

## R1.02 développement d'interfaces WEB - HTML

---

monnerat@u-pec.fr 

6 novembre 2024

IUT de Fontainebleau

Première partie I

## **Introduction**

Internet - Web

Le fonctionnement et protocoles

Protocoles

Topologie

Notion d'URL

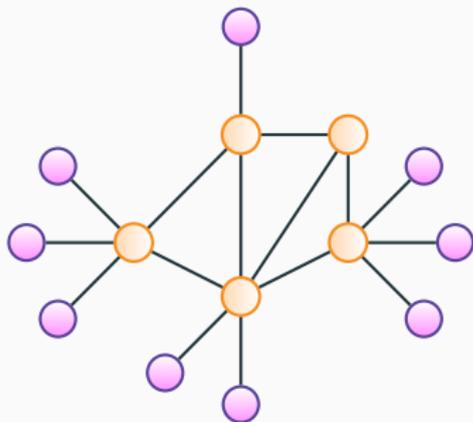
UTF-8

**Internet - Web**

- Réseau de réseaux
- Applications offrant des services + protocoles associés, basées sur TCP/IP.
- Mode Client/Serveur
- Comprend, entre autre :
  - Couche réseau (transport)
  - Gestion des noms et adresses
  - Outils et protocoles dédiés
  - Le(s) langage(s) HTML
- Beaucoup de technologies.
- Normalisation par le World Wide Web Consortium 

# Pourquoi faire ?

- Recherche d'informations !
- Communication entre les gens.
- Commerce électronique, vente aux enchères.
- Gestion de comptes en banques.
- Démarches administratives (impôts, etc...)
- Peer-to-peer
- Téléphonie, visiophonie, radio, vidéos, télévision, ...
- Enseignement, travail à distance.



- Permet d'accéder à des documents liés entre eux, sur des machines différentes.
- Architecture basée sur :
  - La localisation  $\rightsquigarrow$  URL
  - Le protocole  $\rightsquigarrow$  HTTP
  - Le langage  $\rightsquigarrow$  HTML
- Succès grâce :
  - Interfaces graphiques conviviales.
  - Grande diversité et quantité d'information.
  - Multi-platerformes.
  - Frameworks de développement mobile avec React Native, Flutter, Cordova, Xamarin, Ionic, etc.

Interconnexion d'applications de toutes sortes, sur toutes platerformes et tournant sur tout type de matériel.

# **Le fonctionnement et protocoles**

# Comment ça marche ?

1. Le client envoie une requête HTTP au serveur web.



# Comment ça marche ?

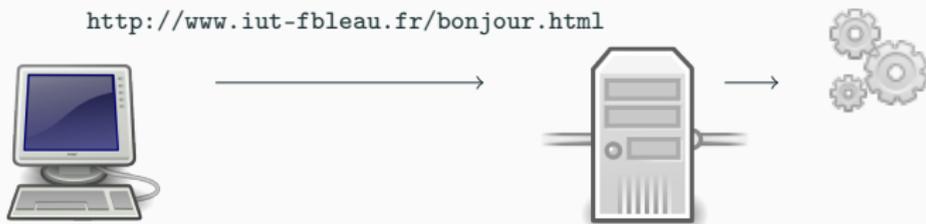
1. Le client envoie une requête HTTP au serveur web.
2. Le serveur vérifie la demande, les autorisations et renvoie la réponse.

`http://www.iut-fbleau.fr/bonjour.html`



# Comment ça marche ?

1. Le client envoie une requête HTTP au serveur web.
2. Le serveur vérifie la demande, les autorisations et renvoie la réponse.



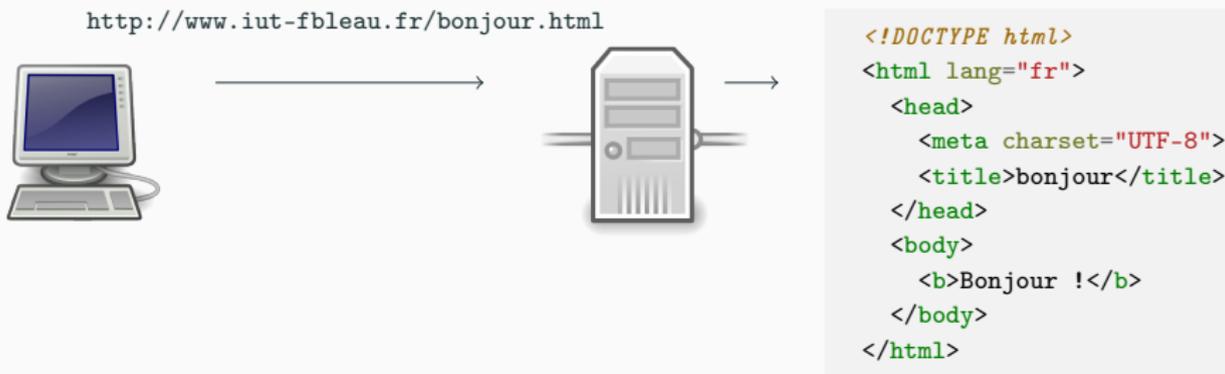
# Comment ça marche ?

1. Le client envoie une requête HTTP au serveur web.
2. Le serveur vérifie la demande, les autorisations et renvoie la réponse.



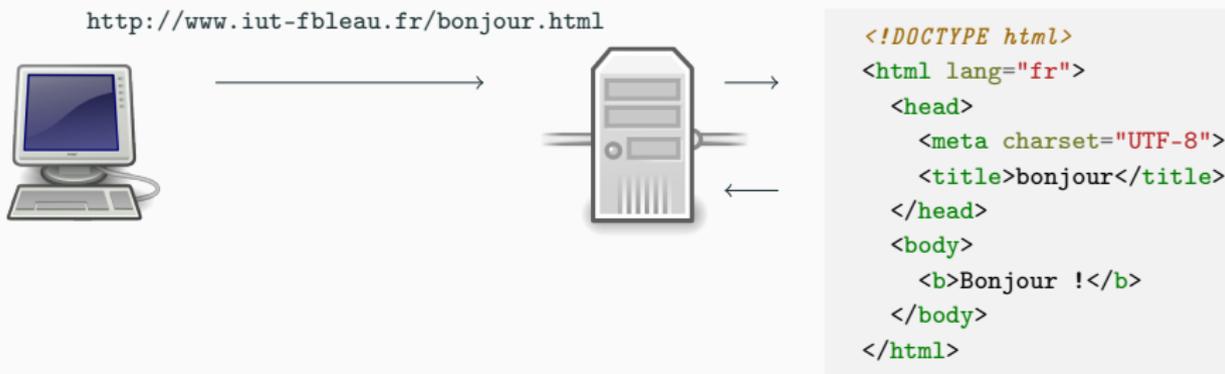
# Comment ça marche ?

1. Le client envoie une requête HTTP au serveur web.
2. Le serveur vérifie la demande, les autorisations et renvoie la réponse.



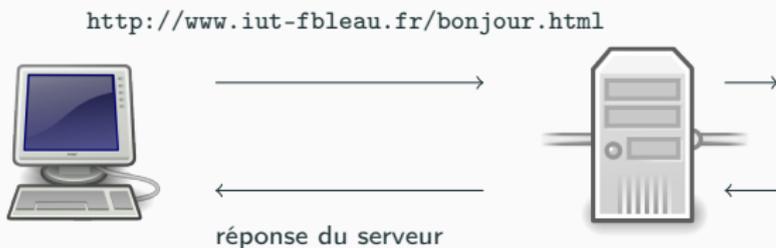
# Comment ça marche ?

1. Le client envoie une requête HTTP au serveur web.
2. Le serveur vérifie la demande, les autorisations et renvoie la réponse.



# Comment ça marche ?

1. Le client envoie une requête HTTP au serveur web.
2. Le serveur vérifie la demande, les autorisations et renvoie la réponse.

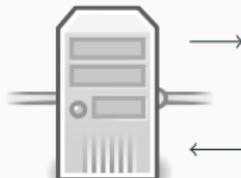


```
<!DOCTYPE html>
<html lang="fr">
  <head>
    <meta charset="UTF-8">
    <title>bonjour</title>
  </head>
  <body>
    <b>Bonjour !</b>
  </body>
</html>
```

# Comment ça marche ?

1. Le client envoie une requête HTTP au serveur web.
2. Le serveur vérifie la demande, les autorisations et renvoie la réponse.
3. Le navigateur interprète la réponse reçue et affiche le fichier html.

`http://www.iut-fbleau.fr/bonjour.html`

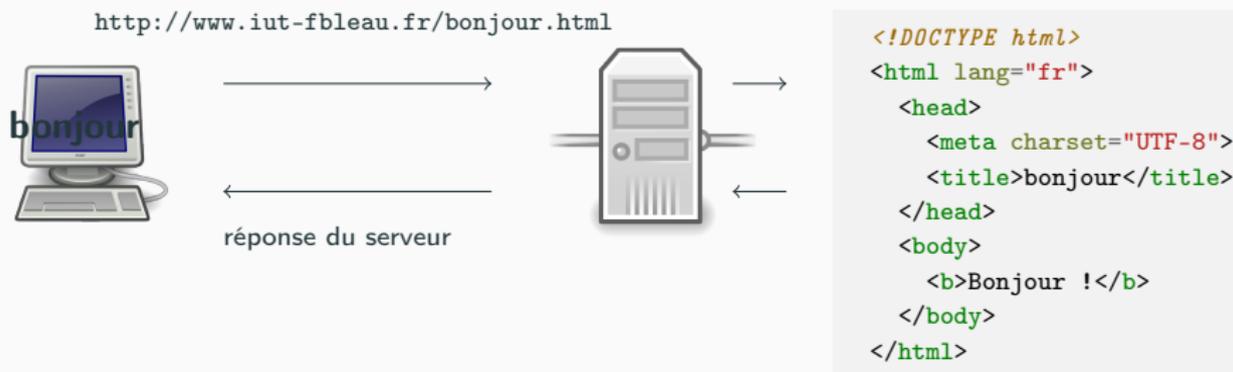


réponse du serveur

```
<!DOCTYPE html>
<html lang="fr">
  <head>
    <meta charset="UTF-8">
    <title>bonjour</title>
  </head>
  <body>
    <b>Bonjour !</b>
  </body>
</html>
```

# Comment ça marche ?

1. Le client envoie une requête HTTP au serveur web.
2. Le serveur vérifie la demande, les autorisations et renvoie la réponse.
3. Le navigateur interprète la réponse reçue et affiche le fichier html.

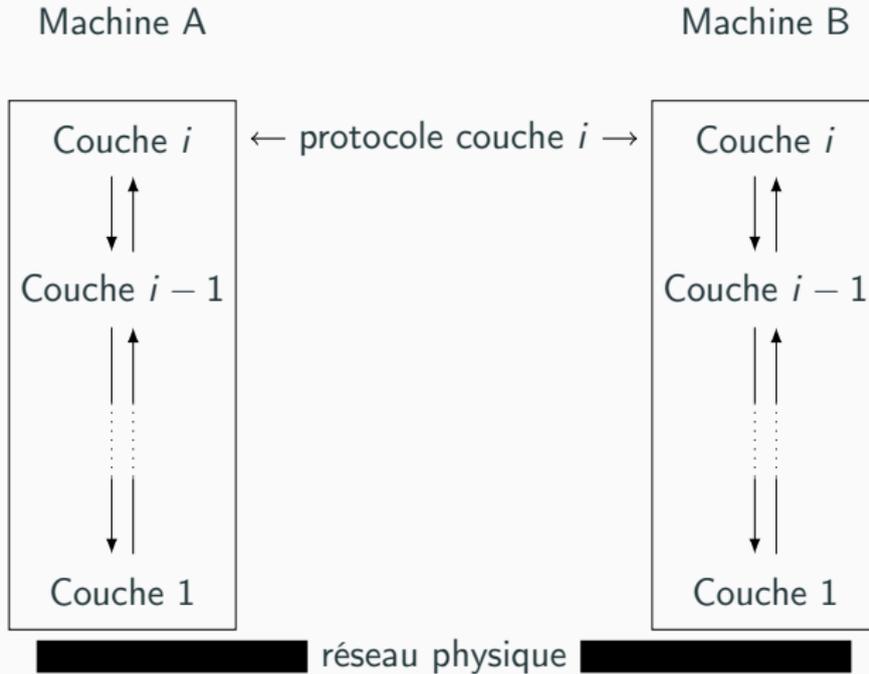


Il peut y avoir en plus :

- Des contrôles par compte, domaine, adresse ip,...
- Des exécutions de codes coté serveur et/ou client.

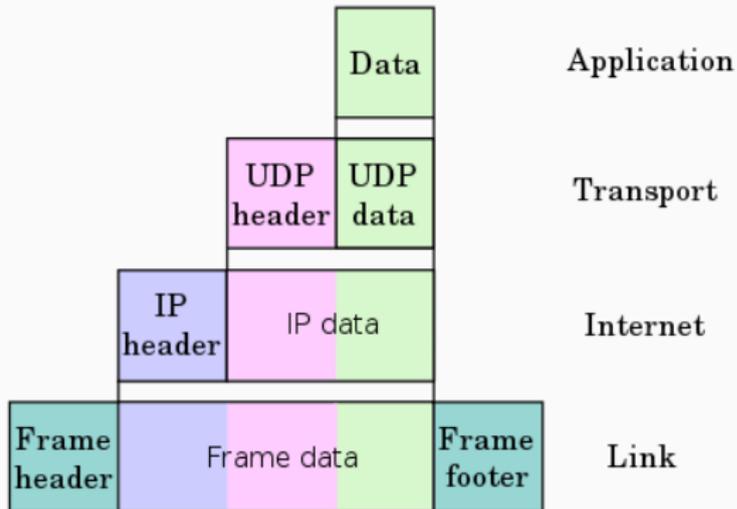
# Le fonctionnement et protocoles

## Protocoles

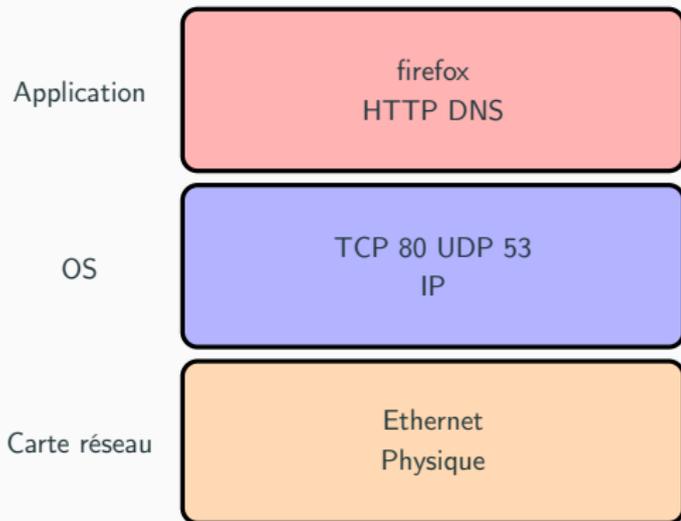


Application		
Présentation	ftp, telnet, <b>http</b> , smtp,...	Messages
Session		
Transport	TCP ou UDP	Segment TCP, Datagramme UDP
Réseau	Protocole de routage : IP	Datagrammes
Liaison	802.x, PPP, HDLC	Trames
Physique	Physique	Physique

# Encapsulation



# Requête HTTP



File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help

Filter:  Expression... Clear Apply Enregistrer

No.	Time	Source	Destination	Protocol	Length	Info
6	2.307594000	10.11.80.33	37.58.131.227	TCP	66	32985
7	2.307668000	10.11.80.33	37.58.131.227	HTTP	521	GET
8	2.314780000	37.58.131.227	10.11.80.33	TCP	66	http
9	2.315365000	37.58.131.227	10.11.80.33	TCP	1388	[TCP
10	2.315380000	10.11.80.33	37.58.131.227	TCP	66	32985

▶ Frame 7: 521 bytes on wire (4168 bits), 521 bytes captured (4168 bits) on interface 0

- ▶ Ethernet II, Src: Sony\_67:a0:9c (00:1d:ba:67:a0:9c), Dst: Cisco\_93:08:00 (00:1d:e5:93:08:00)
- ▶ Internet Protocol Version 4, Src: 10.11.80.33 (10.11.80.33), Dst: 37.58.131.227 (37.58.131.227)
- ▶ Transmission Control Protocol, Src Port: 32985 (32985), Dst Port: http (80), Seq: 1, Ack: 1, Len: 455
- ▶ Hypertext Transfer Protocol
  - ▶ GET / HTTP/1.1\r\n
  - Host: www.iut-fbleau.fr\r\n
  - User-Agent: Mozilla/5.0 (X11; Linux x86\_64; rv:32.0) Gecko/20100101 Firefox/32.0\r\n
  - Accept: text/html,application/xhtml+xml,application/xml;q=0.9,\*/\*;q=0.8\r\n
  - Accept-Language: fr,en-us;q=0.7,en;q=0.3\r\n
  - Accept-Encoding: gzip, deflate\r\n
  - DNT: 1\r\n
  - Cookie: txp\_remember=0; storedtemplate=calm; style\_cookie=null; phpbb3\_h7zbj\_u=55; phpbb3\_h7zbj\_k=; phpbb3\_h7zbj\_k=
  - Connection: keep-alive\r\n
  - \r\n
  - [Full request URI: http://www.iut-fbleau.fr/]

```

0000  00 1d e5 93 08 00 00 1d ba 67 a0 9c 08 00 45 00  .....g....E.
0010  01 fb eb b4 40 00 40 06 c9 ff 0a 0b 50 21 25 3a  ..k.@. ...P!%:
0020  83 e3 80 d9 00 50 cf a3 44 3d b5 8c 99 c0 80 18  ....P. D=.....
0030  00 e5 05 37 00 00 01 01 08 0a 00 57 2f 90 61 92  ...7.... ..W/.a.
0040  32 9c 47 45 54 20 2f 20 48 54 54 50 2f 31 2e 31  2.GET / HTTP/1.1
0050  04 0a 48 6f 72 74 2a 20 77 77 77 2e 69 75 74 2d  Host: www.iut

```

Frame (frame), 521 bytes      Packets: 685 · Displayed: 685 (100.0%)      Profile: Default

# Exemple

```
[denis@portabledenis ~]$ ncat -C dwarves.iut-fbleau.fr 80
GET /hello.html HTTP/1.1
Host: dwarves.iut-fbleau.fr
```

```
HTTP/1.1 200 OK
Date: Wed, 06 Nov 2024 10:07:09 GMT
Server: Apache/2.4.54 (Unix) OpenSSL/1.1.1q PHP/8.1.7
Last-Modified: Wed, 06 Nov 2024 09:54:47 GMT
ETag: "93-6263b82151716"
Accept-Ranges: bytes
Content-Length: 147
Content-Type: text/html
```

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <title>Hello world !</title>
  </head>
  <body>
    Helle world !
  </body>
</html>
```

# Le fonctionnement et protocoles

## Topologie

Permet la résolution des noms en adresse ip. Par exemple, pour la machine `www.iut-fbleau.fr`, le navigateur interroge un serveur DNS du client (free, etc) qui découpe l'adresse autour des points :

- `fr` : le (cc)TLD (Top Level Domain). Il faut trouver un dns responsable de `.fr`, à savoir l'afnic.
- `iut-fbleau` : un serveur dns de l'afnic donnera l'adresse d'un dns responsable du sous domaine `iut-fbleau`.
- `www` : celui-ci lui donnera l'adresse ip de `www`.

`www.iut-fbleau.fr` est résolue en **37.58.131.227**

# Le fonctionnement et protocoles

## Notion d'URL

# Adressage des documents

## Solution

- **URL** : *Uniform Resource Locator* = adressage universelle de ressources.
- 3 parties : le protocole (comment), le nom (où) et le nom du document (quoi).
- **URL**  $\subset$  **URI** *Universal Resource Identifier*.

## Exemples

- `http://www.iut-fbleau.fr/sitebp`
- `https://www.google.fr:8888/img/plan.php?x=12&y=20`
- `ftp://user:password@www.iut-fbleau.fr/account/`

# Composition d'une URL

`protocole://hostname:port/path?arguments#anchor` 3 parties :

- `protocole` (comment ?)
- `adresse` (où ?)
- `document` (quoi ?)

La racine / de path est définie par la configuration du serveur Web. (rien à voir a priori avec la `racine du système de fichier`)

l'url peut contenir des arguments, et un point d'ancrage (à la fin).

`http://www.youtube.com:80/results?search_query=cat#search-results`

**UTF-8**

# Représentation du texte

Avant de structurer un document, on s'intéresse au texte.

Comment le représenter (l'encoder) dans un fichier ?

أَرْبَعٌ تَحْتِ، Доброе утро, Comment ça va ?, ¡Hola !

Historiquement, encodage 1 caractère = 1 octet (8 bits) :

- ASCII sur 7 bits (128 caractères),
- ASCII étendu 8 bits (256 caractères, dont 128 de « symboles »,
- Latin 1 : ASCII 7 bits + 128 caractères « ouest-européens » (lettres accentuées française, italienne, etc.)
- Latin 2 : ASCII 7 bits + 128 caractères « est-européens » (Serbe, Hongrois, Croate, Tchèque, etc. )
- etc.

Problèmes : impossibilité de mélanger les alphabets. Chaque logiciel interprétait les octets de manière prédéfinie.

## Universal (Character Set) Transformation Format 8 bit

- Un organisme (ISO) donne un code à chaque symbole. C'est le standard Unicode (la version actuelle, 12.1 référence 137 994 symboles).
- Encodage à taille variable « universel » (contient tous les alphabets connus)
- Compatible avec ASCII 7 bits

Nombre d'octets	Octet 1	Octet 2	Octet 3	Octet 4
1	0xxxxxxx			
2	110xxxxx	10xxxxxx		
3	1110xxxx	10xxxxxx	10xxxxxx	
4	11110xxx	10xxxxxx	10xxxxxx	10xxxxxx

# Exemples

A	→	41 <sub>16</sub>	→	01000001 <sub>2</sub>	≡	41
ẽ	→	1ec5 <sub>16</sub>	→	0001111011000101 <sub>2</sub>	≡	E1 BB 85
🙄	→	1f435 <sub>16</sub>	→	00011111010000110101 <sub>2</sub>	≡	F0 9F 90 B5

Avantages :

- compatible ASCII 7 bits (d'anciens documents texte en anglais sont toujours lisibles),
- pas d'espace gaspillé (à l'inverse d'UTF-32 ou tous les caractères font 32 bits).

Inconvénients :

- Caractères à taille variable : il faut parcourir le texte pour trouver le nième caractère,
- Les vieux logiciels doivent être adaptés.

## Deuxième partie II

# HTML

Historique

Balise/Tag

Imbrication

Structure générale

Principaux éléments

# Historique

# Un peu d'histoire

L'aventure a commencé au début des années 1990. Quelques acteurs et inventeurs :

- Tim Berners-Lee
- Le C.E.R.N.
- Mosaic, Netscape



Figure 1: Tim Berners-Lee



Figure 2: Le premier serveur Web

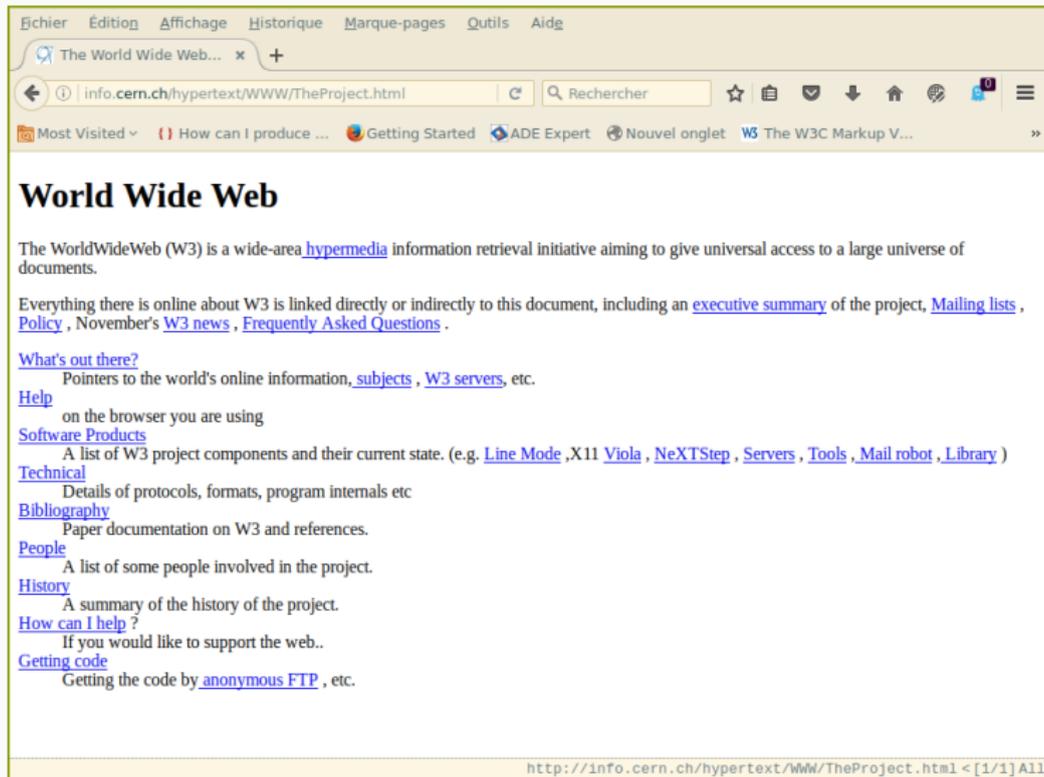


Figure 3: Le premier site internet

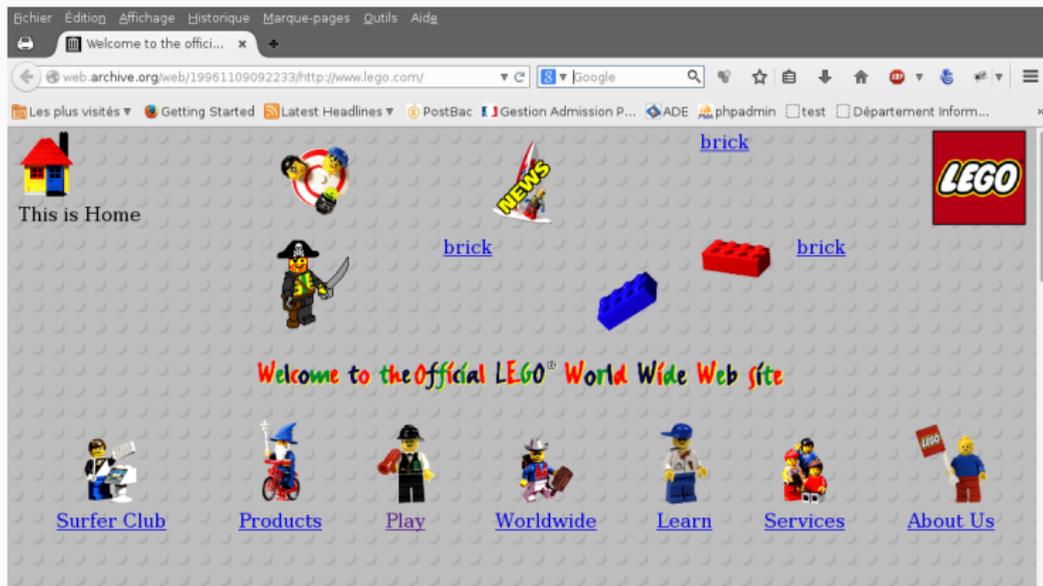
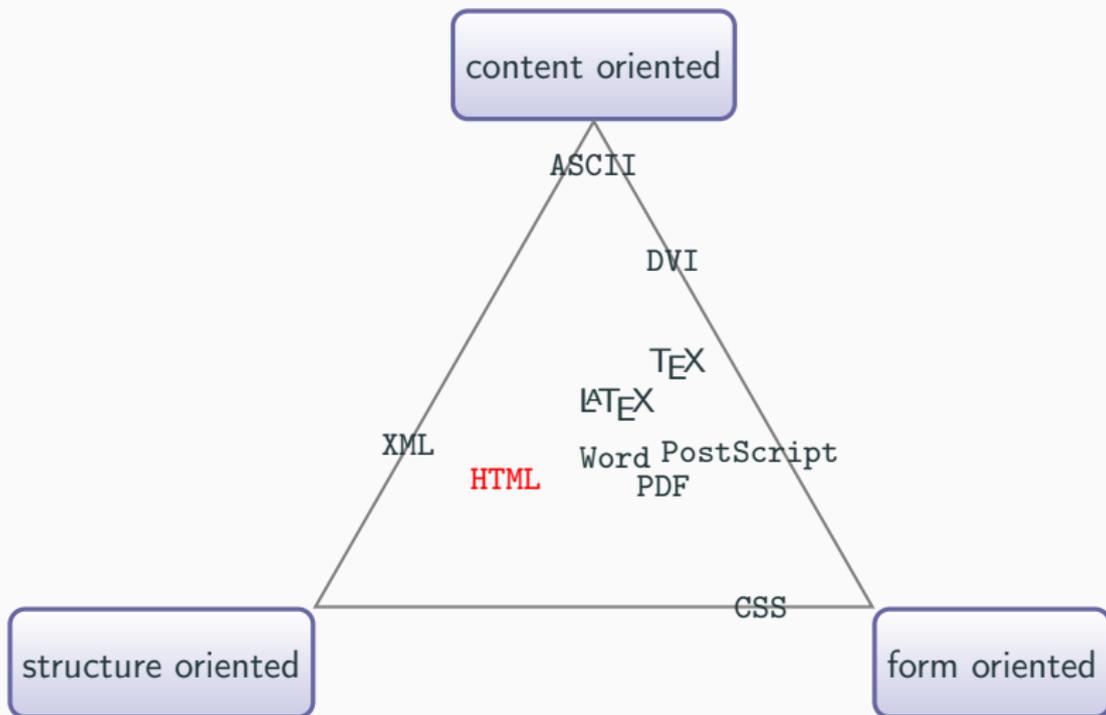


Figure 4: Le premier site (pré)historique de Lego en 1996

# HTML : langage de structuration de documents (?)



**Balise/Tag**

- HTML (Hyper Text Markup Language) est un langage à balises (tag).
- Décrit les pages WEB.
- Version 4.01 en 1999

<http://www.w3.org/TR/html401/> 

- Version 5

<http://www.w3.org/TR/html5/> 

- Intégration de la vidéo et du son, support pour le dessin.
- Ajout d'attributs, de contrôles et de balises structurantes.
- Adaptation aux périphériques.

## Balise : mot clé entre chevrons "<" et ">" (case insensitive)

- Généralement, une balise de début (ouvrante) et de fin (fermante).
- Certaines balises sont seules : <br> (elles n'ont pas de contenus)

## Attributs : données additionnelles d'une balise

- Une paire `nom="valeur"` (minuscule, souvent case sensitive)
- Placée dans la balise ouvrante.
- Il peut y en avoir plusieurs : `width`, `id`, `class`, `onclick`, ...



## Balises et attributs

```
<a href="http://www.iut-fbleau.fr"> iut </a>
```

- Balise ouvrante <a>

## Balises et attributs

```
<a href="http://www.iut-fbleau.fr"> iut </a>
```



- Balise ouvrante <a>
- Attribut href de la balise

## Balises et attributs

```
<a href="http://www.iut-fbleau.fr"> iut </a>
```

- Balise ouvrante <a>
- Attribut href de la balise
- Contenu texte de la balise

## Balises et attributs

```
<a href="http://www.iut-fbleau.fr"> iut </a>
```

- Balise ouvrante <a>
- Attribut href de la balise
- Contenu texte de la balise
- Balise fermante </a>

## Balises et attributs

```
<a href="http://www.iut-fbleau.fr"> iut </a>
```

- Balise ouvrante <a>
- Attribut href de la balise
- Contenu texte de la balise
- Balise fermante </a>

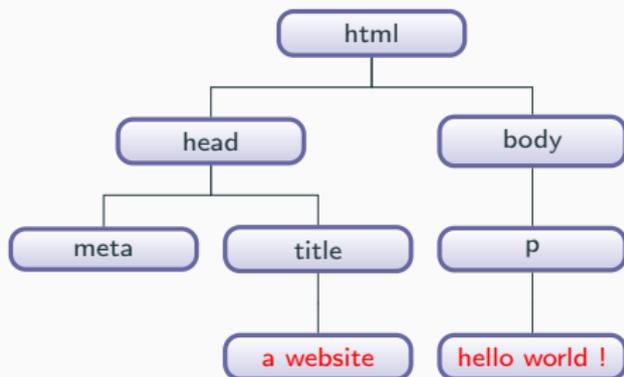
Certaines balises sont autofermantes (elles n'ont pas de contenu) :  
<br> , <embed> , <hr>, <img>, <input>, <link>, <meta>,  
<param>, <source>, <wbr>

# Imbrication

# Imbrication des balises

L'écriture d'un document html syntaxiquement correct obéit à des règles (grammaire) précises :

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <title>a website</title>
  </head>
  <body>
    <p>hello world !</p>
  </body>
</html>
```



Représentation arborescente

# Structure générale

# Structure minimale d'un document html

Structure du document html

```
<!DOCTYPE html>  
<html lang="en">  
  <head>  
    <meta charset="UTF-8">  
    <title>a website</title>  
  </head>  
  <body>  
    <p>hello world !</p>  
  </body>  
</html>
```

# Structure minimale d'un document html

## Structure du document html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <title>a website</title>
  </head>
  <body>
    <p>hello world !</p>
  </body>
</html>
```

Un préambule **DOCTYPE** qui indique la syntaxe utilisée dans le document.

# Structure minimale d'un document html

## Structure du document html

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>a website</title>
  </head>
  <body>
    <p>hello world !</p>
  </body>
</html>
```

La balise `html` est la racine du document. Elle contient :

- la balise `head`, qui contient des métadonnées.
- la balise `body`, qui contient le contenu.

# Structure minimale d'un document html

## Structure du document html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <title>a website</title>
  </head>
  <body>
    <p>hello world !</p>
  </body>
</html>
```

La balise **head**, qui contient ici :

- la balise **meta** qui permet de rajouter des mots clés, le type du contenu, une description. Utilisée par les navigateurs et robots (référencement).

- un titre avec la balise **title**

On peut rajouter :

- Des ressources utilisées par la page avec la balise **link**.
- Des références aux fichiers javascript avec la balise **script**.

# Structure minimale d'un document html

## Structure du document html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <title>a website</title>
  </head>
  <body>
    <p>hello world !</p>
  </body>
</html>
```

La balise **body** contient les balises affichées dans le navigateur.

### Validation de la syntaxe

<http://validator.w3.org> 

# Correction de la syntaxe

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <title>a website</title>
  </head>
  <body>
    <p>hello world !</p>
  </body>
</html>
```

## Validation

http://validator.w3.org



The screenshot shows the W3C Markup Validation Service interface. The browser address bar displays "http://validator.w3.org/check". The page title is "[Valid] Markup Validation of exe...". The main content area displays a green "Congratulations" heading followed by a message: "The uploaded document 'exemple.html' was successfully checked as HTML5. This means that the resource in question identified itself as 'HTML5' and that we successfully performed a formal validation of it. The parser implementations we used for this check are based on [validator.nu](#) (HTML5)." The browser's menu bar includes "Fichier", "Édition", "Affichage", "Historique", "Marque-pages", "Zone", "Outils", and "Aide". The browser's address bar also shows "validator.w3.org/check" and "w3c validator". The browser's toolbar includes "Les plus visités", "Getting Started", "Latest Headlines", "PostBac", "ADE", and "Espace etudiant".

## En-tête : bonnes pratiques

Il est très fortement conseillé de toujours préciser, en plus de la balise `<title>`.

- le codage des caractères,
- l'auteur,
- la description,
- les mots-clés.

```
<head>
  <meta charset="utf-8">
  <meta name="description" content="mon site">
  <meta name="author" content="DM">
  <meta name="keywords" content="html, but info">
  <title>Ma première page html</title>
</head>
```

## Principaux éléments

# Principaux éléments

Éléments de structuration de page

# Structuration de page

Les éléments, introduits avec html5 (section, article, nav, aside, header, footer) définissent des portions du document avec une sémantique particulière.

**header** Section d'introduction d'un article, d'une autre section ou du document entier (en-tête de page).

**nav** Section possédant des liens de navigation principaux (au sein du document ou vers d'autres pages)

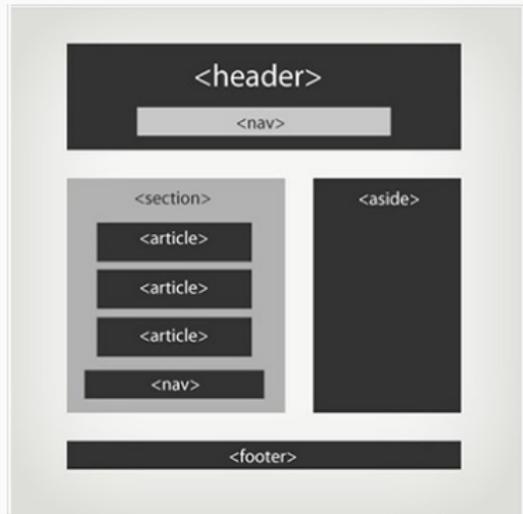
**main** Contenu principal de la page

**section** Section générique regroupant un même sujet, généralement avec un titre

**article** Section de contenu indépendante

**aside** Section dont le contenu est un complément par rapport à ce qui l'entoure, qui n'est pas forcément en lien direct avec le contenu mais qui peut apporter des informations supplémentaires.

**footer** Section de conclusion d'une section ou d'un article, voire du document entier (pied de page).



# Principaux éléments

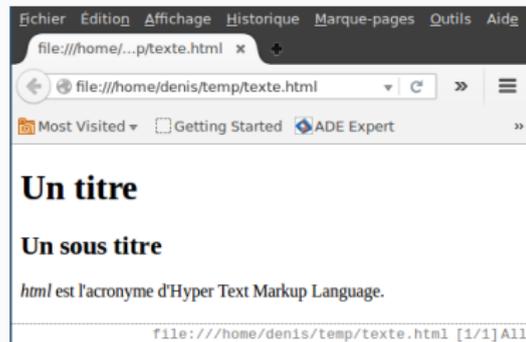
Textes - sémantique

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Commentaires</title>
</head>
<body>
  <!--
  Ceci est un commentaire
  sur plusieurs lignes !
  -->
</body>
</html>
```

Il s'agit de balises qui ont un contenu texte avec une **sémantique précise** :

- `<p>paragraphe</p>`
- `<h1> ... <h6>` : titre
- `<em>emphase</em>`
- `<strong>renforcement</strong>`
- `<pre>affiche le contenu tel quel</pre>`
- `<blockquote>mise en valeur d'un paragraphe</blockquote>`
- `<abbr></abbr>`
- `<address></address>`
- etc.

```
<!DOCTYPE html>
<html lang="fr">
  <head>
    <meta charset="UTF-8">
    <title>Texte</title>
  </head>
  <body>
    <h1>Un titre</h1>
    <h2>Un sous titre</h2>
    <p>
      <em>html</em> est l'acronyme
      d'Hyper Text Markup Language.
    </p>
  </body>
</html>
```



# Principaux éléments

Textes - mise en forme

- `<font face="Comic Sans MS" color="#0000dd" size="4">texte</font>`
- `<br>` passage à la ligne
- `<i>italique</i>`
- `<b>gras</b>`
- `<u>souligné</u>`
- `<strike>texte barré</strike>`
- `<u>souligné</u>`
- `<big>`, `<small>`
- `<sub>`, `<sup>`
- etc

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Mise en forme</title>
</head>
<body>
  <p>Du <b>gras</b>, de <i>l'italique</i>,<br>
  du texte <del>barré</del>, du texte
  <u>souligné</u>, en <sub>bas</sub>, en <sup>haut</sup>,
  etc.
  </p>
</body>
</html>
```

Fichier Édition Affichage Historique Marque-pages Outils Aide

file:///home...p/texte.html x file:///home/.../textef.html x +

file:///home/denis/Enseignements/Cours/m

Most Visited Getting Started ADE Expert

Du **gras**, de *l'italique*,  
du texte ~~barré~~, du texte souligné, en <sub>bas</sub>, en <sup>haut</sup>, etc.

file:///home/denis/Ensei...ew/listings/textef.html [2/2] All

# Principaux éléments

Listes

- Liste ordonnée <ol>, liste à puces <ul>
- Élément de chaque liste <li>

```
<ul>
  <li>item 1</li>
  <li>item 2</li>
</ul>
```

- Liste de définition <dl> (définition list)
- ajouter un terme <dt> et sa définition <dd>

```
<dl>
  <dt>html</dt>
  <dd>langage utilisé pour affiché du contenu sur le web</dd>
  <dt>balise</dt>
  <dd>élément présentant certaines fonctionnalités</dd>
  <dt>attribut</dt>
  <dd>élément spécifique à une balise pouvant prendre certaines valeurs</dd>
</dl>
```

Fichier Édition Affichage Historique Marque-pages Outils Aide

file:///home/...p/liste.html x +

file:///home/denis/temp/liste.html

Most Visited Getting Started ADE Expert Tsunamixte 2015

## Liste non ordonnée

- item1
- item2
- item3

## Liste de définitions

html  
langage utilisé pour affiché du contenu sur le web

balise  
élément présentant certaines fonctionnalités

attribut  
élément spécifique à une balise pouvant prendre certaines valeurs

---

file:///home/denis/temp/liste.html [1/1] All

# Principaux éléments

Liens

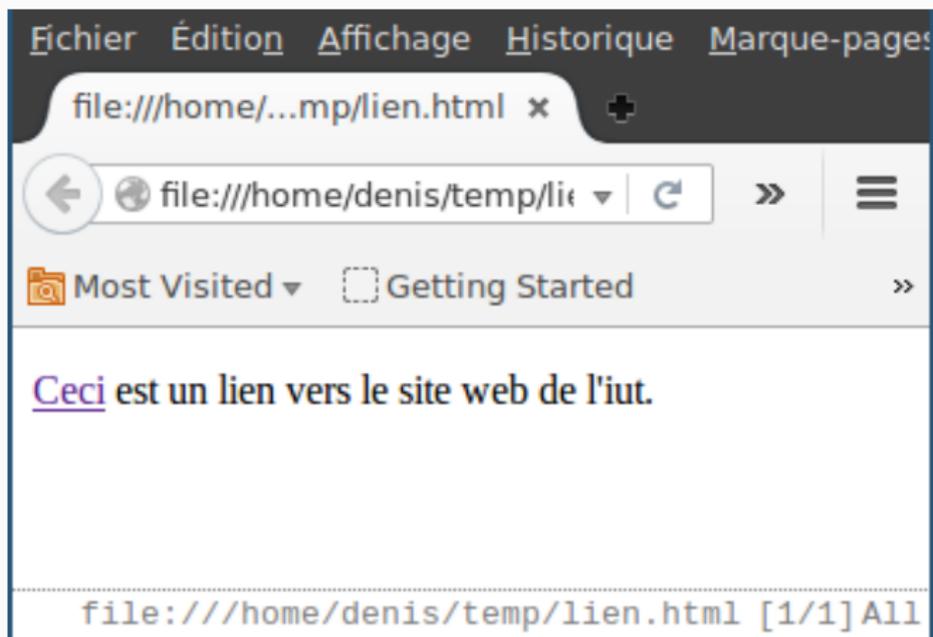
```
<a href="url page" target="ou le lien est ouvert">contenu</a>
```

target représente où le lien doit être ouvert :

- s'il manque, la même page
- `_blank` un nouvel onglet ou nouvelle page
- autres valeurs : `_self`, `_parent`, `_top`

url est l'url du lien. il peut être :

- absolu : `http://www.iut-fbleau.fr`
- relatif par rapport
  - à l'adresse de base si document contient un élément base,
  - à l'adresse de la page en cours sinon.
- Il peut contenir en plus un endroit précis dans la page (attribut `id` d'une élément).



```
<h1>Sports</h1>
<p>
  <a href="../sport.html#foot">Le football </a>
  <a href="#arc">Le tir à l'arc</a>
  <a href="http://www.ff-handball.org/">handball</a>
</p>
```

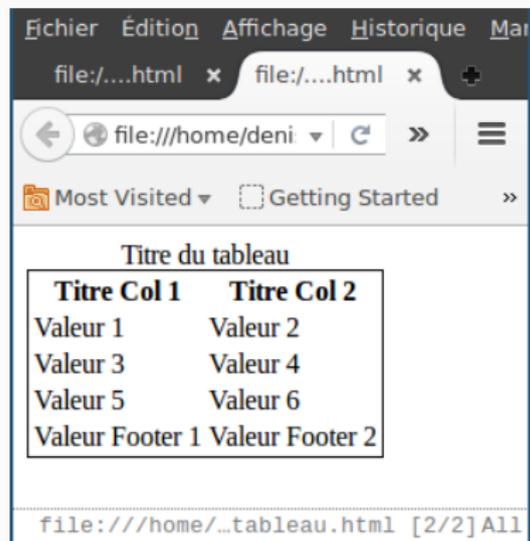
Pouvez-vous expliquer les différences ?

# Principaux éléments

## Tableaux

# Tableaux

```
<table style="border:solid 1px black">
  <caption>Titre du tableau</caption>
  <thead>
    <tr>
      <th>Titre Col 1</th>
      <th>Titre Col 2</th>
    </tr>
  </thead>
  <tfoot>
    <tr>
      <td>Valeur Footer 1</td>
      <td>Valeur Footer 2</td>
    </tr>
  </tfoot>
  <tbody>
    <tr>
      <td>Valeur 1</td>
      <td>Valeur 2</td>
    </tr>
  </tbody>
</table>
```



# Principaux éléments

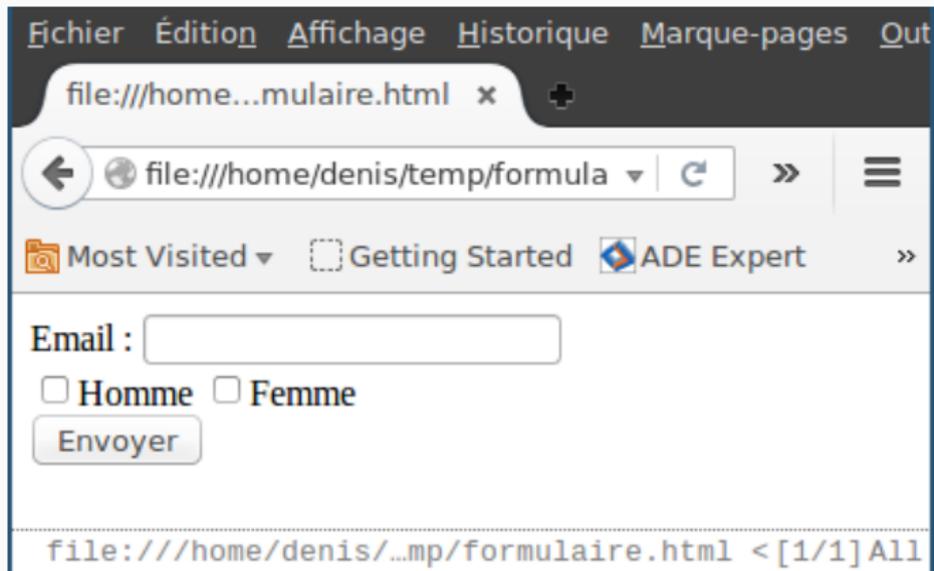
## Formulaires

Un formulaire permet la saisie et l'envoi de données au serveur :

```
<form action="url" method="post ou get">
  <label for="mail">Mail</label>
  <input type="text" id="mail" required>
  <input type="submit" name="soumission" value="envoyer">
</form>
```

- `url` : définit l'url du script à qui sont envoyées les données sur le serveur.
- `method` : les données sont envoyées soit en `get` (dans l'url), soit en `post` (dans le corps de la requête http).

Un formulaire contient un ou plusieurs éléments de type `input`, `select`, `textarea`, ... qui permettent à l'utilisateur de saisir des données.



- Il est conseillé d'ajouter à chaque champ un `label` (étiquette décrivant le champ)
- L'attribut `for` du `label` doit se référer à l'attribut `id` du champ correspondant
- L'attribut `placeholder` contient le texte affiché par défaut (et peut, à l'inverse, contenir tout type de caractères)
- La balise `<fieldset>` permet un rendu agréable par défaut

Nombreux types de champs possibles :

- Cases à cocher (checkbox), boutons radio (radio), zones de texte (textarea), sélections (select), barres de boutons (toolbar), liens à téléverser (file), etc.
- Variantes de text : tel , email , url , date , color , etc.

**Attention**, certains nouveaux types de champ (tel, date, color, etc.) ne s'affichent pas toujours bien sur tous les navigateurs.

On peut rajouter des contraintes de validations avec l'attribut `required` et `pattern`.

# Principaux éléments

## Multimedia

# Multimedia

Image : balise <img>

```

```

Video : balise <video>.

```
<video controls="controls" poster="exemples/exemplechatbg.jpg">  
  <source src="exemples/exemplechat.webm" type="video/webm">  
  <source src="exemples/exemplechat.mp4" type="video/mp4">  
  <source src="exemples/exemplechat.ogv" type="video/ogg">  
  Votre navigateur ne supporte pas le tag &lt;video&gt;.  
</video>
```

Son : balise <audio>

Question : que représente &lt; et &gt;, et pourquoi l'utiliser ?

# Principaux éléments

Éléments interactifs

- `<menu>` : menu qui apparait après un clic sur un bouton (encore expérimental).
- `<dialog>` : fenêtre de dialogue, éventuellement modale.
- `<details>` : révèle ou masque une information, avec un résumé en utilisant l'élément `<summary>`

- `<div>` : ne représente rien de particulier. Cette balise (une division) permet néanmoins de regrouper du contenu pour lui donner un sens particulier. A utiliser avec l'attribut `class` par exemple.
- `<span>` : comme la balise `div`, mais avec un affichage en mode ligne (et non bloc).

# Principaux éléments

Index des balises html(5)

## Structures

```
<!--...-->  
<!DOCTYPE>  
<html>  
<head>  
<title>  
<meta />  
<body>
```

## Sections

```
<div>  
<span>  
<section>  
<header>  
<hgroup>  
<nav>  
<article>  
<details>  
<summary>  
<figure>
```

...

```
<figcaption>  
<aside>  
<footer>
```

## Références

```
<base />  
<link />  
<style>  
<script>  
<noscript>
```

## Cadres

```
<frameset>  
<frame />  
<noframes>  
<iframe>
```

## Listes

```
<dir>  
<ol>  
<ul>  
<li>  
<dd>  
<dl>  
<dt>
```

## Liens

```
<a>  
<map>  
<area>
```

## Multimédia

```
<img />  
<video>  
<track>  
<audio>
```

...

```
<source>  
<embed>  
<applet>  
<object>  
<param />  
<canvas>  
<svg>
```

## Tableaux

```
<table>  
<caption>  
<colgroup>  
<col />  
<thead>  
<tbody>  
<tfoot>  
<tr>  
<th>  
<td>
```

## Formulaires

```
<form>
<fieldset>
<legend>
<label>
<button>
<input />
<textarea>
<select>
<optgroup>
<option>
<isindex>
<menu> 
<command> 
<datalist> 
<output> 
<keygen> 
```

## Rendus visuels

```
<center>
<hr />
<br />
<wbr /> 
<meter> 
<progress> 
```

## Textes - mise en forme

```
<b>
<basefont />
<bdi> 
<bdo>
<big>
<font>
<i>
<mark> 
<strike>
<sub>
<tt>
```

...

```
<u>
<s>
<small>
```

## Texte - sémantique

```
<abbr>
<acronym>
<address>
<blockquote>
<cite>
<code>
<del>
<dfn>
<em>
<h1> ...<h6>
<ins>
<kbd>
<p>
<pre>
<q>
```

...

```
<rp> 
<rt>  <ruby> 
<samp>
<strong>
<time> 
<var>
```