

      <p>Je suis un paragraph dans un div</p>

    </div>

L’élément div est généralement un élément du type bloc, ce qui signifie qu’il sépare un bloc d’un document HTML et occupant la largeur maximale de la page. Par défaut le navigateur donne une règle CSS par défaut : ***display : block;***

Note : il fortement recommandé par le W3C de considérer le div comme un élément de dernier recours, lors qu’aucun autrement élément ne convient. L’utilisation d’élément plus appropriés au lieu de l’element div conduit à une meilleure accessibilité pour les lecteurs d’ecran et une maintenabilité plus facile pour les autres développeur ou auteurs.

Par exemple un article de blog serait balisé à l’aide d’un élément article, un chapitre a l’aide d’une section, les aides à la navigation d’une page en utilisant **nav**, et un groupe de contrôle de formulaire en utilisant **<fieldset>**

Les éléments div peuvent être utiles à des fins stylistiques, ou pour envelopper plusieurs paragraphes dans une section qui doivent tous être annoté de la même manière.

* 1. NESTING

C’est une pratique très courante de placer un div à l’intérieur d’un autre div. Ceci est généralement appelé imbrication des éléments et permet de diviser davantage les éléments en sous sections ou d’aider les développeurs avec le style css.

    <div class="outer-div">

      <div class="inner-div">

        <p>This a paragraph</p>

      </div>

      <div class="inner-div">

        <p>This is an other paragraph</p>

      </div>

    </div>

* **Nesting inline and block éléments :** quand vous faites l’imbrications des éléments vous devez garder en tête qu’il y a des éléments inline et des éléments en bloc.

Les éléments en bloc ajoutent un saut de ligne en background, CAD les éléments imbriqués viendront toujours à la ligne suivante,

Les éléments inline peuvent être positionnés les uns à côté des autre par défaut.

* Eviter l’imbrication profonde de <div> : une mise en page de conteneur imbriquée profonde montre souvent un mauvais style de codage. De fois la mises en place des certains coins arrondis ou certaines autres fonctions créent souvent un tel code HTML. Pour la plupart de la dernière génération de navigateur, il existe des homologues CSS3. Essayer d’utiliser le moins possible d’élément HTML.

Note : un div ne doit pas être imbriqué à plus de 6 niveau.

1. Les ELEMENTS DE SECTIONS
   1. Nav

L’élément nav est principalement destiné à être utilisé pour les sections qui contiennent les principaux blocs de navigations pour le site web.

Cela peut même inclure des liens vers d’autres parties de la page web (par exemple, des ancres, ou d’autre page entièrement)

* **Inline items**

Ce qui affichera un ensemble d’hyperliens en ligne

    <nav>

      <a href="https://google.com">Google</a>

      <a href="https://yahoo.com">Yahoo</a>

      <a href="https://bing.com">Bing</a>

    </nav>

* **Utiliser les éléments de la liste si nécessaire**

Si le contenu représente une liste d’élément, utilisez un élément de liste pour afficher et améliorer l’expérience utilisateur.

Note le rôle navigation

 <nav role="navigation">

      <ul>

        <li><a href="https://google.com" rel="external">Google</a></li>

        <li><a href="https://google.com" rel="external">Google</a></li>

        <li><a href="https://google.com" rel="external">Google</a></li>

      </ul>

    </nav>

* Eviter les utilisations inutiles

L’élément footer de la page peut avoir une liste de liens vers d’autres parties du site (Faq, etc), l’element footer seul est suffisant dans ce cas, vous n’avez pas besoin d’envelopper davantage vos liens avec un élément nav

<!-- UTILISATION INUTILES -->

    <footer>

      <nav>

        <a href="https/google.com">Google</a>

      </nav>

    </footer>

    <!-- LE FOOTER SEUL SE SUFFIT -->

    <footer>

      <a href="https/google.com">Google</a>

    </footer>

NOTES :

<main> element ne doit pas etre utiliser dans la balise <nav>

Ajouter un role ARIA role= navigation à l’élément <nav> est conseillé pour aider certains agents utilisateurs (navigateurs, lecteur d’écran) qui ne supporte pas HTML5 et fourni aussi plus de contexte a ceux qui le font.

Lecteur d’écran ou screen reader : sont des logiciels permettant aux utilisateurs aveugles ou malvoyants de naviguer sur le site.

* 1. Header

L’élément header représente le contenu d’introduction du contenu de son ancêtre le plus proche, ou de la partie d’un élément racine. Le header contient typiquement un groupe du contenu introductif ou de navigation.

Note : cet élément ne sectionne pas le contenu, il introduit plutôt de nouvelle section.

    <header>

      <p>Welcome to...</p>

      <h1>Voidwars</h1>

    </header>

Un header peut etre utilisé dans un article pour introduire le contenu de l’article

    <article>

      <header>

        <h1>FLEX BOX : The definitive guide</h1>

      </header>

      <p>

        Lorem ipsum dolor sit amet consectetur adipisicing elit. Ipsum eum

        voluptatum possimus beatae tempora? Voluptate at reiciendis suscipit

        impedit laboriosam veritatis dolorem repudiandae fuga ratione, odio

        nisi, rem, amet incidunt.

      </p>

    </article>

* 1. Main

L’élément <main> contient le contenu principal de la page web, ce contenu est unique à la page individuelle, et ne doit pas apparaitre ailleurs sur la page. Les éléments comme <header>, <footer>, <nav> doivent être placé en dehors de <main>.

* L’élément <main> ne doit jamais être utilisé qu’une seule fois sur une seule page.
* L’élément <main> ne doit jamais être inclus comme descendant de <article>, <aside>, <footer>,<header>, ou <nav>.

Dans l’exemple nous affichons un seul article de blog et les commentaires

<header>

      <nav role="navigation">....</nav>

    </header>

    <main>

      <h1>Individual Blog Post</h1>

      <p>Lreom</p>

      <article>

        <h2>References</h2>

        <p>.......</p>

      </article>

      <article>

        <h2>Commentaires</h2>

        <p>Les commentaires</p>

      </article>

    </main>

    <footer>.....</footer>

* Le blog post est contenu dans l’élément <main> pour indiquer qu’il s’agit du contenu principal de cette page et donc unique.
* Les balises <header> et <footer> sont sœurs de l’element <main>

Notes : la specification HTML5 reconnait l’element <main> comme un element de regroupement et non comme un element de section.

* Le role principal est role = main. Vous pouvez donner l’attribut role = main à d’autre element qui sont destiné à etre utilisé comme contenu principal pour aider les agents utilisateurs qui ne prennent pas en charge html5 et egalement pour fournir plus de contexte à ce qui le font.
* L’element <main> a par defaut le role=main et il n’est pas nécessaire à le fournir de nouveau
  1. Footer

L’element footer contient la partie pied de la page

    <footer>

      <a href="https/google.com">Google</a>

    </footer>

* 1. Section

L’élément section représente une section générique pour regrouper thématiquement le contenu. Chaque élément de section généralement devrait pouvoir être identifié avec un élément d’entête (de titre) comme enfant de la section.

* On peut utiliser une section à l’interieur d’un article et vice versa
* Chaque section doit avoir un titre pour definir la region
* N’utiliser pas section comme conteneur de style géneral

Dans l’exemple suivant nous affichons un seul article de blog avec plusieurs chapitres et chaque chapitre est une section (un ensemble de contenu regroupés par theme , qui peuvent etre identifiés par les elements de titre dans chaque section).

    <article>

      <header>

        <h2>BLOG POST</h2>

      </header>

      <p>Introduction du post</p>

      <section>

        <h3>Chapter 1</h3>

        <p>Description</p>

      </section>

      <section>

        <h3>Chapter 2</h3>

        <p>Description</p>

      </section>

      <section>

        <h3>Chapter 3</h3>

        <p>Description</p>

      </section>

    </article>

* 1. Article

Un element article contient du contenu autonome comme des articles, des articles de bog et des commentaires d’utilisateurs.

Un blog (section) avec plusieurs publications (articles) et commentaires (artilces) peut ressembler a ceci :

<section>

      <article>

        <header>

          <h1>BLOG POST</h1>

          <time datetime="2016-04-04">13th March 2016</time>

        </header>

        <p>

          Lorem ipsum dolor sit amet, consectetur adipisicing elit.

          Consequuntur, quisquam? Laboriosam suscipit

        </p>

        <p>

          corrupti placeat earum quaerat pariatur officiis corporis, aliquam,

          dolorum officia dolores saepe, voluptate sit laborum! Alias, explicabo

          aut!

        </p>

        <section>

          <h2>Comments <small>related to Blog Post</small></h2>

          <article>

            <p>Excellent</p>

          </article>

        </section>

      </article>

    </section>

Note : l’élément main ne doit pas être descendant de l’élément article

1. Navigation Bars