

*Web : principes  
de base  
(HTML/CSS)*

DA1  
Henallux

# Module 2

# Le langage HTML

# Au programme...

## ➤ Le langage HTML

- *Qu'est-ce que c'est exactement ?*
- *Syntaxe du HTML*

## ➤ Focus sur quelques cas spécifiques

- *Balises <img/> et <a>*
- *Attributs généraux*

## ➤ Les métadonnées

- *Kézako ?*
- *L'en-tête d'un document HTML*

## ➤ Le point sur les URL

- *Format des URL*
- *URL relatives et URL absolues*



# Le langage HTML

*Au programme de ce chapitre...*

## ➤ **Qu'est-ce que c'est exactement ?**

- *Quel type de langage est-ce ?*


## ➤ **Syntaxe du HTML**

- *Principes généraux*

---

Ensuite : *Focus sur quelques cas spécifiques du HTML*

# Qu'est-ce c'est exactement ?

- Langage HTML = langage de programmation ?
  - HTML est un langage de balisage (markup language).
    - = langage permettant de spécifier la structure d'un document, de sorte qu'elle puisse être ensuite utilisée par un programme (pour analyse, visualisation...)
  - Ce n'est pas un langage de programmation.
- C'est un descendant du XML.
  - XML = eXtensible Markup Language
    - Langage générique (chaque application définit ses balises)
  - Exemple :

```
<examen bloc="1" cours="web">  
  <question>Que signifie le X de XML ?</question>  
  <réponse nbPoints="3">extensible</réponse>  
</examen>
```

# Syntaxe : exemples tirés du labo

Balise ouvrante et balise fermante :

```
<h1>La Chimay Bleue</h1>
```

ou balise unique :

```
<hr/>
```

Ajout d'attributs :

```

```

```
<meta charset="utf-8" />
```

Imbrication :

```
<body>
  <table>
    <tr>
      <th>Titre</th>
    </tr>
    <tr>
      <td>Élément</td>
    </tr>
  </table>
</body>
```

# Syntaxe : indentation

```
<h1>Récolter</h1>
<p>Avec un <strong>outil</strong>, vous
obtenez les ressources suivantes :</p>
<ol>
  <li>du bois,</li>
  <li>de la pierre, </li>
  <li>des diamants.</li>
</ol>
<table>
  <caption>Ressources minées</caption>
  <tr>
    <th>Ressource</th>
    <th>Couche</th>
  </tr>
  <tr>
    <td>Charbon</td>
    <td>toutes</td>
  </tr>
</table>
```

## Indentation :

- Faire ressortir la structure
- Aligner balise ouvrante/fermante
- ... quand c'est pertinent

```
<!DOCTYPE html>           Pour éviter les
<html>                     indentations excessives
<head>
  <meta charset="utf-8" />
  <title>Ma page</title>
</head>
<body>
  <h1>Mon titre</h1>
  <p>Mon contenu</p>
</body>
</html>
```

# Syntaxe du HTML

- Syntaxe générale des balises (= tags) :

```
<tag attr="val" attr="val"> contenu </tag>  
<tag attr="val" attr="val" />
```

- Nom de la balise entièrement en minuscules
- Ne pas oublier de fermer les balises (uniques ou pas)
- Rappels : pas de balises croisées !

- Syntaxe générale des attributs (= attributes) :

```
attr="val"
```

- Nom de l'attribut entièrement en minuscules
- Suivi de = puis de la valeur entre guillemets
- Valeurs : toujours entre guillemets (même si numériques !)

# Syntaxe du HTML

- Cas particulier des attributs booléens :

`<tag attr> contenu </tag>`      `<tag attr />`

- On indique juste le nom de l'attribut pour lui donner la valeur "vrai".
- Seul moyen de donner la valeur "fausse" : ne rien indiquer.
- Autres écritures "standard" pour les attributs booléens vrais :  
     `attr=""`                      `attr="attr"`

- Exemples

<code>&lt;audio autoplay</code>	<code>src="song.mp3"&gt;&lt;/audio&gt;</code>	} true
<code>&lt;audio autoplay=""</code>	<code>src="song.mp3"&gt;&lt;/audio&gt;</code>	
<code>&lt;audio autoplay="autoplay"</code>	<code>src="song.mp3"&gt;&lt;/audio&gt;</code>	
<code>&lt;audio</code>	<code>src="song.mp3"&gt;&lt;/audio&gt;</code>	— false



# Focus sur quelques cas spécifiques

*Au programme de ce chapitre...*

## ➤ **Balises déjà rencontrées**

- *Le point sur ce qui a été utilisé dans le labo 1*

## ➤ **Insertion d'une image avec `<img/>`**

- *Et quelques informations sur les formats d'images*

## ➤ **Insertion d'un hyperlien avec `<a>`**

- *Hyperlien vers une autre page ou au sein de la même page*

## ➤ **Attributs généraux**

- *Quelques attributs utilisables avec toutes les balises*

---

Ensuite : *Les métadonnées*

# Balises déjà rencontrées

## Structure du document

<code>&lt;html&gt;</code>	tout le document
<code>&lt;head&gt;</code>	en-tête/métadonnées
<code>&lt;body&gt;</code>	corps/contenu

## Métainformations

<code>&lt;title&gt;</code>	titre du document
<code>&lt;meta /&gt;</code>	encodage (et plus)

## Texte

<code>&lt;strong&gt;</code>	texte important
<code>&lt;em&gt;</code>	texte en emphase
<code>&lt;br/&gt;</code>	passage à la ligne forcé

## Contenu

<code>&lt;h1&gt; &lt;h2&gt;...</code>	titres (jusqu'à <code>&lt;h6&gt;</code> )
<code>&lt;p&gt;</code>	paragraphe
<code>&lt;hr/&gt;</code>	ligne horizontale
<code>&lt;img/&gt;</code>	image

## Information structurée

<code>&lt;table&gt;</code>	tableau
<code>&lt;tr&gt;</code>	ligne
<code>&lt;td&gt; &lt;th&gt;</code>	cellule (titre)
<code>&lt;caption&gt;</code>	légende
<code>&lt;ul&gt; &lt;ol&gt;</code>	(un)ordered list
<code>&lt;li&gt;</code>	élément (liste)

# Insertion d'une image : `<img/>`

- Préciser le fichier source :

```

```



- Description de l'image (accessibilité, problème de téléchargement...) :

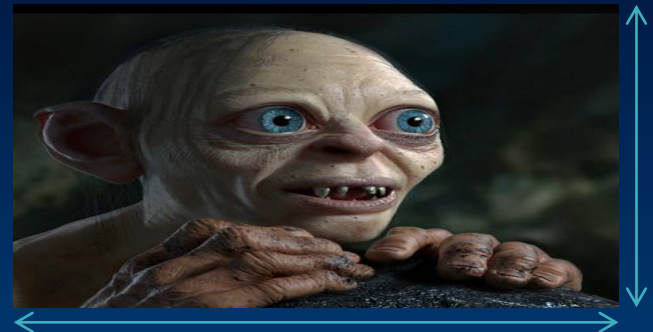
```

```

- Préciser la taille (en pixels) :  
*méthode temporaire avant CSS !*

```

```



# Insertion d'une image : `<img/>`

- Types d'images : **raster** vs **vectoriel**
  - Raster : trame de pixels (colorés)
    - Plus de nuances (couleurs, formes)
  - Vectoriel : description géométrique (courbes...)
    - Plus léger, lignes nettes, peut être redimensionnée sans perte de qualité
- Quelques autres critères :
  - Gestion de la **transparence** ou non
  - Compression avec **perte** ou non
  - Efficacité de la compression (**poids**)
  - Permet l'**animation** ou pas



raster,  
sans  
transparence



Image détournée  
avec transparence



Site pour détourer des images :  
<https://www.remove.bg/>

# Insertion d'une image : `<img/>`

Aperçu de quelques formats d'images :

Format	Raster ou Vectoriel	Transparence	Animation	Note
<b>JP(E)G</b>	Raster	non	non	Excellente compression, mais avec perte de précision
<b>PNG</b>	Raster	oui	non	Meilleure qualité mais fichier plus lourd
<b>WebP</b>	Raster	oui	oui	Meilleure compression que JPEG et PNG mais support plus limité?
<b>GIF</b>	Raster	oui	oui	Possibilités limitées (par exemple max 256 couleurs)
<b>SVG</b>	Vectoriel	oui	oui	

# Insertion d'un hyperlien : `<a>`

- Lien vers une autre page HTML :

Cliquez [ici](#) !

Cliquez `<a href="http://www.henallux.be">ici</a>` !

- À l'intérieur de la balise : texte (ou image ou ...) servant de lien.
- href = hypertext reference
- Pour ouvrir cette nouvelle page...
  - dans un nouvel onglet :  
`<a href="..." target="_blank">...</a>`
  - dans la même page (option par défaut) :  
`<a href="..." target="_self">...</a>`

# Insertion d'un hyperlien : `<a>`

- Lien vers un autre endroit sur la page :
  - Créer une ancre à la destination (ancre = point de repère invisible) :  
`<a id="hautDePage"></a>`
  - Créer le lien :  
`<a href="#hautDePage">Vers le haut de la page</a>`
- Lien vers une section spécifique sur une autre page :  
`<a href="menu.html#dessert">Voir le dessert</a>`



# Attributs généraux en HTML

= qui peuvent être utilisés avec n'importe quelle balise

- Donner un nom (qui doit être unique) à un élément HTML

Identificateur : `<a id="hautDePage">...</a>`

- Préciser la langue d'un élément HTML (ou de toute la page)

Langage : `<html lang="fr">...</html>`

- Infobulle au survol d'un élément HTML

Tooltip : `<h2 title="tuyau pour l'examen !">...</h2>`

- Deux autres attributs généraux revus plus tard

Classe : `<li class="important">...</li>`

Style : `<p style="font-weight: bold">...</p>`

**Balises génériques**

Tuyau pour l'examen



# Les métadonnées

*Au programme de ce chapitre...*

## ➤ **Kézako ?**

- *De quoi s'agit-il ? À quoi ça sert ?*

## ➤ **L'en-tête d'un document HTML**

- *Quelques métadonnées qu'on peut y placer*

---

Ensuite : *Le point sur les URL*

# Métadonnées... kézako ?

- Métadonnées (**metadata**) = métainformations
  - = Données/informations sur les données / le contenu
  - Exemple : langue d'un texte, encodage du fichier...
- Les métadonnées concernant (tout) le contenu d'un document HTML sont placées **dans son en-tête**.
- Utilité :
  - **Référencement** : analyse par les fureteurs (bots qui parcourent le web pour construire les bases de données des sites de recherche)
  - Lecture correcte du contenu (par exemple langue pour text-to-speech)

```
<!doctype html>
<html>
  <head>
    ...
  </head>
  <body>
    ...
  </body>
</html>
```

*En-tête*

# L'en-tête d'un document HTML

Quelques-unes des métadonnées qu'on peut y placer...

- Titre de la page (affiché dans l'onglet du navigateur et utilisé pour les bookmarks)

Titre : `<title>Mes sites web préférés</title>`

- Langue du document

Langue : `<html lang="en">...</html>`

- Plusieurs autres options sont regroupées sous les balises
  - `<meta/>` : informations  
(souvent avec les attributs `name` et `content`)
  - `<link/>` : lien avec un fichier  
(avec les attributs `rel` et `href`).

# Les métadonnées sous `<meta/>`

- Encodage des caractères

Encodage : `<meta charset="utf-8" />`

- Auteur et description de la page

Auteur : `<meta name="author" content="Jules Simon" />`

Description : `<meta name="description" content="..." />`

- Description utilisée par les fureteurs (détection de mots-clefs).
  - Ancienne option `name="keywords"` plus vraiment conseillée car la plupart des fureteurs l'ignorent (suite à des abus).
- Définir la manière dont la page sera affichée sur mobiles

Viewport : `<meta name="viewport" content="..." />`

- Abordé plus loin, dans le module sur le web responsive.

# Les métadonnées sous `<link/>`

- Icône (utilisé par le navigateur et pour les bookmarks)

Icône : `<link rel="icon" href="monicone.ico" type="image/x-icon" />`

- L'attribut type précise le format de l'image.
- Généralement format ico (ou png ou gif sur certains navigateurs).

- Fichier de styles CSS

Style : `<link rel="stylesheet" href="messtyles.css" />`

- Revu dans les modules suivants

- Tant pour `<meta/>` que pour `<link/>`, il existe de nombreuses autres options d'utilisation.

Voir <https://developer.mozilla.org> pour plus d'info.

# Le point sur les URL

*Au programme de ce chapitre...*

## ➤ **Format d'écriture d'une URL**

*De quoi est composée une URL ?*

## ➤ **URL absolues et URL relatives**

*Si on reste à l'intérieur d'un même "site", on peut raccourcir l'écriture de l'URL et préciser comment atteindre la cible à partir du document HTML courant.*

# Format d'écriture d'une URL

Une URL se compose de plusieurs parties :

`http://www.monsite.be/vacances/images/plage.jpg`

`http://`

protocole

`www.monsite.be`

nom de  
domaine

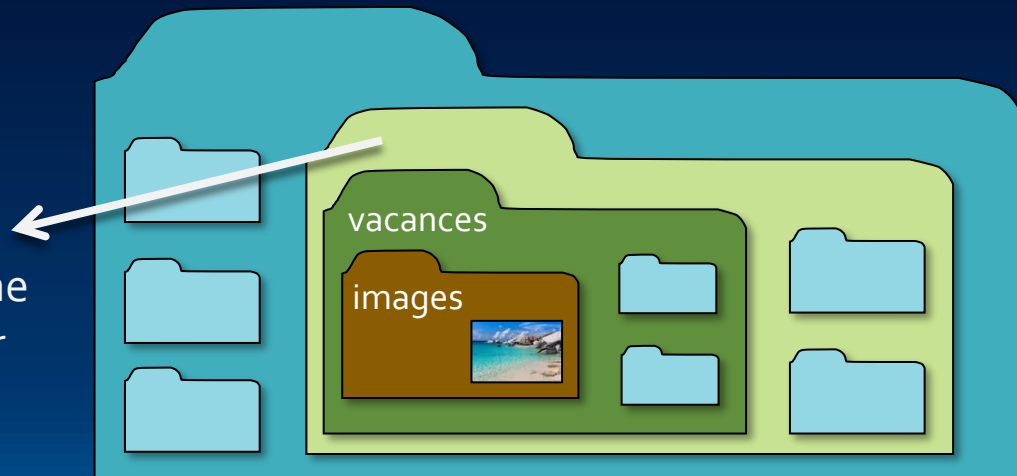
`/vacances/images/plage.jpg`

adresse absolue  
de la ressource

Sur le serveur :

répertoire racine  
du site `www.monsite.be`

- pas forcément la racine de la machine-serveur
- noté `"/`



# URL raccourcie

- À l'intérieur du site (par exemple dans l'attribut d'une balise `<img/>`), on peut **écourter l'adresse absolue** :
  - On ne précise pas (plus) le protocole ni le domaine.
  - Syntaxe : l'adresse commence par `/` (= répertoire racine)

~~`http://www.monsite.be/`~~`vacances/images/plage.jpg`



Dans ce document HTML, on peut faire référence à l'image en utilisant

`/vacances/images/plage.jpg`

- Et si on déplace le document et le dossier images ?
- Et si on renomme le dossier vacances ?



# URL relative

- À l'intérieur du site, on peut aussi utiliser des **URL relatives**
  - On décrit l'adresse cible par rapport à l'emplacement de la source.
  - On décrit le chemin pour aller de la source à la cible.
  - Syntaxe : l'adresse **ne** commence **pas** par **/**.



Dans ce document HTML, on peut faire référence à l'image en utilisant

**images/plage.jpg**

ou (inutilement compliqué)

**../vacances/images/plage.jpg**  
(.. signifie remonter d'un niveau)

# Comment écrire une URL alors ?

- URL longue

`http://www.monsite.be/vacances/images/plage.jpg`

- On risque de devoir résoudre le nom de domaine à chaque fois !
- Le moins bon choix mais inévitable s'il s'agit d'un domaine différent.

- URL raccourcie

`/vacances/images/plage.jpg`

- Plus efficace
- Mais difficile à mettre à jour en cas de déplacement des fichiers !

- URL relative

`images/plage.jpg`

- À utiliser le plus souvent possible

# Exercice : URL relatives

Où se trouvent les images correspondant à ces URL ?

(adresses indiquées dans le document représenté)

- 1 plage1.jpg
- 2 images/plage2.jpg
- 3 images/2014/plage3.jpg
- 4 ../plage4.jpg
- 5 ../dessins/plage5.jpg
- 6 ../../divers/plage6.jpg
- 7 ./plage7.jpg
- 8 /plage8.jpg
- 9 /vacances/plage9.jpg

