



Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Centre Universitaire de Mila
Institut des Sciences et de la Technologie



Développement Web Avancé

Département MI

s.meghzili@centre-univ-mila.dz



Présentation de l'auteur

- **Dr. Said MEGHZILI**
- **Maitre de Conférences B**
- **Centre universitaire de Mila**
- **s.meghzili@univ-mila.dz**



Objectifs du Cours

Ce **cours** vous apporte les connaissances et compétences nécessaires pour :

- Mener à bien un projet **SpringBoot/Angular**
- Mettre en place la sécurité côté client Angular
- Développer une application **Java EE** (Web) en réalisant la persistance des objets avec le Framework **Hibernate**
- Savoir tester séparément le service back-end Spring et le client Angular



Pré-requis

Pour pouvoir tirer le maximum de ce cours, il faut connaître :

- **Matière développement Web (2 année info)**
- **Matière programmation orienté objet (2 année info)**



- **Master 1 en Informatique, spécialité STIC**



L'évaluation finale se fait à travers:

- **Un examen final, qui compte pour 67% de la note finale.**
- **Évaluation continue à raison de 33% restant (note TP –Mini-projet, Interro).**



Contenu du Cours (1)

- **Chapitre 1 : Rappels sur les technologies du Web**
 - **1.1 Introduction**
 - *L'internet et le Web*
 - *Le protocole HTTP*
 - **1.2 Développement Coté Client**
 - *Le langage XHTML*
 - *Les feuilles de style CSS*
 - *Programmation Javascript et Ajax*
 - **1.3 Développement Coté Serveur**
 - *Le langage PHP*
 - *Le langage JSP et les servlets*
 - *Le langage XML*



Contenu du Cours (2)

- **Chapitre 2 : Les frameworks de développement Web**
 - ⌚ *Définition*
 - ⌚ *Architecture et fonctionnement*
 - ⌚ *Installation et configuration*
 - ⌚ *Modèle Vue Contrôleur (MVC)*
 - ⌚ *Gestion des bases de données*
 - ⌚ *Gestion des formulaires*
 - ⌚ *Etude de cas*



- **Chapitre 3 : *Mise en place d'un site Internet***
 - ⌚ *L'hébergement de sites Web*
 - ⌚ *Le référencement Internet*
 - ⌚ *Généralités sur la sécurité des sites Web*



1. **TP 01: Préparation de la plateforme de développement
Création du premier projet Angular/SpringBoot**
1. **TP 02: Développement d'une application Web CRUD en
utilisant Angular/SpringBoot**
2. **TP 03: Authentification et gestion de rôles avec
Angular/SpringBoot**



Mini-projets (1)

- 1. site web dynamique pour une agence de location de voiture**
- 2. site web dynamique d'une Auto-école**
- 3. site web dynamique pour gérer les rendez-vous dans une clinique médicale**
- 4. site web dynamique pour la gestion des examens QCM en ligne**
- 5. site web dynamique pour la réservation en ligne des chambres d'hôtels**
- 6. site web dynamique pour le covoiturage**
- 7. site web dynamique pour la mise en relation entre clients et professionnels**
- 8. site web dynamique pour un magasin en ligne**
- 9. site web dynamique pour une agence de voyage de tourisme**
- 10. site web dynamique pour une agence immobilière**
- 11. site web dynamique pour la gestion d'un laboratoire de recherche**
- 12. site web dynamique pour les petites annonces**
- 13. site web dynamique pour les offres et demandes d'emplois**



Mini-projets (2)

14. site web dynamique pour la vente de matériels informatiques
15. site web dynamique pour la gestion de scolarité
16. site web dynamique pour la vente aux enchères
17. site web dynamique pour la communication entre les enseignants et les étudiants en ligne
18. site web dynamique pour la gestion et le suivi des journées scientifiques
19. site web dynamique pour l'achat et la vente d'une pharmacie
20. site web dynamique pour l'organisation d'une conférence internationale
21. site web dynamique pour un centre de formation professionnel
22. site web dynamique pour un complexe sportif
23. site web dynamique pour la gestion des inscriptions des bacheliers
24. site web dynamique pour la gestion des écoles
25. site web dynamique pour la gestion d'un parc animalier
26. site web dynamique pour l'assistance automobile : intervention et dépannage



Remarques:

- **Les étudiants doivent se regrouper en trinôme et choisir un sujet de la liste.**
- **Les étudiant peuvent aussi proposer des sujets de leurs choix.**
- **La consultation des projet déterminera la note finale de TP.**



L'internet et le Web (1)

L'internet : C'est un ensemble d'ordinateurs reliés a l'échelle mondiale.

C'est une **infrastructure** sur laquelle repose des **services** tel que :

- Transfert de mail,
- Transfert de fichiers,
- Service DNS,
- Service de téléphonie (VOIP),
- Service de visioconférence,
- Le Web (World Wide Web),
-

Le Web : C'est un système qui permet de **publier** et **consulter** des pages web et des ressources (images vidéo ..etc) sur l'internet.

Le web se compose d'un ensemble de **pages** web et **ressources** reliés par des liens appelés **hyperliens**. Internet **≠** Web



L'internet et le Web (2)

- **Une Page Web** : C'est un document généralement codé en **HTML** qui peut être consulté en utilisant un **navigateur Web** (**Google chrome**, **Firefox**, **Internet explorer** ..etc).

- **Un Site Web** : C'est un ensemble de **pages Web** qui présente quelque chose. C'est en quelque sorte une **vitrine**.

- **Une Application Web** : C'est un logiciel (interaction, traitement, base de données ..etc) **hébergé** dans un **serveur web**, et qui utilise des pages web comme **interface**.

On accède a l'application web en utilisant un navigateurs Web.

L'avantage des applications web est qu'on na pas besoin d'installer l'application sur les PC des utilisateurs, ils utilisent un simple navigateurs web pour accéder a l'application.



L'internet et le Web (3)

- Chaque **ressource** sur le Web est identifié par un **URI** (Uniform Resource Identifier).
- Une ressource peut être **abstraite** ou **physique**, par exemple :
 - Une **image** ou un **documents** électronique,
 - Un **serveur** ou une passerelle **HTTP**
 - Equation mathématique
 - Un être humain
 - Un livre dans une bibliothèque
 - Des types de relations (par exemple, "parent" ou "employé")
 - Des valeurs numériques (1, 2, 3, infinit ..etc)



URI (1)

- Un **identifiant** de ressources uniforme (URI, Uniform Resource Identifier) est une **séquence compacte de caractères**, Par exemple :

<ftp://ftp.is.co.za/rfc/rfc1808.txt>

<http://www.ietf.org/rfc/rfc2396.txt>

<https://doi.org/10.1007/s11277-019-06130-6>

[ldap://\[2001:db8::7\]/c=GB?objectClass?one](ldap://[2001:db8::7]/c=GB?objectClass?one)

<mailto:John.Doe@example.com>

<news:comp.infosystems.www.servers.unix>

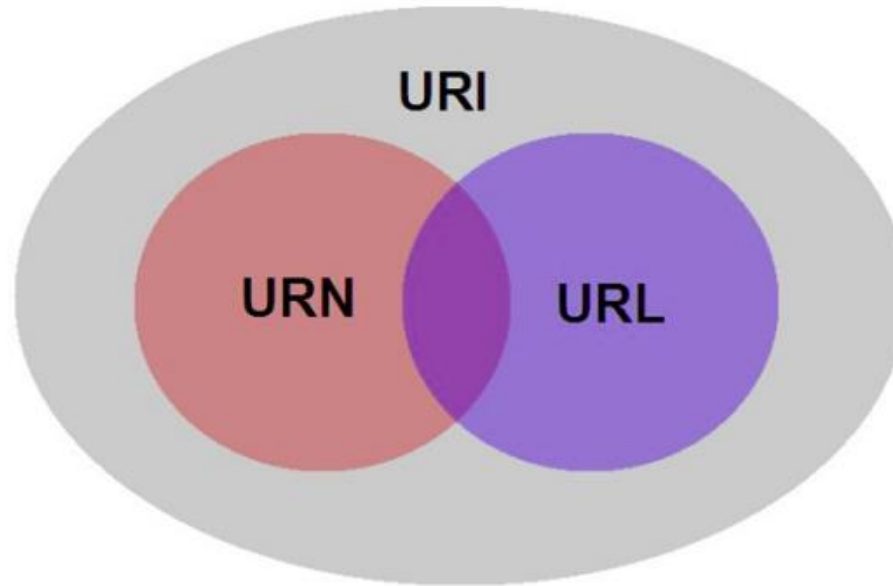
<tel:+1-816-555-1212>

<telnet://192.0.2.16:80/>

<urn:oasis:names:specification:docbook:dtd:xml:4.1.2>



URI (2)



Un **URI** peut encore être classé comme localiseur (**URL** : Uniform Resource Locator), comme nom (**URN** : Uniform Resource Name) , ou **les deux**.



URI (3)

- Un **URN** est un identifiant **unique** dans le **temps** et l'**espace** pour une ressource, il doit permettre d'identifier une ressource de manière **permanente**, même si la ressource est **déplacée** ou **supprimée**,
par exemple :

urn:ietf:rfc:3986

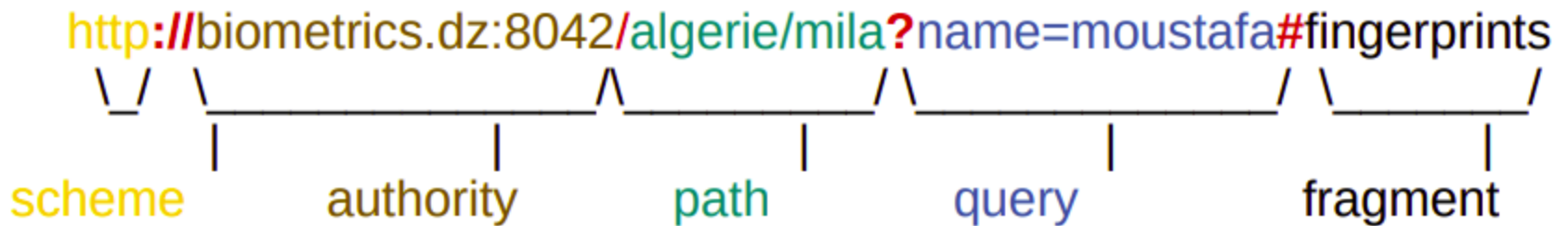
- Un **URL** se réfère à l'ensemble des URI qui, en plus de l'identification de ressource, **donnent le moyen de localiser la ressource**, par exemple :

http://www.centre-univ-mila.dz/**ar**/index.php



URI (4)

« Exemple d'un URI »



- Une URL a la structure suivante :

Le Protocole	Identifiant et Mot de passe (Optionnel)	Nom du serveur	Port (optionnel si le port = 80)	Chemin de la ressource
https://	User:password@	www.centre-univ-mila.dz	:80	/ar/index.php

<https://www.centre-univ-mila.dz?param1=valeur1¶m2=valeur2>

<https://www.centre-univ-mila.dz?param1=valeur1¶m2=valeur2#ancre1>



- Les sites et applications Web sont **hébergés** (stockés) dans des machins appelés **Serveurs Web** (web serveur).

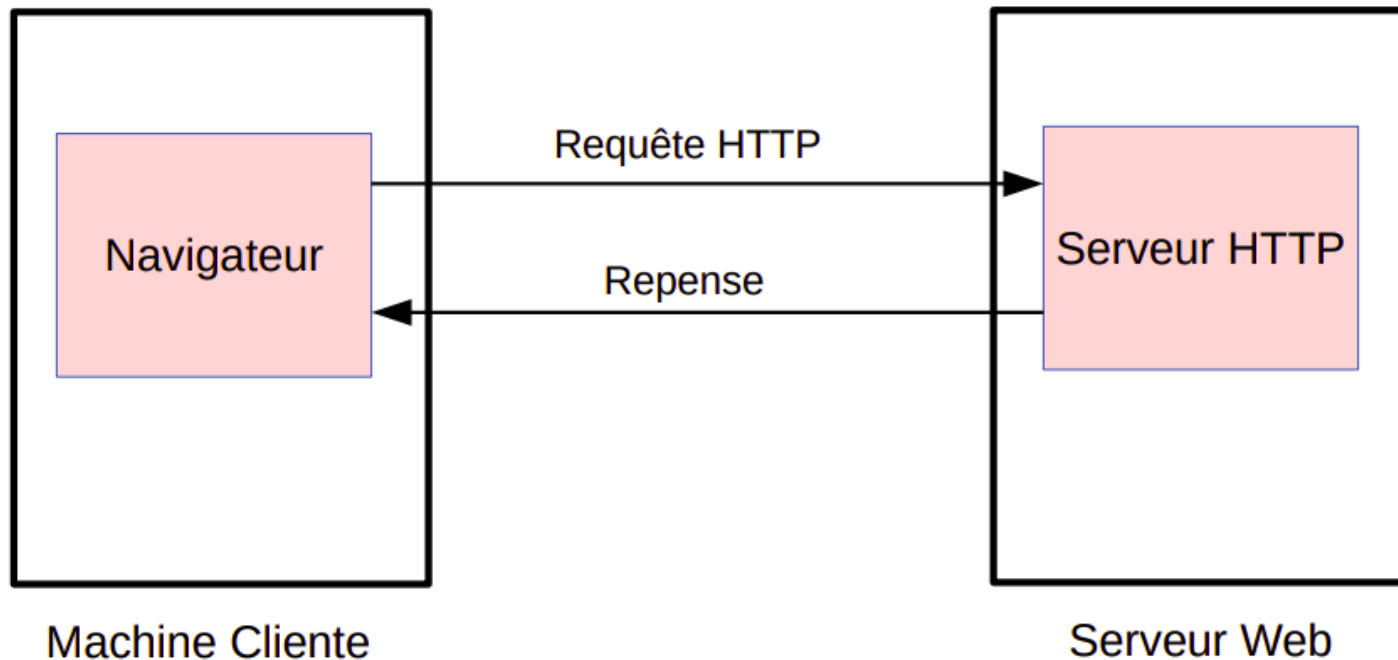
On appelle serveur Web :

- Le **programme** qui gère les requêtes Web par exemple Apache, Jboss, IIS, nginx ..etc.
- Le **matériel** qui stocke les pages du site ou de l'application Web



Protocole HTTP

- Pour communiquer avec un serveur Web et obtenir les ressources, un client web utilise le protocole **HTTP** (HyperText Transfer Protocol).
- HTTP est un protocole client-serveur qui repose sur le protocole TCP/IP.



Il existe plusieurs version du protocole **HTTP** : HTTP 0.9, HTTP 1.0, HTTP 1.1, et HTTP 2.0.

- **HTTP 0.9** : un version très simple qui permet de récupérer une ressource web a partir d'un serveur Web en utilisant la méthode **GET**. Le protocole HTTP 0.9 procède comme suit :

- 1) Le client ouvre une connexion **TCP/IP** avec le Serveur Web
- 2) Le Client envoi le requête en utilisant la méthode **GET**
- 3) Le serveur envoi une repense au client
- 4) Le serveur ferme la connexion

Protocole HTTP

Il existe plusieurs version du protocole HTTP : HTTP 0.9, HTTP 1.0, HTTP 1.1, et HTTP 2.0.

- **HTTP 1.0** (RFC 1945) : Il utilise le même principe de connexion que HTTP 0.9. cependant il définit d'autre méthodes (**POST, HEAD**)
- **HTTP 1.1** (RFC 2616) : Il permet d'envoyer plusieurs requêtes avec une seule connexion TCP. Il définit aussi d'autre méthodes (**PUT, DELETE, TRACE, OPTIONS**)
- **HTTP 2.0** (RFC 7540): Il Conserve la majorité de la syntaxe de HTTP 1.1 mais apporte des améliorations notamment dans la **réduction du temps** de chargement des pages Web.



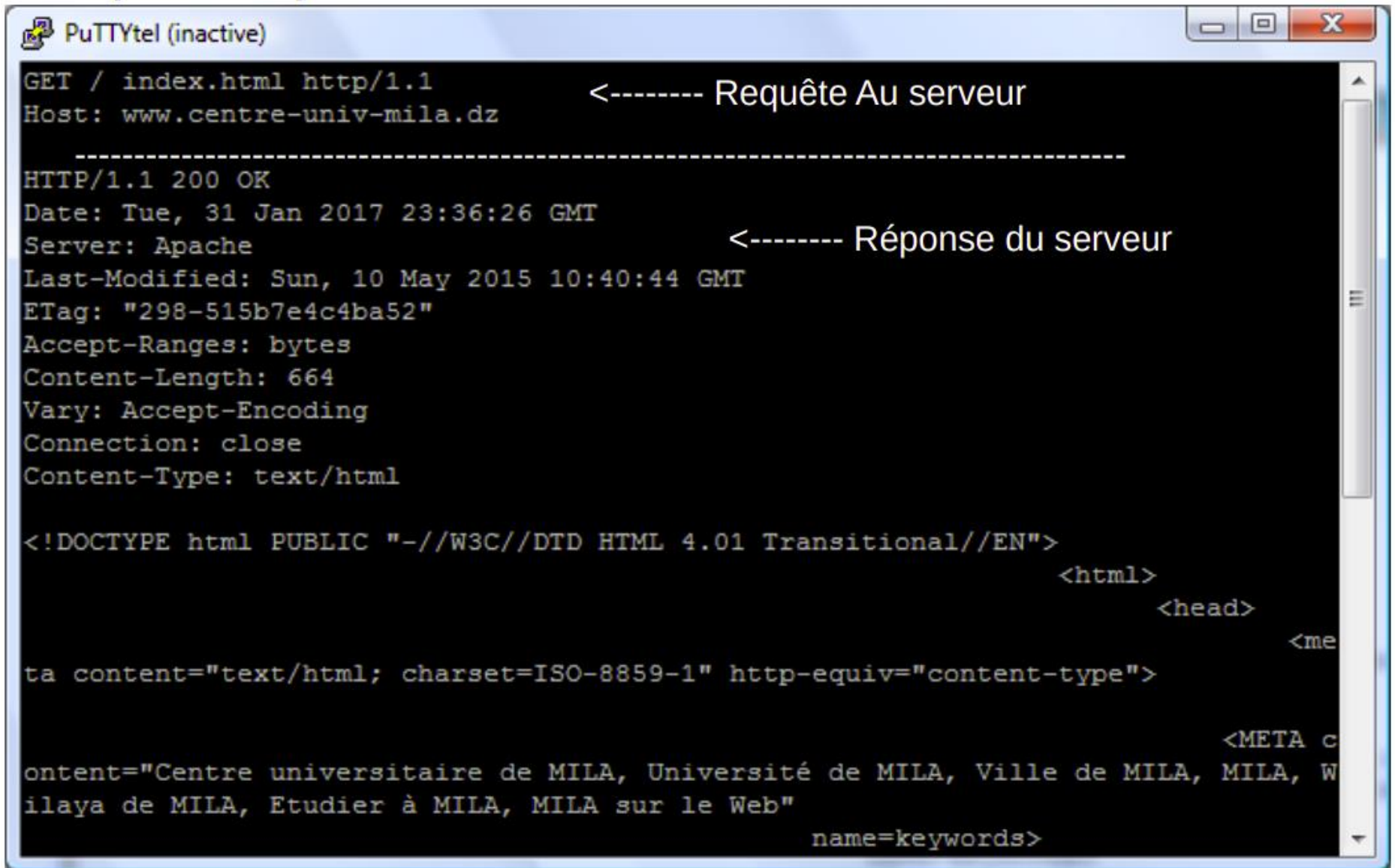
Protocole HTTP

Méthode	Description
GET	Permet de demander une ressource identifié par sont URI.
HEAD	Permet de demander des informations de l'entête d'une ressource Web sans demander la ressource elle même. L'entête contiens des information sur le serveur Web (Nom, version) hébergeant la ressource , et sur la ressource elle même (date de la dernière modification, Type de la ressource, taille ..etc).
POST et PUT	Permettent d'envoyer des information a un serveur Web.
DELETE	Permet de supprimer une ressource Web identifié par sont URI.
CONNECT	Utilisé pour se connecter a un serveur Web à partir d'un serveur Proxy.
TRACE	Permet au client web de connaître le contenu du message reçu par le destinataire final de sa requête et ainsi déterminé les différents proxy qui ont acheminé le message du client jusqu'au destinataire final.
OPTIONS	Permet de demander les options fournis par le serveur Web.



Protocole HTTP

Exemple de requêtes HTTP :



The screenshot shows a PuTTYtel terminal window with the following text:

```
GET / index.html http/1.1
Host: www.centre-univ-mila.dz

-----
HTTP/1.1 200 OK
Date: Tue, 31 Jan 2017 23:36:26 GMT
Server: Apache
Last-Modified: Sun, 10 May 2015 10:40:44 GMT
ETag: "298-515b7e4c4ba52"
Accept-Ranges: bytes
Content-Length: 664
Vary: Accept-Encoding
Connection: close
Content-Type: text/html

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
                                <html>
                                    <head>
                                        <meta content="text/html; charset=ISO-8859-1" http-equiv="content-type">
                                            <META c
ontent="Centre universitaire de MILA, Université de MILA, Ville de MILA, MILA, W
ilaya de MILA, Etudier à MILA, MILA sur le Web"
                                name=keywords>
```

Annotations in the image:

- <----- Requête Au serveur (pointing to the request lines)
- <----- Réponse du serveur (pointing to the response lines)

Le langage XHTML (EXtensible HyperText Markup Language) : C'est un langage à base de balises servant à écrire des **pages Web** mais d'une manière plus rigoureuse que **HTML**, ce qui permet une meilleure **portabilité** des pages Web.

- XHTML est presque identique à HTML mais **plus rigoureux**
- XHTML est fondé sur la syntaxe **XML**
- XHTML est supporté par la plus part des navigateurs Web



- Les balises XHTML doivent être **correctement imbriqués** :

On écrit :

```
<strong><span>Contenu </span></strong>
```

Et non pas :

```
<strong><span> contenu </strong></span>
```

- Les balises XHTML doivent toujours être **fermés**
- Les balises XHTML doivent être en **minuscules**
- Les documents XHTML doivent avoir un balise **racine**



- Le langage **HTML5** est une version plus récente du langage **HTML**. Cette version ajoute plusieurs nouvelles balises notamment : **section** , **article** , **nav** , **aside** , **header** , **footer**. Ils ajoutent aussi plusieurs attributs.
- **HTML5** permet aussi la **validation des formulaires** grâce à la définition de patronnes.



Le langage Javascript : C'est un langage de programmation de scripts qui est exécuté principalement par les navigateurs Web sur les machines clientes. Javascript permet de donner du **dynamisme** aux pages Web, on peut par exemple :

- Faire **bouger** (modifier, supprimer) les éléments d'une page Web
- Mettre à jour les éléments de la page sans la **recharger**
- Réagir aux **événements** produits par l'utilisateur (**Clic**, déplacement souris, presse d'une touche clavier ..etc)
- Utiliser des **menus**, des sélecteurs de date, des **barres de progression** ..etc



Ajax

L'Ajax (**A**synchronous **J**avascript and **X**ml): C'est une technologie qui permet de réduire le temps de latence d'une application Web classique.

Navigateur Web



requête HTTP

données HTML+CSS

Serveur



Sans Ajax

Navigateur Web



requête HTTP

données XML

Serveur



Avec Ajax

Le langage PHP (Personal Home Page ---> Hypertext Preprocessor) : C'est un langage de programmation interprété utilisé pour créer des pages Web dynamiques. C'est parmi les langages de développement Web les plus utilisés actuellement.

- Facile a apprendre et dispose d'une bibliographie importante
- Gratuit et supporté par la majorité des hébergeurs Web
- Dispose d'un multitude de frameworks



JSP (Java Server Pages) : C'est une technologie permettant d'insérer du code java dans un fichier HTML celui-ci prend l'extension « .jsp ». Le code java dans le fichier jsp est interprété par le serveur Web comme vue précédemment avec le langage PHP.

Servlet : C'est une classe java qui permet d'intercepter des requêtes HTTP et retourner du code HTML.

JSP = du **java** dans le **HTML**

Servlet = du **HTML** dans le **java**



Le langage XML (Extensible Markup Language) : C'est un langage qui permet de structurer et d'organiser les documents. **XML** est basé sur les balises comme dans **HTML** mais contrairement a ce dernier il permet d'ajouter a volonté de nouvelles balises.

XML est le **successeur** de HTML sur le web.

Le langage XML :

- Est simple, universelle et extensible
- Permet de séparer le contenu d'un document de la présentation ce qui facilite leur **traitement automatique** (tris, réorganisation, ...etc)
- Ajoute une certaine sémantique au document ce qui facilite l'**interopérabilité**



Langage XML (2)

Exemple (<http://xml.chez.com/initiation/>) :

DU HTML :

<H2>Bibliographie XML</H2>

** Jean-Christophe Bernadac et François Knab, <I>Construire une application XML</I>, Paris, Eyrolles, 1999**

** Alain Michard, <I>XML, Langage et Applications</I>, Paris, Eyrolles, 1998 **

** William J. Pardi, <I>XML en Action</I>, Paris, Microsoft Press, 1999, adapté de l'anglais par James Guerin**



Langage XML (3)

DU XML : (Ajouter de la Sémantique)

```
1 <?xml version="1.0" encoding="ISO-8859-1"?>
2 <BIBLIO SUBJECT="XML">
3 <BOOK ISBN="9782212090819" LANG="fr" SUBJECT="applications">
4 <AUTHOR>
5 <FIRSTNAME>Jean-Christophe</FIRSTNAME>
6 <LASTNAME>Bernadac</LASTNAME>
7 </AUTHOR>
8 <AUTHOR>
9 <FIRSTNAME>François</FIRSTNAME>
10 <LASTNAME>Knab</LASTNAME>
11 </AUTHOR>
12 <TITLE>Construire une application XML</TITLE>
13 <PUBLISHER>
14 <NAME>Eyrolles</NAME>
15 <PLACE>Paris</PLACE>
16 </PUBLISHER>
17 <DATEPUB>1999</DATEPUB>
18 </BOOK>
19 <BOOK ISBN="9782212090529" LANG="fr" SUBJECT="général">
20 <AUTHOR>
21 <FIRSTNAME>Alain</FIRSTNAME>
22 <LASTNAME>Michard</LASTNAME>
23 </AUTHOR>
24 ....
25 ....
26 </BOOK>
27 </BIBLIO>
```

