Développement web

DUT Info 2, 2019-2020

Thomas FRESSIN

thomas.fressin@u-pem.fr

PLAN DE LA PRÉSENTATION

- Dév. front-end/back-end
- Modèle d'exécution du web
- HTML
- CSS
- PHP
- MySQL

BIBLIOGRAPHIE

- Cours de Pierre-Nicolas Clauss (LORIA)
- Cours de Romain Lebreton, Sébastien Gagné, Auréline Quatrehomme
- Wikipédia
- http://www.commentcamarche.net/contents/html/
- http://www.apprendre-php.com
- http ://www.php.net/manual/fr

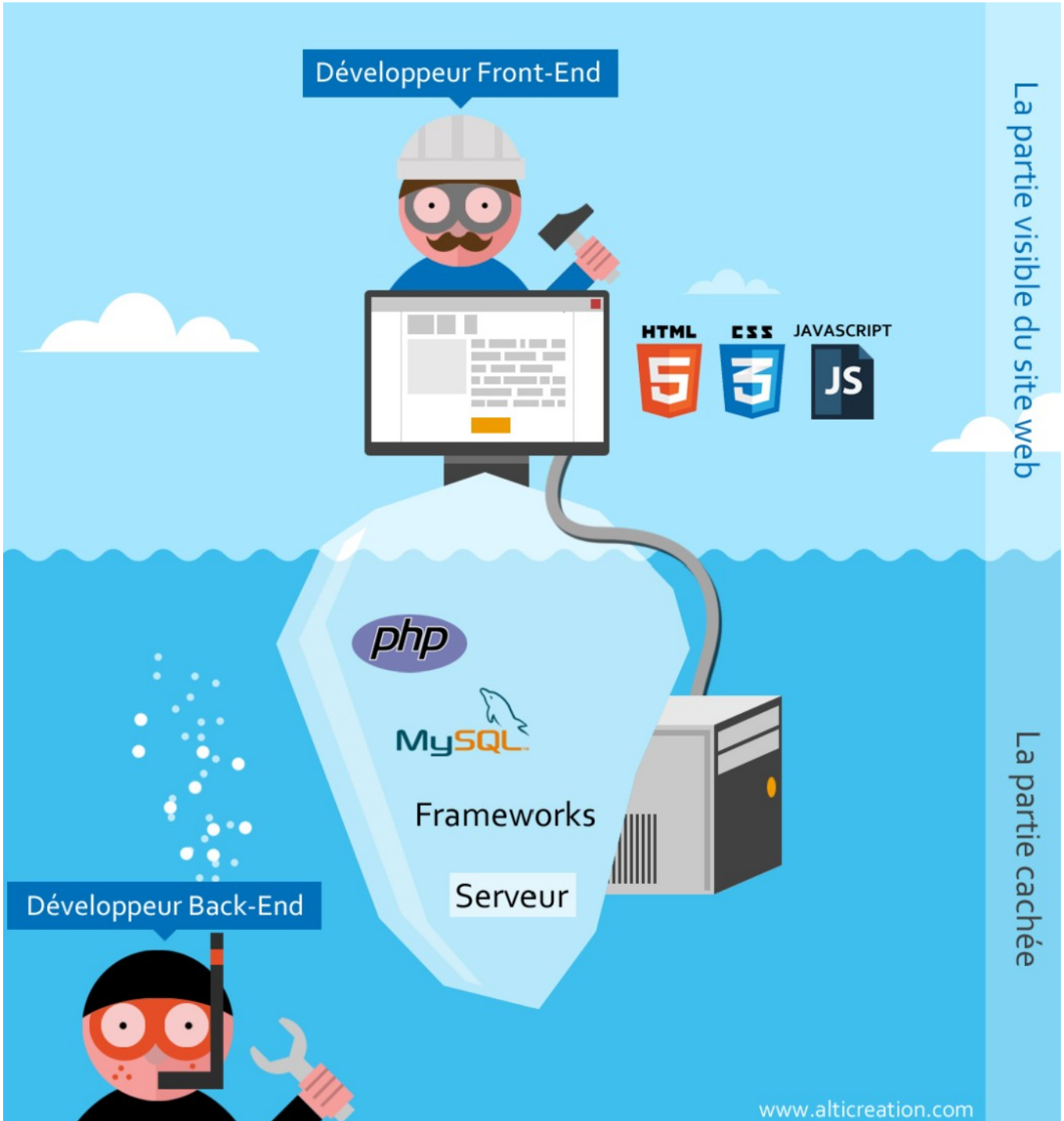
LE DEVELOPPEMENT FRONT-END / BACK-END

Frontend


HTML
CSS
CMS
JS
FLASH


HTML
CSS
Javascript
jQuery
Bootstrap
PHP
MySQL
WordPress

Développement Front-End


Développeur Front-End
HTML
CSS
JAVASCRIPT
La partie visible du site web
php
MySQL
Frameworks
Serveur
La partie cachée
Développeur Back-End
www.alticreation.com

FRONT-END

FULL-STACK

BACK-END

FRONT-END

Le « développement web frontal » (front-end en anglais) correspond aux ***productions HTML, CSS et JavaScript d'une page internet*** *ou d'une application qu'un utilisateur peut voir et avec lesquelles il peut interagir directement.*

Le principal défi du développement web frontal est de toujours ***s'adapter aux dernières évolutions*** *; les outils et les techniques de développement étant en évolution constante.*

La conception des sites internet doit également être capable d'offrir une ***bonne ergonomie de lecture*** *en facilitant la navigation et* ***l'obtention d'information****. Cet objectif est d'autant plus compliqué que les lecteurs utilisent maintenant différentes plateformes de format et de taille variés. Le développeur doit donc s'assurer que le site internet* ***apparaît correctement*** *sur l'ensemble des navigateurs Web et des plateformes/appareils disponibles.*

Hypertext : *une page doit être relier à d'autres pages*
→ comprendre l'architecture de l'information et les relations des pages

Markup : *le contenu est intégré à une structure de page balisée*

Règles *pour traduire le DOM dans une forme visuelle*

Règles de style en cascade*. Ce sont un ensemble de règles qui décrivent la priorité avec laquelle les styles sont rendus sur une page*

Contrôler les données *saisies dans des formulaires HTML*

Interagir avec le document HTML *via le DOM*

BACK-END

En informatique, un back-end (parfois aussi appelé un arrière-plan) est un terme désignant un étage de sortie d'un logiciel devant produire un résultat. On l'oppose au front-end (aussi appelé un frontal) qui lui est la partie visible de l'iceberg.

Illustration du concept :
Dans un magasin, on trouve une ***arrière-boutique*** *où sont stockés les articles, et un bureau qui assure le* ***bon fonctionnement du magasin****. Il s'agit du back-end, de tout ce qui se passe en arrière-plan sans que* ***le client ne s'en rende compte****.*
Dans ce même magasin, on retrouve un service à la clientèle et des étalages. Il s'agit du front-end, de ce que le client voit.

MODÈLE D'EXÉCUTION DU WEB

MODÈLE D'EXÉCUTION DU WEB

CLIENT

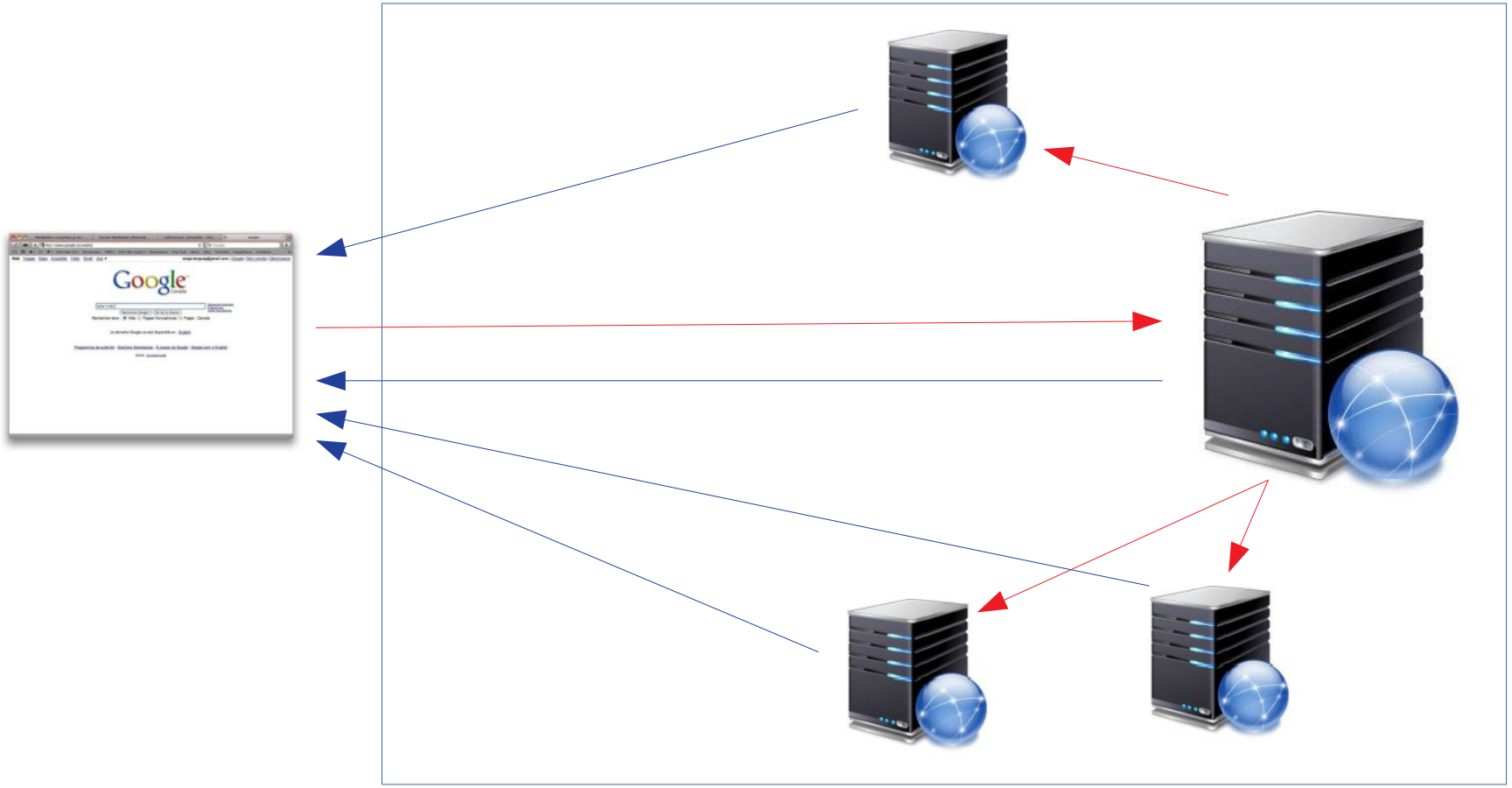
INTERNET

Requête HTTP
(protocole de transfert
hypertexte)

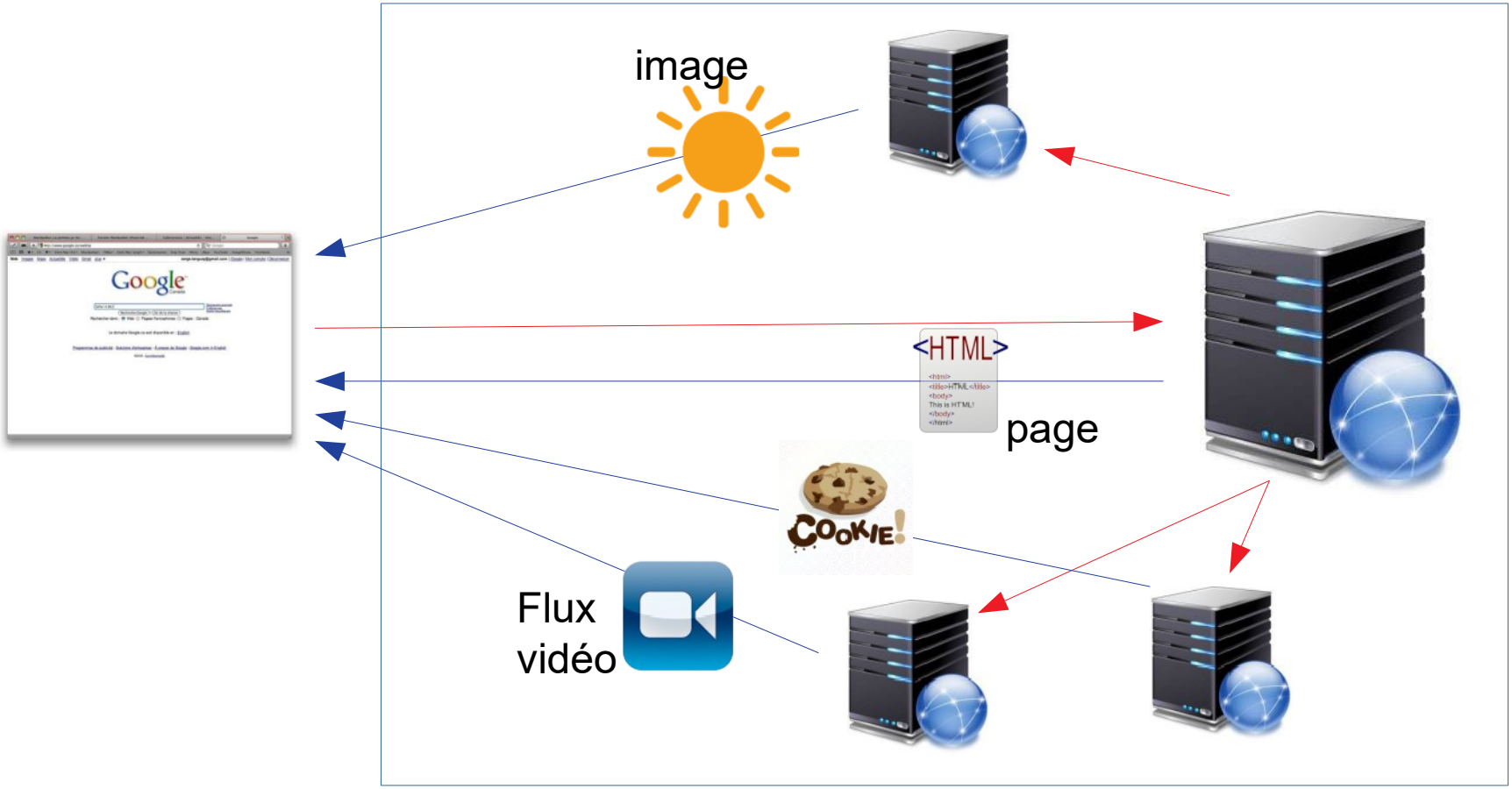
Code « web »

SERVEUR

MODÈLE D'EXÉCUTION DU WEB



MODÈLE D'EXÉCUTION DU WEB



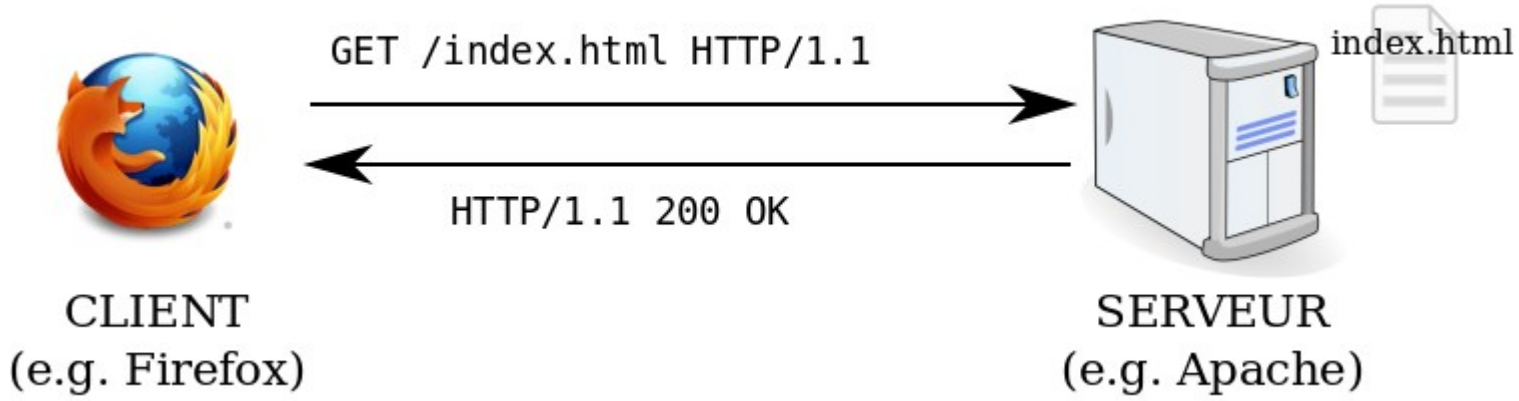
REQUÊTE HTTP

```
GET /~rletud/index.html HTTP/1.1
Host: infolimon.iutmontp.univ-montp2.fr
```

La réponse est alors:

```
HTTP/1.1 200 OK
Date: Tue, 08 Sep 2015 13:32:19 GMT
Server: Apache/2.2.14 (Ubuntu)
Last-Modified: Tue, 08 Sep 2015 13:06:07 GMT
Accept-Ranges: bytes
Content-Length: 5781
Content-Type: text/html

<html><head>... (contenu de index.html)
```

REQUÊTE HTTP



MODÈLE D'EXÉCUTION DU WEB

Ressources :

- Code html (page)
- Code css (style)
- Code js (interaction)
- Image
- Flux vidéo
- Cookie
- Redirection
- ...

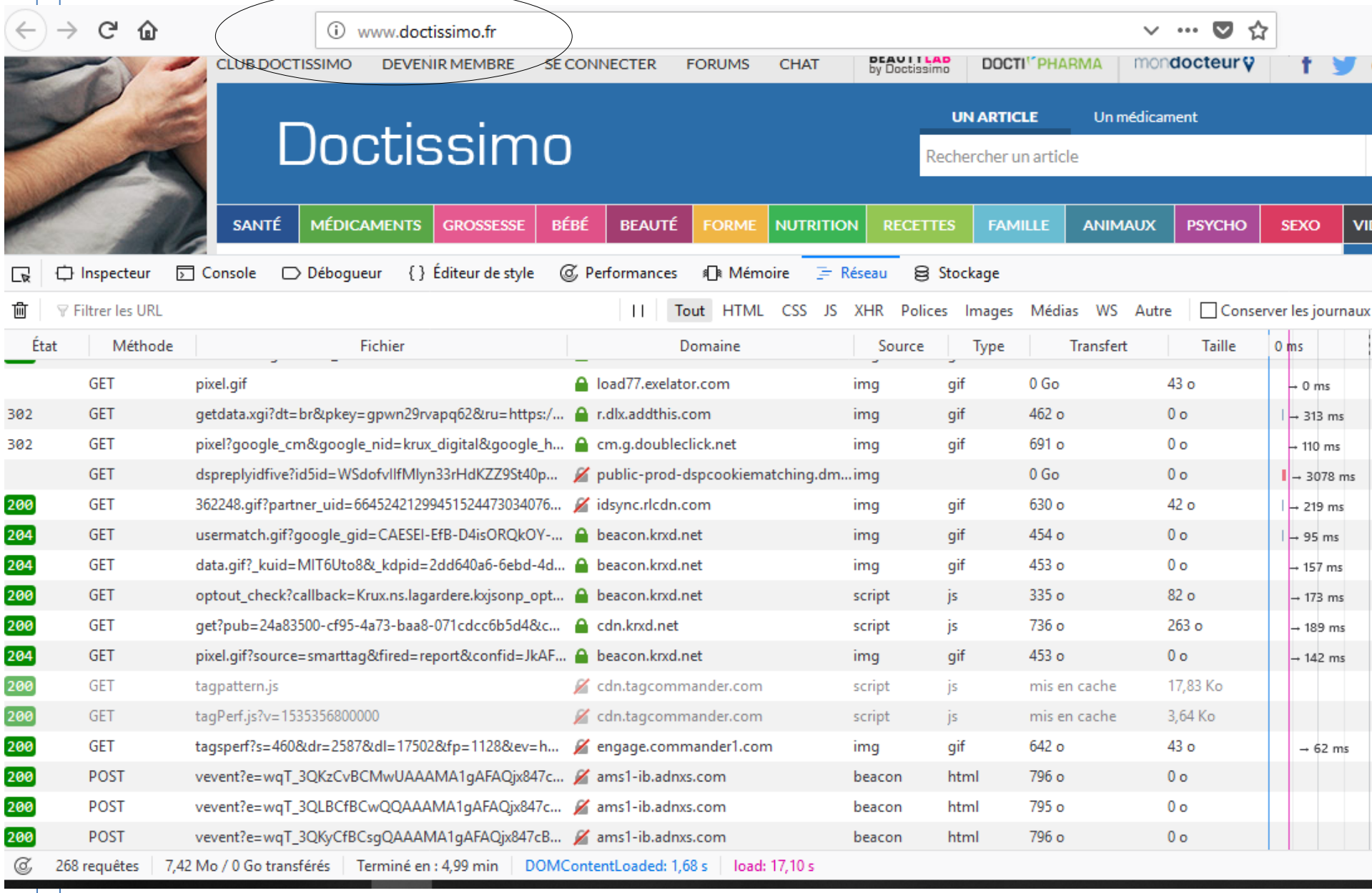
Principaux codes HTTP :

- 200 : succès
- 301 et 302 : redirection perm./temp.
- 401 : utilisateur non authentifié
- 403 : accès refusé
- 404 : page non trouvée
- 500 et 503 : erreur serveur
- 504 : le serveur n'a pas répondu


```
<!DOCTYPE html>
<html class="doc-rubric-home" lang="fr" dir="ltr">
<head>
    <!-- Google Tag Manager -->
    <script>
        (function(w,d,s,l,i){w[l]=w[l]||[];w[l].push(
                {'gtm.start': new Date().getTime(),event:'gtm.js'}
        );var f=d.getElementsByTagName(s)[0],
                j=d.createElement(s),dl=l!='dataLayer'?'&l='+l:'';j.async=true;j.src=
                'https://www.googletagmanager.com/gtm.js?id='+i+dl;f.parentNode.insertBefore(j,f);
        })(window,document,'script','dataLayer','GTM-W6ZWD3F');
    </script>
    <!-- End Google Tag Manager -->

    <title>Santé et bien être avec Doctissimo</title>
    <meta charset="utf-8"/>
    <meta name="viewport"                       content="width=device-width, initial-scale=1, minimum-scale=1.0, maximur
    <meta name="description"                    content="Toute l'information santé et bien être avec Doctissimo. Découvr
    <meta name="keywords"                       content="actualité, santé, magazine, féminin, médical, médicaments, mala

    <meta property="og:title"                   content="Santé et bien être avec Doctissimo" />
    <meta property="og:description"             content="" />
    <meta property="og:type"                    content="article" />
    <meta property="og:url"                     content="http://www.doctissimo.fr/" />
    <meta property="og:image"                   content="http://cdn3-doctissimo.ladmedia.fr/bundles/doctissimofrwww/ther

    <meta name="twitter:title"                  content="Santé et bien être avec Doctissimo" />
    <meta name="twitter:description"            content="" />
    <meta name="twitter:card"                   content="summary" />
    <meta name="twitter:site"                   content="@Doctissimo" />
    <meta name="twitter:image"                  content="http://cdn3-doctissimo.ladmedia.fr/bundles/doctissimofrwww/ther

                                                    <meta name="verify-v1"                      content="J3d601
    <meta name="robots"                         content="index, follow" />
    <meta name="application-name"               content="Doctissimo" />
    <meta name="msapplication-tooltip"          content="Santé et bien être avec Doctissimo" />
    <meta name="msapplication-window"           content="width=1024;height=768" />
    <meta name="msapplication-navbutton-color"  content="#42529d" />
    <meta name="msapplication-task"             content="name=Rechercher un article;action-uri=http://recherche.doctissi
    <meta name="msapplication-task"             content="name=Rechercher un médicament; action-uri=http://sante.doctissi
```

Écouter le réseau

(F12 ou Menu Outils/Outils de développement puis onglet Réseau)



MODÈLE D'EXÉCUTION DU WEB

Du client au serveur

Il peut exister des machines intermédiaires servant de relais

- Un **proxy** (serveur mandataire) peut modifier les réponses et requêtes qu'il reçoit et peut gérer un cache des ressources demandées.
- Une **passerelle** (ou gateway) est un intermédiaire modifiant le protocole utilisé.
- Un **tunnel** transmet les requêtes et les réponses sans aucune modification, ni mise en cache.

- Utilisation de **protocoles** : SSL/TLS/… HTTPS

HTML 5

Hypertext Markup Language /
Langage de balisage d'hypertexte

HTML : historique

1991 HTML
1994 HTML 2
1996 CSS 1 + JavaScript
1997 HTML 4
1998 CSS 2
2000 XHTML 1
2002 Tableless Web Design
2005 AJAX
2009 HTML 5

HTML 5 : généralités

- Langage standardisé

• Date de première version :	28 octobre 2014
• Date de dernière version :	21 décembre 2017
• N° de version actuelle :	5.2
• Paradigme	Langage de balisage
• Auteur	WHATWG
• Développeur	WHATWG et W3C
• Version en dév.	5.3
• Influencé par	SGML
• A influencé	Wikicode, BBCode
• Écrit en	SGML
• Site web	https://www.w3.org/TR/html52/
• Extensions de fichiers	HTML: .html, .htm XHTML: .xhtml, .xht, .xml

HTML : généralités

- Language descriptif
  - Pas de séquences de contrôle
  - Description de la sémantique du document

- Balises
  - Balise ouvrante : <tag>
  - Balise fermante : </tag>
  - Les deux en une : <tag />
  - Attributs : <tag attribut="valeurs">

- Standardisé
  - W3C : http ://www.w3c.org
  - Dernière version : HTML 5
    - Strict
    - Transitional
    - Frameset
  - Validation automatique : http ://validator.w3.org
  - Balise DOCTYPE, sur la première ligne du fichier

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http ://www.w3.org/TR/html4/loose.dtd">
```

HTML, XHTML, HTML5...

- XHTML
  - Le XHTML est un langage normé (+ contraignant pour le développeur) permettant (en théorie) un standardisation de la façon de coder pour un affiche (censé être) compatible sur tous les navigateurs/OS
  - Il existe plusieurs versions : Strict, Transitional
  - Il est possible de vérifier le respect de son code sur un validateur (celui de la W3C par exemple)
  - Avis professionnel : faut tendre vers le XHTML sans trop perdre de temps avec (la compatibilité OS/navigateurs est + importante)

- HTML5
  - Le HTML5 est la nouvelle version du HTML5.
  - C'est un nouveau standard participant au web sémantique (donner du sens aux balises pour faciliter l'interprétation informatique, l'interconnexion des données du web, ..)
  - Il permet en outre : + de dynamisme, la géolocalisation, la portabilité mobile, ...

HTML : généralités

- Encapsulation de type « pile »

```
<a> <b> </b> </a>      → correct
<a> <b> </a> </b>      → incorrect
```

- Structure arborescente

Arbre minimal :

```
<html>
  <head>
    <title>…</title>
  </head>
  <body>
      …
  </body>
</html>
```

métadonnées

données

page HTML

HTML : généralités

- Commentaires

```
<!- ici mes commentaires -->
```

- HTML + XML => XHTML
  - Balises toujours fermées
  - Transformation en d'autres formats

- HTML donne un découpage selon la sémantique du document

- Mise en forme visuelle
  - en HTML : moins lisible, plus de code
  - en CSS : plus lisible, regroupement et généralisation

HTML : espacements

- Code :

```
  mon    texte,   comme\t
 ça,    j'\n
              aime.
..
beaucoup.
```


- Résultat :

```
mon  texte,  comme  ça,  j'aime...
beaucoup.
```

- Conclusions :
  - Les suites de caractère d'espacement équivaut à un seul caractère d'espacement
  - Les expressions **\t** et **\n** sont ignorés
  - Il faut utiliser des balises pour mettre en forme le texte

HTML : mise en forme du texte

- Mise en forme au niveau paragraphe

` `	Aller à la ligne
` `	Insérer un espace
`<p>…</p>`	Paragraphe
`<div>…</div>`	Cadre
`<pre>…</pre>`	Tout est reproduit tel quel

- Mise en forme au niveau phrase

`<em>…</em>`	Mise en valeur (italique)
`<strong>…</strong>`	Mise en valeur forte (gras)
`<cite>…</cite>`	Citation courte (gras italique)
`<q>…</q>`	Citation courte (entre guillements)
`<blockquote>…</blockquote>`	Citation longue (gras)
`<code>…</code>`	Extrait de code source (gras italique petit)
`<abbr>…</abbr>`	Abréviation (gras italique)
`<acronym>…</acronym>`	Acronyme (gras italique)
`<sup>…</sup>`	Mettre en exposant
`<sub>…</sub>`	Mettre en indice
`<big>…</big>`	Plus gros
`<small>…</small>`	Plus petit

Beaucoup d'autres : `<dfn>`, `<samp>`, `<kbd>`, `<var>`, `<ins>`, `<del>`, …

HTML : liens hypertextes

- Balise des liens hypertextes

`<a>…</a>`	Lien
`<a href="lien">texte</a>`	Lien externe
`<a href="mailto:address@server.com">mail</a>`	Lien courriel
`<a href="javascript:fonction()">texte</a>`	Lien javascript
`<a name="ancre" />` `<a href="lien#ancre">texte</a>`	Lien interne (ancre)
`<a … target="_blank">texte</a>`	Nouvel onglet

- Balise des liens hypertextes
  - Possibilité de lien sur une image
  - Choisir le(s) bon(s) mot(s) pour servir de lien
  - Lier les pages de manière cohérente

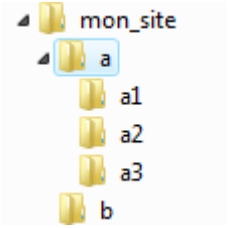
HTML : liens hypertextes

- Adresse absolue d'un lien

```
<a href="http://fressin.fr/">T. FRESSIN</a>
<a href="fressin.fr/fr/index.php">T. FRESSIN</a>
```

- Adresse relative d'un lien

```
Ma page 'index.html' est dans le dossier a.
J'ai une page 'toto.html' dans le dossier a2.
Pour créer le lien index vers toto :
<a href="a2/toto.html">Lien</a>

Ma page 'toto.html' est dans le dossier a2.
J'ai une page 'tata.html' dans le dossier b.
Pour créer le lien toto vers tata :
<a href=« ../b/tata.html">Lien</a>
```



HTML : listes

- Liste ordonnée

```
<ol>
  <li>article 1</li>
  <li>article 2</li>
  <li>article 3</li>
</ol>
```

1. article 1
2. article 2
3. article 3

- Liste non-ordonnée

```
<ul>
  <li>article 1</li>
  <li>article 2</li>
  <li>article 3</li>
</ul>
```

- article 1
- article 2
- article 3

- Liste de définition

```
<dl>
  <dt>article 1</dt>
  <dd>définition 1</dd>
  <dt>article 2</dt>
  <dd>définition 2</dd>
</dl>
```

article 1
définition 1
article 2
définition 2

HTML : tableaux

- Tableau

```
<table border="1">
  <caption> Voici le titre du tableau </caption>
  <tr>
    <th> Titre A1 </th>
    <th> Titre A2 </th>
    <th> Titre A3 </th>
    <th> Titre A4 </th>
  </tr>
  <tr>
    <th> Titre B1 </th>
    <td> Valeur B2 </td>
    <td> Valeur B3 </td>
    <td> Valeur B4 </td>
  </tr>
</table>
```

Voici le titre du tableau

Titre A1	Titre A2	Titre A3	Titre A4
Titre B1	Valeur B2	Valeur B3	Valeur B4

- Attributs utiles

`rowspan="x"`	fusionner x lignes
`colspan="x"`	fusionner x colonnes
`align`	alignement horizontal dans les th, tr, td
`valign`	alignement vertical dans les th, tr, td

HTML : images

- Image

```
<img src="path_image.ext" alt="Texte de remplacement"
  title="Texte à afficher" />
```

- Spécifications
  - Formats acceptés des images : jpeg, jpg, png, gif, svg

```
<img src="equilateral.png" srcset="equilateral.svg"/>
```

  - Principaux attributs :
    - `src`
    - `align`
    - `alt`
    - `title`
    - `width`
    - `height`

HTML : caractères spéciaux

- Codage de la page
  - Nous travaillons au format `ISO-8859-1` (*Latin-1* ou *Europe occidentale*)
- Caractères spéciaux
  - Les accentuations et symboles sont des caractères spéciaux.
  - Mal codés (en ISO ou en HTML), ces caractères seront mal affichés en sortie.

Code HTML	Code ISO	Rendu
&eacute;	&#233;	é
&egrave;	&#232;	è
&acirc;	&#226;	â
&Auml;	&#196;	Ä
&ccdeil;	&#231;	ç
&amp;	&#38;	&
&euro;	&#128;	€
&quot;	&#34;	«
	&#137;	‰
…	…	…

HTML : métadonnées

```
<head>
<title>Titre</title>

<meta name="description" lang="fr"  content="description" />
<meta name="keywords" lang="fr" content="keywords FR" />
<meta name="publisher" content="nom prenom" />
<meta name="revisit-after" content="30 days" />
<meta name="reply-to" content="courriel" />
<meta name="author" lang="fr" content="nom prenom" />
<meta name="robots" content="all" />
<meta name="distribution" content="global" />
<meta name="identifier-url" content="url" />
<meta name="date-creation-yyyymmdd" content="20070927" />
<meta name="date-revision-yyyymmdd" content="20080409" />
<meta name="verify-v1" content="code" />

<link rel="shortcut icon" href="http://url_absolue/favicon.ico" />

<script type="text/javascript">...</script>
<style type="text/css">...</style>

</head>
```

HTML5

~=

HTML + CSS + JS

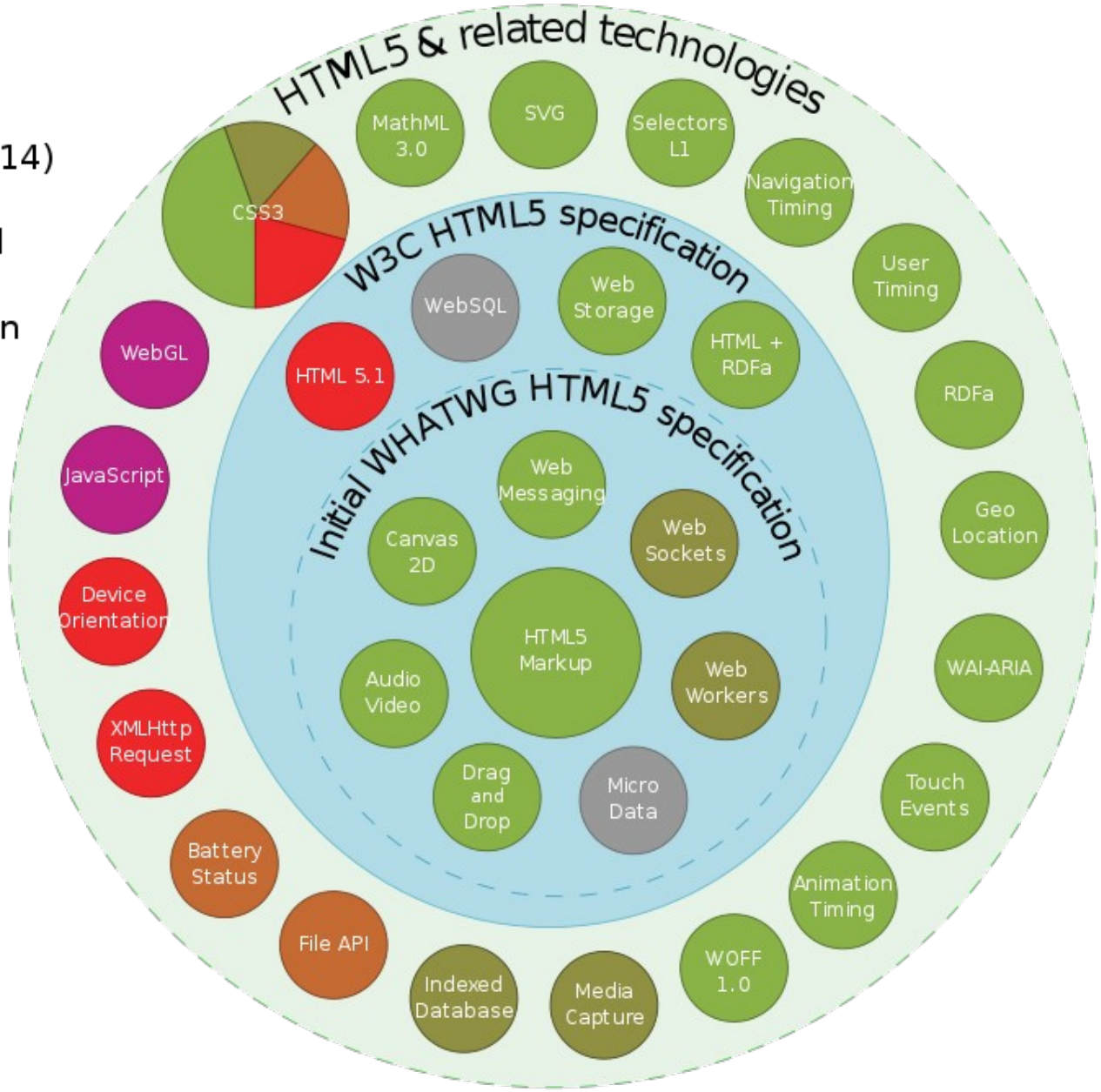
HTML 5 : nouvelles possibilités

- **Sémantique & balisage**
- **CSS 3**
- Graphics / Multimedia
- Vidéo, canvas
- Offline / Storage
- BDD, stockage offline, objet, ...
- Realtime / Communication
- Evénements, notifications, ...
- File / Hardware Access
- Drag&drop, géoloc

HTML5

Taxonomy & Status (October 2014)

- Recommendation/Proposed
- Candidate Recommendation
- Last Call
- Working Draft
- Non-W3C Specifications
- Deprecated or inactive



Sémantique & balisage

Balises plus simples et explicites

```
<!DOCTYPE html>
<html lang="fr">

<head>
  <meta charset="UTF-8">
  ...
</head>
```

```
<header> ... </header>

<nav> ... </nav>

<section>
  <article>
     ...
  </article>
  <article>
     ...
  </article>
</section>

<aside> ... </aside>

<figure> ... </figure>

<footer> ... </footer>
```

Développement web

- `main` : définit le contenu principal de la page, il doit être unique dans la page.
- `section` : définit les sections du document, telles que les chapitres, en-têtes, pieds de page.
- `article` : partie indépendante du site, comme un commentaire.
- `aside` : associé à la balise qui le précède.
- `header` : spécifie une introduction, ou un groupe d'éléments de navigation pour le document.
- `footer` : définit le pied de page d'un article ou un document. Contient généralement le nom de l'auteur, la date à laquelle le document a été écrit et / ou ses coordonnées.
- `nav` : définit une section dans la navigation.
- `figure` : définit des images, des diagrammes, des photos, du code, etc.
- `figcaption` : légende pour la balise <figure>.
- `audio` : pour définir un son, comme la musique ou les autres flux audio (streaming).
- `video` : permet d'insérer un contenu vidéo en streaming.
- `track` : permet d'insérer un sous-titre (au format WebVTT) à une vidéo affichée avec la balise vidéo.
- `embed` : définit un contenu incorporé, comme un plug in.
- `mark` : définit un texte marqué.
- `meter` : permet d'utiliser les mesures avec un minimum et maximum connus, pour afficher une jauge.
- `progress` : définit une barre de progression sur le travail en cours d'exécution.
- `time` : définit une date ou une heure, ou les deux. Cette balise a été abandonnée en octobre 2011 en faveur de la balise `data`[6] avant d'être réintroduite[7].
- `canvas` : utilisé pour afficher des éléments graphiques, il faut utiliser un script pour l'animer.
- `command` : définit un bouton. Cette balise est uniquement supportée par Internet Explorer 9. Il n'est donc pas recommandé de l'utiliser[8, 9] [source insuffisante].
- `details` : précise les détails supplémentaires qui peuvent être masqués ou affichés sur demande.
- `keygen` : permet de générer une clé (sécurisé).
- `output` : représente le résultat d'un calcul.
- `ruby`, `rt` et `rp` : annotations ruby.

De meilleurs tags sémantiques

```
<!DOCTYPE html>
<html lang="fr">

<head>
   <meta charset="UTF-8">
   ...
</head>

<body>

 <header>
  <hgroup>
    <h1>Page title</h1>
    <h2>Page subtitle</h2>
  </hgroup>
 </header>

 <nav>
  <ul>
    Navigation...
  </ul>
 </nav>
```

```
<section>
 <article>
  <header>
    <h1>Title</h1>
  </header>
  <section>
    Content...
  </section>
 </article>
 <article>
  <header>
    <h1>Title</h1>
  </header>
  <section>
    Content...
  </section>
 </article>
</section>
```

```
<aside>
 Top links...
</aside>

<figure>
  <img src="..."/>
  <figcaption>Chart 1.1</figcaption>
</figure>

<footer>
 Copyright ©
 <time datetime="2010-11-08">2010</time>.
</footer>

</body>
```

Listes autocomplétées

```
<input list="cars"/>
<datalist id="cars">
  <option value="BMW"/>
  <option value="Ford"/>
  <option value="Volvo"/>
</datalist>
```

Balises sémantiques

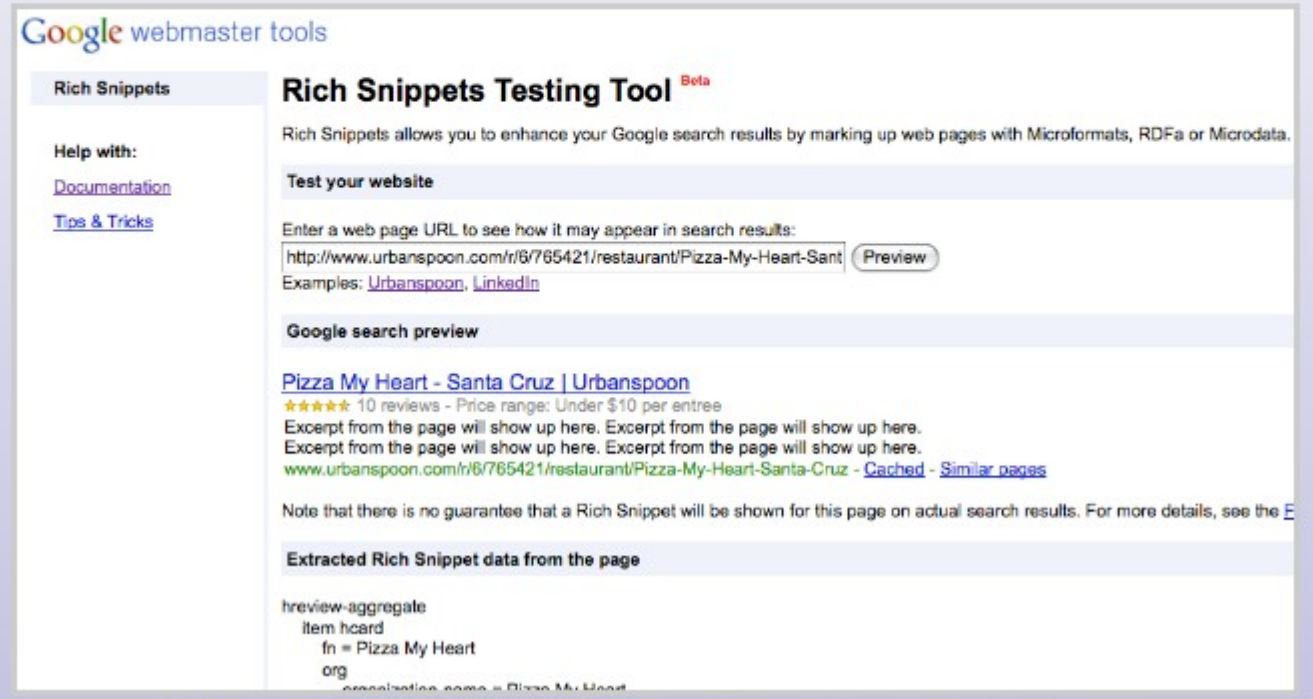
```
<meter min="0" max="100" low="40" high="90" optimum="100" value="91">A+</meter>
Your score is: A+

<progress>working...</progress>
Download is:

<progress value="75" max="100">3/4 complete</progress>
Goal is:
```

Relations et liens décrits

```
<link rel="alternate" type="application/rss+xml" href="http://myblog.com/feed"/>
<link rel="icon" href="/favicon.ico"/>
<link rel="pingback" href="http://myblog.com/xmlrpc.php"/>
<link rel="prefetch" href="http://myblog.com/main.php"/>
...

<a rel="archives" href="http://myblog.com/archives">old posts</a>
<a rel="external" href="http://notmysite.com">tutorial</a>
<a rel="license" href="http://www.apache.org/licenses/LICENSE-2.0">license</a>
<a rel="nofollow" href="http://notmysite.com/sample">wannabe</a>
<a rel="tag" href="http://myblog.com/category/games">games posts</a>
...
```

Microdata

```
<div itemscope itemtype="http://example.org/band">
 <p>My name is <span itemprop="name">Neil</span>.</p>
 <p>My band is called <span itemprop="band">Four Parts Water</span>.</p>
 <p>I am <span itemprop="nationality">British</span>.</p>
</div>
```

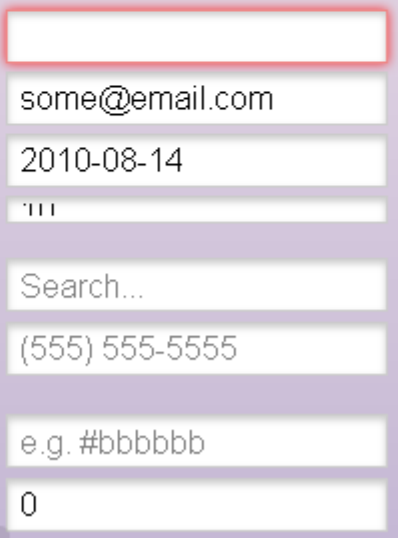


Rich Snippets Testing Tool at http://www.google.com/webmasters/tools/richsnippet

Nouveaux types de formulaires

```
<style>
  [required] {
    border-color: #88a;
    -webkit-box-shadow: 0 0 3px rgba(0, 0, 255, .5);
  }
  :invalid {
    border-color: #e88;
    -webkit-box-shadow: 0 0 5px rgba(255, 0, 0, .8);
  }
</style>
```

```
<input type="text" required />

<input type="email" value="some@email.com" />

<input type="date" min="2010-08-14" max="2011-08-14" value="2010-08-14"/>

<input type="range" min="0" max="50" value="10" />

<input type="search" results="10" placeholder="Search..." />

<input type="tel"  placeholder="(555) 555-5555"
        pattern="^\(?\d{3}\)?[-\s]\d{3}[-\s]\d{4}.*?$" />

<input type="color" placeholder="e.g. #bbbbbb" />

<input type="number" step="1" min="-5" max="10" value="0" />
```

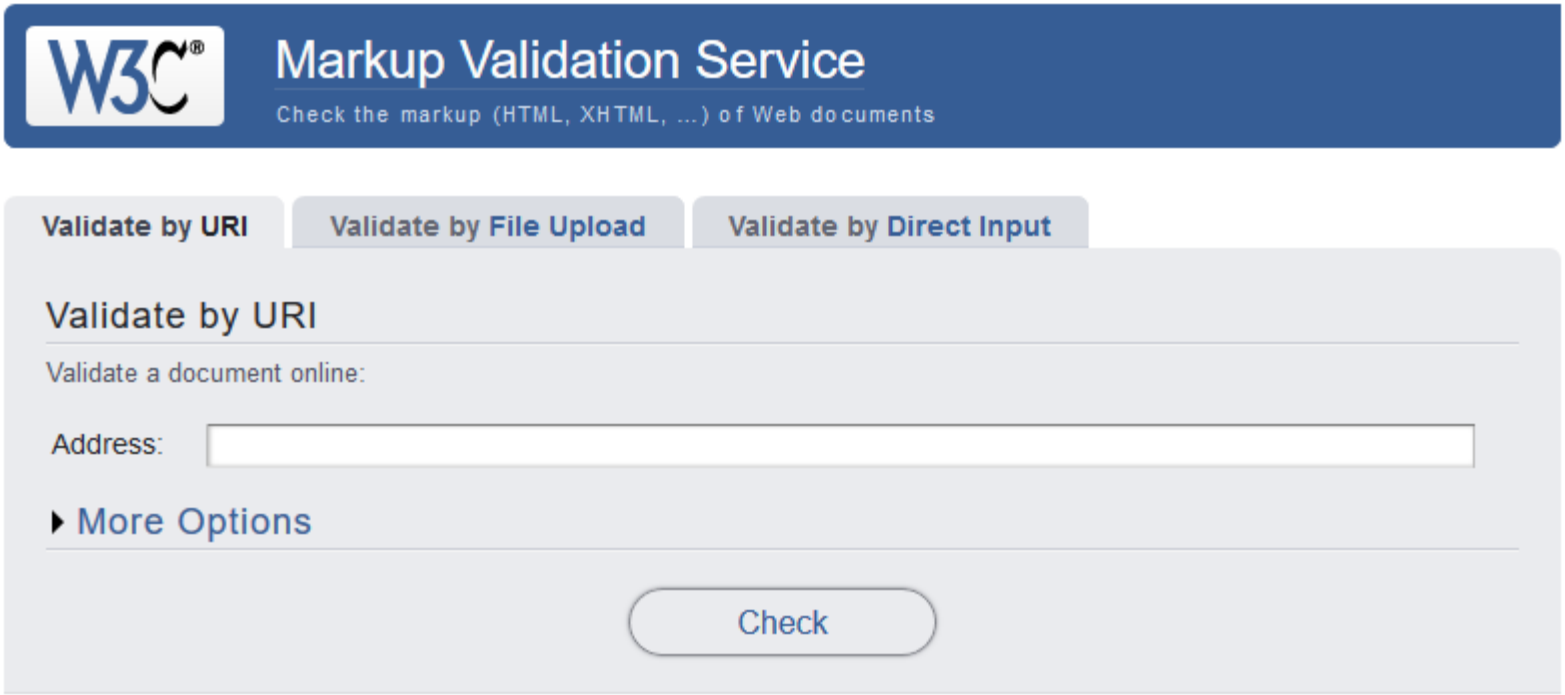


Formulaires pour mobiles

type="text"	type="number"	type="email"	type="tel"
			
Android Device	Android Device	iPhone Device	iPhone Device

Validation du code

https://validator.w3.org/



CSS

Cascading Style Sheets / Feuilles de style en cascade

CSS : utilité

- Utilités
  - Séparer la forme du fond
    - HTML décrit le fond
    - CSS décrit la forme
  - Centralisation de l'aspect visuel
  - Optimiser les modifications
    - Il suffit de modifier un style à un endroit pour l'appliquer partout
  - Factoriser/rendre plus lisible le code

CSS : où décrire les classes ?

- Description interne à la page :
  - Dans les métadonnées :

```
<head>
     <style type="text/css">
          <!--
          tous_les_styles_ici
          -->
     </style>
</head>
```

  - Dans le corps de la page :

```
<body>
     <span style="un_style">…</style>
</body>
```

- Description externe (dans un fichier css) :

```
<head>
     <link rel="stylesheet" type="text/css" href="url.css">
</head>
```

CSS : format d'une classe

- Format :

```
classe {
  attribut: valeurs; [attribut: valeurs; …]
}
```

//la classe peut correspondre à une balise html (pour que le style s'applique à cette balise)
//si la classe ne s'applique pas à une balise html spécifique, on précède le nom de la classe d'un point

- Exemple classique de classes :

```
body {
     font-family:Arial, Helvetica, sans-serif;
     font-size:12px;
     margin:0px;
}
.td_menu {
     height:25px;
     width:380px;
     vertical-align:middle;
     text-align:left;
}
a.menu:link, a.menu:visited, a.menu:active, a.menu:hover {
     color:#6A0055;
     letter-spacing: 3px;
     text-decoration: none;
}
```

CSS : utilité concrète

- Code sans CSS (avant)

```
<html>
 <head />
 <body>
  <div style="color:#6A0055; font-family:Arial;"><strong>text 1</strong></div>
  <div style="color:#6A0055; font-family:Arial;"><strong>text 2</strong></div>
  <div style="color:#6A0055; font-family:Arial;"><strong>text 3</strong></div>
  <div style="font-family:Arial;">text 4</div>
 </body>
</html>
```

- Code avec CSS (après)

```
<html>
 <head>
  <style type="text/css"><!--
   .style1 { color:#6A0055; font-family:Arial; font-weight: bold; }
   .style2 { font-family:Arial; }
  --></style>
 </head>
 <body>
  <div class="style1">text 1</div>
  <div class="style1">text 2</div>
  <div class="style1">text 3</div>
  <div class="style2">text 4</div>
 </body>
</html>
```

CSS 3

Sélecteurs CSS 3

Selectors

```
.row:nth-child(even) {
  background: #dde;
}
.row:nth-child(odd) {
  background: white;
}
```

Row 1
Row 2
Row 3
Row 4

Image-like display

```
div {
  display: inline-block;
}
```

Specific attributes

```
input[type="text"] {
  background: #eee;
}
```

Negation

```
:not(.box) {
  color: #00c;
}
:not(span) {
  display: block;
}
```

More specific targetting

```
h2:first-child { ... }

div.text > div { ... }
h2 + header { ... }
```

Webfonts

```
@font-face {
  font-family: 'LeagueGothic';
  src: url(LeagueGothic.otf);
}

@font-face {
  font-family: 'Droid Sans';
  src: url(Droid_Sans.ttf);
}

header {
  font-family: 'LeagueGothic';
}
```

LeagueGothic font with no image replacement

Multi-colonnes

```
-webkit-column-count: 2;
-webkit-column-rule: 1px solid #bbb;
-webkit-column-gap: 2em;
```

In March 1936, an unusual confluence of forces occurred in Santa Clara County.

A long cold winter delayed the blossoming of the millions of cherry, apricot, peach, and prune plum trees covering hundreds of square miles of the Valley floor. Then, unlike many years, the rains that followed were light and too early to knock the blossoms from their branches.

Instead, by the billions, they all burst open at once. Seemingly overnight, the ocean of green that was the Valley turned into a low, soft, dizzyingly perfumed cloud of pink and white. Uncounted bees and yellow jackets, newly born, raced out of their hives and holes, overwhelmed by this impossible banquet.

Then came the wind.

It roared off the Pacific Ocean, through the nearly uninhabited passes of the Santa Cruz Mountains and then, flattening out, poured down into the great alluvial plains of the Valley. A tidal bore of warm air, it tore along the columns of trees, ripped the blossoms apart and carried them off in a fluttering flood of petals like foam rolling up a beach.

This perfumed blizzard hit Stevens Creek Boulevard, a two-lane road with a streetcar line down its center, that was the main road in the West Valley. It froze traffic, as drivers found themselves lost in a soft, muted whiteout. Only the streetcar, its path predetermined, passed on...

Opacité

```
color: rgba(255, 0, 0, 0.75);
background: rgba(0, 0, 255, 0.75);
```

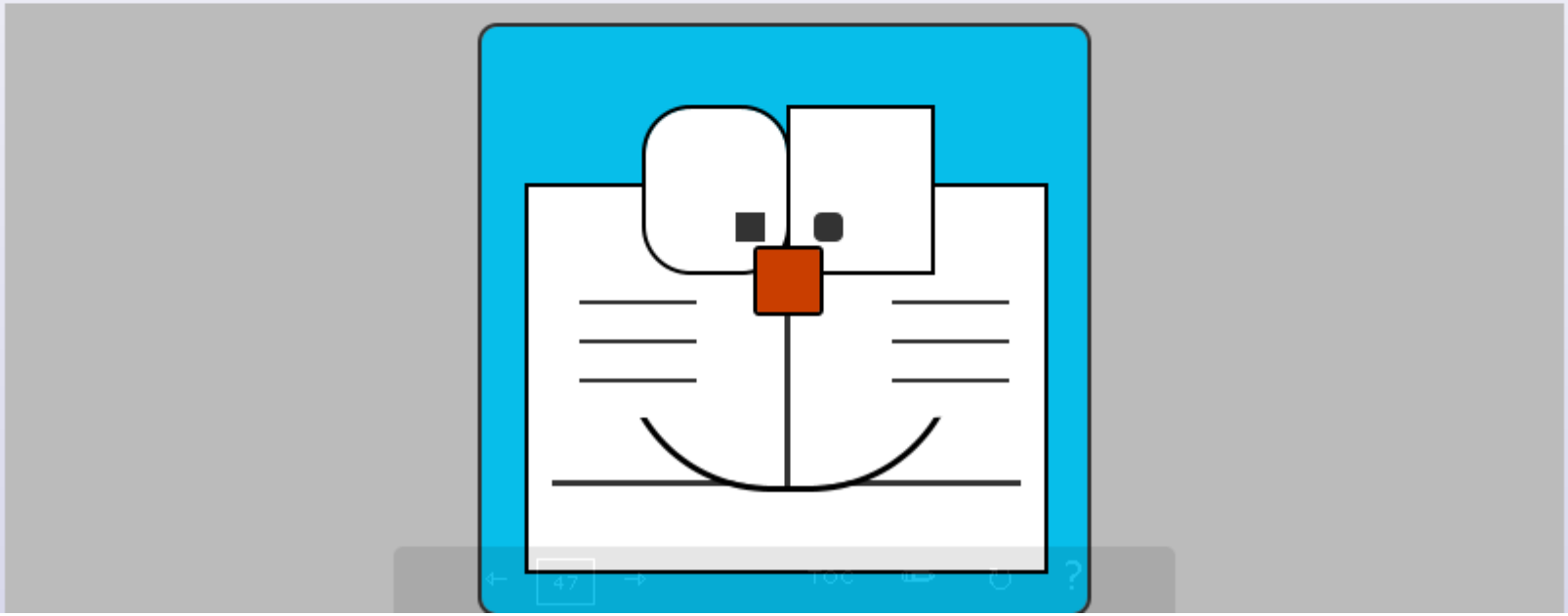
Teinte, luminosité, saturation (TLS)

```
color: hsla(
  128
  75%,
  33%,
  1.00);
```

HSL example

Angles arrondis

```
face: border-radius: 10px;
left eye: border-radius: 25px;
right eye: border-radius: 0px;
base white: border-radius: 0px;
mouth: border-radius: 0px;
nose: border-radius: 3px;
left black eye: border-radius: 0px;
right black eye: border-radius: 4px;
```



Dégradés

```
background: -webkit-gradient(linear, left top, left bottom,
                             from(#00abeb), to(white),
                             color-stop(0.5, white), color-stop(0.5, #66cc00))
```

```
background: -webkit-gradient(radial, 430 50, 0, 430 50, 200, from(red), to(#000))
```

Ombrages

```
text-shadow:
  rgba(64, 64, 64, 0.5)
  5px 5px 5px;
```

```
box-shadow:
  rgba(0, 0, 128, 0.25)
  0px 5px 5px;
```

Shadows example

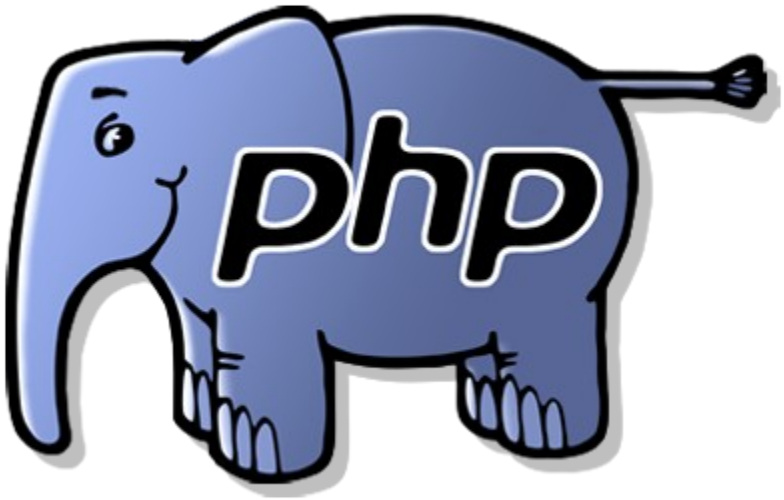
Fonds multiples

```
div {
  background: url(src/zippy-plus.png) 10px center no-repeat,
              url(src/gray_lines_bg.png) 0 center repeat-x;
}
```

Test

PHP

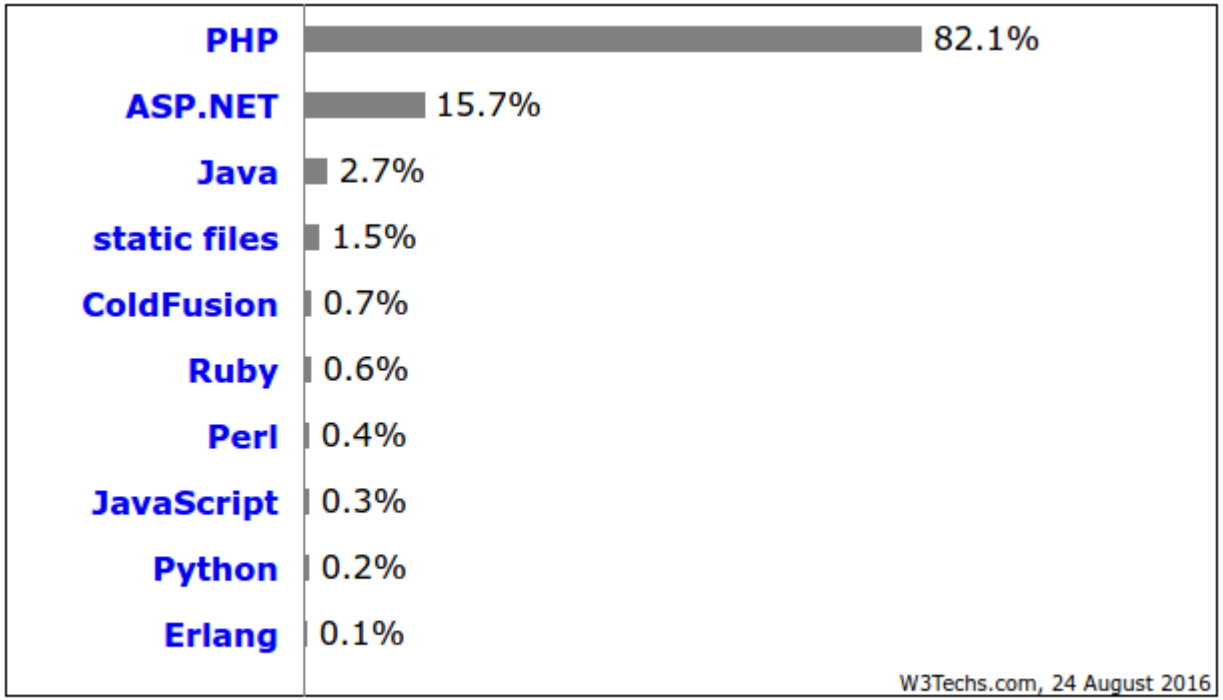
Hypertext Preprocessor



PHP : généralités

- Langage crée en 1994
- Versions actuelle :
  7.3.10 (24 septembre 2019)
- Langage de script
  - Langage interprété
  - Présence d'un interpréteur côté serveur
- Intégré au code HTML
- Syntaxe proche du C et du Java
- Interface simple avec beaucoup de SGBD

Concurrents de PHP



Popularité des langages côté serveur

PHP : modèle d'exécution

1. Le client demande une page PHP
2. Le serveur web exécute le code de la page
2a. Lancement de l'interpréteur
2b. Exécution du code
3. Le serveur web renvoie le résultat de l'exécution
4. Le client affiche le résultat

Pour le client, il est impossible de voir le code PHP.
Seul le résultat de l'exécution est récupéré par le client

PHP : du statique au dynamique

Page statique (côté serveur) :

```
<html>
<head><title>Page statique</title></head>
<body>
Nous sommes le 11-09-2001.
</body>
</html>
```

Affichage (côté client) :

Nous sommes le 11-09-2001.

Page dynamique (côté serveur) :

```
<html>
<head><title>Page statique</title></head>
<body>
Nous sommes le <?php echo date('d-m-Y'); ?>.
</body>
</html>
```

Nous sommes le 01-04-2011.

//Date du jour affichée

PHP : éléments du langage

- Le code PHP s'intègre dans l'HTML de cette façon :

```
ici html <?php ... ?> ici html
```

- Les instructions PHP se finissent par un point virgule

```
instruction_php;
```

- Les commentaires peuvent s'écrire de 2 façons :

```
/* commentaire */    (commentaire bout-en-bout)
// commentaire       (commentaire jusqu'en fin de ligne)
```

PHP : variables

- Les variables sont préfixées par le signe dollar $

```
$nom_variable
```

- Le nom suit les règles classiques (pas d'espace, sensible à la casse)

- Le type des variables n'a pas besoin d'être déclaré (typage implicite)

```
$my_var_03 = 54 ; //entier
$my_var_03 = "pif"; //maintenant, une chaine
```

- Attention aux fautes de frappes dans les noms de variables

- Fonctions sur les variables :

```
isset($var) : renvoie true si $var existe
empty($var) : renvoie true si $var est vide
unset($var) : détruit $var
```

PHP : types des variables

- Entiers : `$var=54;`
- Flottants : `$var=54.3;`
- Chaînes : `$var="54";`
`$var='54';`
- Booléens : `$var=false;`
`$var=true;`

- Fonctions de test :
`is_integer ($var), is_string ($var), … :` renvoie true si $var est un entier, une chaîne, . . .

- On définit une constante à l'aide de la commande `define`
```
define("PI", 3.14);
```
On les utilise directement (sans $) : `echo(PI);`

PHP : chaîne de caractères

- Délimitées par " (contenu interprété)
- Délimitées par ' (contenu non interprété)
- Les unes peuvent contenir les autres
- Sinon backslash

```
$var="e"; echo "pal$var";
$var='e'; echo 'pal$var';
$var="citation : 'ok'";
$var="citation : \"ok\"";
```

- Concaténation : `$v1="pas"; $v2="pas";`

```
echo $v1.$v2;
echo $v1." ".$v2;
```

- Accès à un caractère : `$date="2011-04-01";`
- Longueur de la chaîne
- Sous-chaîne avec `substr($str,start[,len])`

```
echo $date{5}.$date{6};
echo strlen($date);
echo substr($date,5);
echo substr($date,3,3);
```

- Effacer espaces début/fin : `$var=" abc ";`

```
echo trim($var);
```

PHP : tableaux

- Tableaux : `$tab = array("foo"=>"bar", 12=>true);`
  `$tab["foo"]="bar"; $tab[12]=true;` } //lignes identiques

```
<?php
// Création d'un tableau simple.
$array = array(1, 2, 3, 4, 5);
print_r($array);

// On efface tous les éléments, mais on concerne le tableau :
foreach ($array as $i => $value) {
    unset($array[$i]);
}
print_r($array);

// Ajout d'un élément (notez que la nouvelle clé est 5, et non 0).
$array[] = 6;
print_r($array);

// Ré-indexation :
$array = array_values($array);
$array[] = 7;
print_r($array);
?>
```

```
Array
(
    [0] => 1
    [1] => 2
    [2] => 3
    [3] => 4
    [4] => 5
)
Array
(
)
Array
(
    [5] => 6
)
Array
(
    [0] => 6
    [1] => 7
)
```

PHP : opérateurs

Arithmétiques :

```
echo ((1+2-3)*2)/2;
echo 7%3;                    //affiche 1
$var=1; echo $var++;
$var=3; echo $var--;
```

Affectation :

```
echo $var=1;
$var='aa'; echo $var.='b';
$var=1; echo $var+=3;
$var=2; echo $var-=1;
$var=2; echo $var*=2;
$var=2; echo $var/=2;
$var=7; echo $var%=3;
```

Affectaison :

```
==, ===          //égal, identique
!=, <>, !==    //diff, diff, diff ou pas même type
<, >, >=, <=
```

- Logiques :

```
and, &&
or, ||
xor
!
```

- Conditionnel :

```
… ? … : …
```

PHP : conditionnelles et boucles

```
$var=1
if ($var<0) {
  echo "h1";
}
elseif ($var<-3) {
  echo "h2";
}
else {
  echo "h3";
}
```

```
$tab[0]=1;
$tab[1]=1;
$tab[2]="coucou";
for($i=0;$i<3;$i++) {
  echo $tab[$i];
}
```

```
$var="coucou";
switch ($var) {
  case "maman" : echo "h1"; break ;
  case "papa" : echo "h2"; break ;
  default : echo "h3"; break;
}
```

```
$i=0;
while($i<5) {
  $i++; echo $i;
}
```

PHP : fonctions

- Définition :

```
function nomFonction ($param1, $param2, …) {
    …
    return …;
}
```

- Remarques :
  - Pas de type pour les paramètres ou la valeur de retour
  - Nombre fixé de paramètres
  - Le nom ne commence pas par $
  - Le nom est insensible à la casse
  - Le résultat est renvoyé avec la commande `return`
  - Une seule valeur de retour
  - Passage des paramètres par valeur (par défaut)
  - Les variables utilisées à l'intérieur d'une fonctions sont détruites à la fin, sauf si on les définit avec `global`

PHP : exemples de fonctions

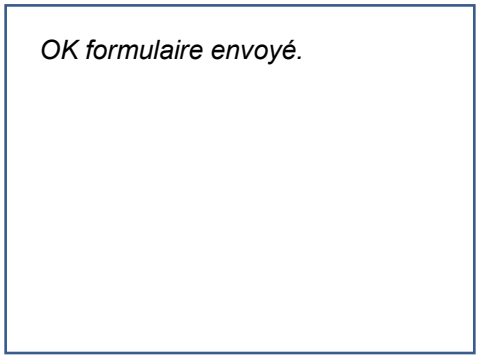
```
function math($nb) {
$nb++; $nb++;
return $nb--;
}
echo math(3);
echo math(-1);
```

```
function test($var) {
switch($var) {
          case 'maman': echo 'mère'; break;
          case 'papa': echo 'père'; break;
          default : echo 'non défini'; break;
}
}
echo test('maman');
```

```
function fonc() {
global $var ;
$var=4;
}
$var=0;
fonc() ;
echo $var ;
```

HTML & PHP : utilisation des formulaires

Page de formulaire



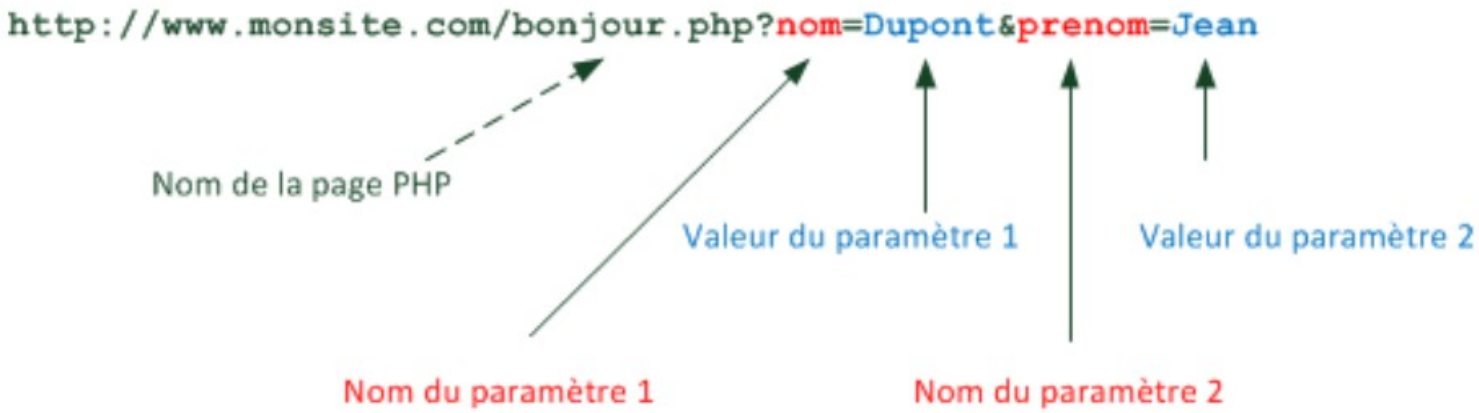
Page de résultat

Transmettre des données entre pages Web

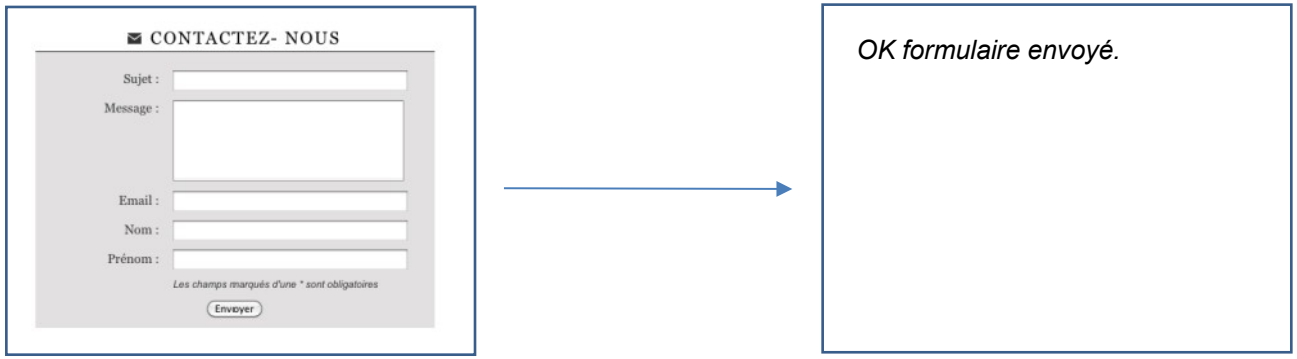
Une *URL* (Uniform Resource Locator) sert à représenter une adresse sur le Web.

- Une URL simple :

- Une URL avec *query string* (chaîne de requête) :



HTML & PHP : utilisation des formulaires



Page de formulaire

Page de résultat

- 2 méthodes de dialogue possibles :
  - GET : données encodées dans l'URL (non persistantes)

    `http://www.google.com/search/search.php?q=fnac`
  - POST : données persistantes et cachées (pas de navigation avec Précédent/Suivant)
- L'objectif dans les 2 cas est de récupérer les données envoyées :
  - `echo $_GET['q'];`
  - `echo $_POST['q'];`
  - `echo $_REQUEST['q']; // affiche du GET ou POST`

Avantages & inconvénients des 2 méthodes

La méthode GET :

- se prête bien à un site en développement car on peut facilement contrôler les valeurs et noms de variables du formulaire.
- Il est facile de créer un lien <a> vers une page traitant un formulaire en méthode GET et d'y envoyer des données via le query string.

La méthode POST :

- est plus propre car les valeurs ne sont plus affichées dans la barre d'adresse du navigateur
- Attention : ces informations ne sont pas vraiment cachées pour autant

HTML & PHP : utilisation des formulaires

- Exemple de formulaire :

```
<form action="page_resultat.html" method="POST">
Prenom :  <input type="text" name="prenom" /><br />
Nom :     <input type="text" name="nom" /><br />
          <input type="submit" value="ENVOYER" />
</form>
```

- Différents types des balises INPUT :
  - `text` zone de texte sur une seule ligne
  - `password` idem, mais avec achage d'étoiles
  - `file` sélection d'un fichier
  - `checkbox` case à cocher
  - `button` bouton simple (pas d'action sans javascript)
  - `hidden` champ "texte" caché
  - `radio` bouton d'option
  - `reset` bouton de remise à zéro
  - `submit` bouton de soumission

PHP : structures de contrôle

```php
<?php

$lang = strtolower($_POST['lang']);
$is_lang_fr = $lang === 'fr';
$is_lang_en = $lang === 'en';

if ($is_lang_fr)
    echo 'Vous parlez français !';
elseif ($is_lang_en)
    echo 'You speak English!';
else
    echo 'Je ne vois pas quelle est votre langue !';
```

PHP : sessions

- On veut parfois garder de l'information entre plusieurs pages :
  - Login / Password
  - Préférences de navigation
  - Sélection de produits à acheter (panier, ...)

  On utilise donc les sessions PHP.

- Les sessions permettent de stocker des informations côté serveur
  - Elles sont identifiées par un numéro qui reste valide tant que le visiteur reste connecté
  - Le numéro est transmis au serveur soit dans l'URL, soit dans un cookie
  - Les données se placent et se récupèrent dans `$_SESSION`, comme pour les formulaires

```
echo $_SESSION['login']=$_POST['login'];
```

- Les sessions utilisent les cookies : il faut donc ouvrir la session avant d'afficher quoi que ce soit. La session existe dès qu'elle est crée et jusqu'à ce qu'elle soit détruite.

Création (et réouverture) : `session_start()`
Destruction : `session_destroy()`
Note : les session s'autodétruisent après un certain temps (glmt 30 min)

PHP orienté objet

Exemple 1

```php
<?php
class Voiture {

  private $marque;
  private $couleur;
  private $immatriculation;

  // un getter
  public function getMarque() {
       return $this->marque;
  }

  // un setter
  public function setMarque($marque2) {
       $this->marque = $marque2;
  }

  // un constructeur
  public function __construct($m, $c, $i)  {
   $this->marque = $m;
   $this->couleur = $c;
   $this->immatriculation = $i;
  }

  // une methode d'affichage.
  public function afficher() {
    // À compléter dans le prochain exercice
  }
}
?>
```

PHP orienté objet

Exemple 2 (1/2)

```php
<?php

class Perso {
    public const PV_INITIAL = 2000;

    private $pv;

    // Constructeur : celui-ci est appelé lors de l'instanciation de l'objet.
    // Ce constructeur possède un paramètre optionnel.
    public function __construct($pv = false) {
        if (!is_numeric($pv) || $pv < 0 || $pv > 100000000)
            $this->pv = self::PV_INITIAL;
        else
            $this->pv = $pv;
    }

    // Accesseurs
    public function getPv() {
        return $this->pv;
    }

    public function isDead() {
        return $this->pv == 0;
    }
}
```

PHP orienté objet

Exemple 2 (2/2)

```php
// Création d'une classe enfant de Perso
class Magicien extends Perso {
    private $magie;
}

// Création d'une instance de classe
$perso = new Perso(1000);

// Utilisation de l'objet
echo 'Votre personnage a ' . $perso->getPV() . ' PV.';

// Constantes de classes
echo 'Le PV par défaut attribué à un nouveau personnage est de ' . Perso::PV_INITIAL . '.';

// Destruction de l'objet
unset($perso);
```

MySQL

SQL : définition

- SQL est un langage puissant de requête

- Il permet de faire une demande complexe à une BDD dans un langage proche de l'anglais
- On l'utilise pour récupérer, ajouter, supprimer et créer des données dans une BDD
- Les BDD utilisent des tables :
  - chaque ligne est un enregistrement de champs avec des valeurs
  - les requêtes se font sur ces champs

- MySQL est un SGBDD puissant
- Il est utilisé principalement dans le cadre des BDD sur internet (sites, blogs, etc.)
- C'est une application libre de droit, très populaire

- phpmyadmin est un outil de gestion de BDD qui embarque le SBGDD MySQL

SQL : requêtes basiques

- Requête interrogative

```
SELECT * FROM PersosXVI;
```

- Requête sélective

```
SELECT Nom, Prenom FROM PersosXVI;
```

Id	Nom	F
1	CARTIER	J
2	CALVIN	J
3	CHASTEL	J
4	PARE	A

- Requête restrictive

```
SELECT * FROM PersosXVI WHERE Prenom = 'Jean';
```

- Décompte

```
SELECT count(*) FROM PersosXVI;
```

- Somme

```
SELECT SUM(Age) FROM PersosXVI;
```

On peut complètement mélanger les types de requêtes

SQL : requêtes complexes

Table 'PersosXVI'

Id	Nom	Prenom	Age	Activite	Roi
1	CARTIER	Jacques	44	Explorateur	3
2	CALVIN	Jean	26	Réformateur	1
3	CHASTEL	Jean	19	Assassin	4
4	PARE	Ambroise	44	Chirurgien	2

Table 'RoiXVI'

Id	Nom
1	Henri II
2	Charles IX
3	Francois Ier
4	Henri IV

- Requête croisée

```
SELECT p.Nom, p.Activite FROM PersosXVI p, RoiXVI r
WHERE p.Roi = r.Id AND r.Nom = 'Francois Ier'
ORDER BY p.Nom DESC;
```

On peut complètement mélanger les types de requêtes

SQL : manipulation des données

Table 'PersosXVI'

Id	Nom	Prenom	Age	Activite
1	CARTIER	Jacques	44	Explorateur
2	CALVIN	Jean	26	Réformateur
3	CHASTEL	Jean	19	Assassin

- Requête d'insertion

```
INSERT INTO PersosXVI(Nom, Prenom, Age, Activite)
VALUES('PARE', 'Ambroise', 44, 'Chirurgien');
```

- Requête de mise à jour

```
UPDATE PersosXVI SET Age = 43 WHERE Nom = 'CARTIER';
```

- Requête de suppression

```
DELETE FROM PersosXVI WHERE Prenom = 'Jean';
```

MySQL & PHP : manipulation des données

- Connexion

```
$mysqli = new mysqli("localhost", "user", "password", "base");
//$cnx est false en cas d'erreur de connexion
```

- Choix de la base

```
mysqli_select_db($mysqli, "base");
```

- Exécution

```
$result = $mysqli->query("SELECT Name FROM City LIMIT 10")
//$resultat a pour valeur false en cas d'erreur
```

- Récupération des résultats

```
$row = $result->fetch_object();
echo $row->prenom;
//si un seul résultat à afficher

while( $row = $result->fetch_object() ) {
  $results[] = $row;
}
//si plusieurs lignes à afficher
```

SQL : fonctions annexes

- Nombre de résultat d'un SELECT

```
echo $mysqli->num_rows;
```

- Nombre de lignes affectées par un INSERT, UPDATE ou DELETE

```
echo $mysqli->affected_rows;
```

- Dernier incrément d'un champ

```
echo $mysqli→insert_id;
```

Protection des chaînes de caractères :

```
$city = $mysqli->real_escape_string($city);
```

Fermeture de la connexion :

```
mysqli_close($mysqli);
```