



Java pour le développement d'applications Web : Java EE

Introduction

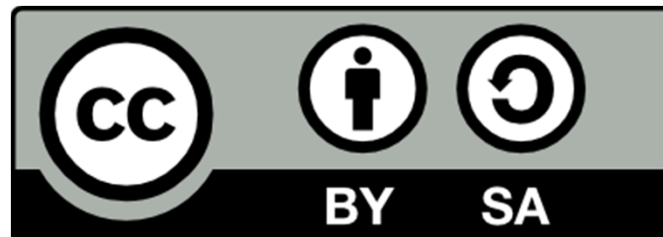
Mickaël BARON - 2007 (Rév. Août 2009)
<mailto:baron.mickael@gmail.com> ou <mailto:baron@ensma.fr>

Creative Commons

Contrat Paternité

Partage des Conditions Initiales à l'Identique

2.0 France



<http://creativecommons.org/licenses/by-sa/2.0/fr>

Le cours

➤ L'objectif de ce cours

- Initiation à la conception d'applications WEB de « qualité »
- Architecture logicielle
- Intérêt spécifique à la partie présentation des applications WEB

➤ Structuration rapide du cours

- Introduction aux technologies WEB et rappels
- Technologies des Servlets, JSP
- Programmation par balises : balises personnalisées
- JSP version 2.0
- Framework : Struts
- Java Server Faces (JSF)

Le cours

PROGRAMME :

Applications Web : notions essentielles

- HTTP : notions fondamentales
- Application Web vs site Web ?
- Fichiers WAR et fichiers EAR
- Navigateurs et serveurs Web

Servlets

- Conteneurs de servlets : architecture et multi-threading
- Écriture de servlets
- Objet HttpRequest - HttpResponse
- Gestion des formulaires
- Forward et inclusion
- Suivi de session
- Binding Listeners
- Filtres

Pages JSP

- Syntaxe des JSP
- Architecture des JSP de type 2
- Séparation Modèle-Vue
- Gestion dynamique des formulaires
- Pages JSP avec JavaBeans
- Scriptlets ou balises de style XML
- Objets implicites
- Inclusions statiques ou dynamiques ?
- Pages d'erreurs

Bibliothèque de Balises

- Utilisation des Tag Libs
- Attributs de temps d'une requête
- Définition de Tag Libs
- Balises simples
- Balises complexes

JSP 2.0 (JSF et JSTL)

- Langage d'expression des JSP
- Java Standard Tag Libraries
- JavaServer Faces

Patterns et bonnes pratiques

- Servlets de contrôle
- Contrôleurs basés sur les commandes
- JavaBeans sous forme de validateurs de formulaires
- Séparation Java-HTML
- Le framework Struts

Architecture et conception

- Architectures des applications Web
- Choix des technologies Web appropriées
- Extensions WAE (Web Application Extensions) à UML
- Accès SGBD et Pool de connexion
- Serveurs d'application, EJB et Java EE

Sécurité sur le Web

- Autorisation - Authentification
- Sécurité déclarative Java EE
- Certificats - SSL

XML dans les applications Web

- Syntaxe XML de base
- DTD et Schémas XML
- Utilisations de XML



Développement objet, composants et Web

CONCEPTION D'APPLICATIONS WEB D'ENTREPRISE AVEC JAVA EE, LES SERVLETS, JSP ET STRUTS JWEB

Si les servlets et les JSP offrent aux développeurs Java la possibilité de créer facilement des pages Web dynamiques, l'association de diverses technologies devient vite déroutante, à mesure que les applications Web gagnent en complexité.

Ce cours intensif vous apportera toutes les compétences nécessaires à la création d'applications Web qui soient à la fois évolutives, sûres et simples à administrer. Une étude de cas est développée tout au long du cours, dont le contenu s'attache avant tout aux questions de logique et de contrôle des applications Web plutôt qu'aux éléments de conception graphique ou de mise en page sous HTML. Vous serez confrontés à des problèmes rencontrés fréquemment dans la conception d'applications Web, et apprendrez à utiliser les Design Patterns Java EE, pour les résoudre.

Ce cours répond aux questions suivantes : "Comment créer une application Web efficace et maintenable ?", "Comment gagner du temps dans les développements Web ?", "Quand utiliser les technologies JSP, JSF, JSTL, EJB, Struts, JavaScript, XML, applets ?".

VOUS ALLEZ APPRENDRE À :

- Construire des interfaces Web à l'aide de JSP, de servlets et de JavaScript
- Écrire des applications Web portables, faciles à administrer, faisant la séparation entre HTML et Java
- Savoir pourquoi passer à JSP 2.0
- Comprendre l'évolution des applications Web vers Struts et JSF
- Décider de l'opportunité d'utiliser telle ou telle technologie Web
- Utiliser efficacement XML et les technologies connexes dans les applications Web
- Invoquer des Enterprise JavaBeans à partir de composants Web

Durée
5 jours - 35 heures

Audience
Développeurs Java expérimentés souhaitant apprendre à utiliser des composants Java EE (servlets et JSP) pour bâtir des applications Web d'entreprise

Pré-requis
Expérience pratique de la programmation avec Java et compréhension de la conception orientée objet ou avoir suivi les cours : JOD ou IJOP (p.14 et p.15)

Méthode pédagogique
50% de travaux pratiques

Prix interentreprises 2006 : 1971 Euros HT

Disponible également en intra-entreprise

Explosion du nombre de formations concernant J2EE (Servlets, JSP, ...)

Déroulement du cours

➤ Pédagogie du cours

- Présentation des concepts
- Illustration avec de nombreux exemples
- Des bulles d'aide tout au long du cours :



Ceci est une alerte



Ceci est une astuce

➤ Pré-requis

- Connaissance de Java
- Connaissance des technologies WEB (HTML, HTTP)

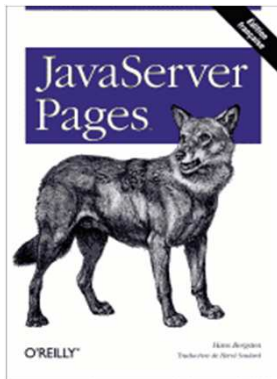
➤ Mise en place du cours

- Livres : **Servlets Java Guide du programmeur** 2^{ème} édition - Jason Hunter et William Crawford - O'Reilly et **JavaServer Pages** - Hans Bergsten - O'Reilly
- Internet : *www.developpez.com*, ...

Bibliothèque



- **Servlets Java : Guide du programmeur**
 - Auteur : Jason Hunter et William Crawford
 - Éditeur : O'Reilly
 - Edition : SE avril 2001 - 715 pages - ISBN : 2841771962



- **JavaServer Pages**
 - Auteur : Hans Bergsten
 - Éditeur : O'Reilly
 - Edition : 2001 - 528 pages - ISBN : 2841771458



- **J2EE Développement d'application Web**
 - Auteur : Benjamin Aumaille
 - Éditeur : ENI
 - Edition : avril 2002 - ISBN : 2746016567

Organisation du cours...

- Partie 1 : Introduction aux technologies WEB
- Partie 2 : Servlets
- Partie 3 : installation d'un serveur d'application
- Partie 4 : JSP
- Partie 5 : Balises personnalisées
- Partie 6 : JSP 2.0
- Partie 8 : Struts
- Partie 7 : Java Server Faces

Powered by
Struts

Organisation du cours...

- Partie 1 : Introduction aux technologies WEB
 - Protocole HTTP
 - Architectures WEB
 - Technologie Java EE
- Partie 2 : Servlets
 - Servlets et API
 - Traitement des données de formulaires
 - Architecture de développement
 - Cycle de vie
 - Suivi de session
 - Collaboration de Servlets
 - Sécurité : authentification
 - Accès aux BD avec JDBC

Organisation du cours...

- Partie 3 : Installation d'un serveur d'application
 - Installation et configuration du serveur d'application Jakarta Tomcat
 - Déploiement des applications WEB
- Partie 4 : JSP
 - JSP et définition
 - Tag (directive, commentaire, déclaration, scriptlet, expression)
 - Objets implicites, cycle de vie et technique de gestion des erreurs
 - Java Beans
 - Collaboration de JSP/Servlets et vers une architecture MVC...
- Partie 5 : Balises personnalisées
 - Définition
 - Conception
 - Déploiement

Organisation du cours...

➤ Partie 6 : JSP 2.0

- Balises personnalisées (2.0)
- Expression Language (EL)
- Java Standard Tag Libraries (JSTL)

➤ Partie 8 : Struts

- Principe du framework
- Éléments de base : *struts-config.xml*, *ActionForm* et *Action*
- Éléments complémentaire : *DynaActionForm* et plug-in *Validator*
- Balises personnalisées : Localisation et ...

➤ Partie 9 : Java Server Faces

- Bean Managé
- Navigation
- Validators, Converters
- Tomahawk, ...



Java pour le développement d'applications Web : Java EE

Introduction aux technologies WEB

Mickaël BARON - 2007 (Rév. Août 2009)
<mailto:baron.mickael@gmail.com> ou <mailto:baron@ensma.fr>


Architectures WEB : applications monolithiques

➤ Caractéristiques

- Applications mêlant présentation, règles métier et les données
- Ne communique pas avec l'extérieur
- Terminaux passifs

➤ Avantages

- Performance
- Sécurité



On parle de logique métier ou de règles métier relative aux fonctionnalités de l'application (Banque : calcul des intérêts d'un compte)

➤ Inconvénients

- Maintenance logicielle
- Ouverture vers d'autres systèmes
- Technologies réseaux propriétaires

Architectures WEB : applications à deux niveaux

➤ Caractéristiques

- Généralement appelées architecture client/serveur
- Un SGBD pour les données et une application pour l'IHM et le contrôle
- Seules les données transitent par le réseau
- Exemples de technologie : Java/Swing avec SGBD mySQL

➤ Avantages

- Réparties la puissance machine sur les clients
- Mise en œuvre du modèle de bases de données relationnelles
- Intégration inter-systèmes au niveau des données possibles

➤ Inconvénients

- Déploiement
- La logique métier est répartie sur les deux composantes
- Maintenance gestion des versions

Architectures WEB : architectures multi-tiers (couches)

➤ Caractéristiques

- Au moins trois niveaux : IHM, les règles métiers et la persistance
- Des normes de communication entre eux
- Exemples de technologie : php/mySQL (3 niveaux au plus), Java EE (Java Enterprise Edition)



Navigateur

Technologies :
JSP / Servlets

Présentation

Technologies :
JavaBean, EJB

Logique métier

Technologies :
JDBC, JMS

Middleware

Technologies :
SGBDR, SGBDO

Persistance

➤ Avantages

- Maintenance
- Nécessite peu de puissance client en cas de clients légers

➤ Inconvénients

- Serveur puissant pour la logique métier

Différents types d'application : client ...

➤ Définition de « client »

- Logiciel médiateur entre l'utilisateur et le service
- Exemples : FTP, messagerie (mailer), navigateur (browser), webmail, jeux, ...

➤ Différentes catégories de client

- **Lourd** : le service est disponible sur le poste client avec possibilité de connexion à des serveurs (appelée aussi application à architecture client/serveur)
 - Exemples : Yahoo Messenger, Word, Money, Battlefield 2, ...
- **Léger** : tout le service est disponible sur des serveurs et l'utilisateur y accède par un conteneur spécialisé (appelée aussi application à architecture multi-tiers)
 - Exemples : Google, Yahoo Mail, ...

Différents types d'application : client lourd

- Technologies : Java/Swing, C#/.NET, Tcl/Tk, C++/QT
- Avantages
 - Interfaces utilisateurs riches (WIMP, POST-WIMP)
- Inconvénients
 - Déploiement (utilisation de CD, téléchargement/installation)
 - Gestion des versions (patch, problème de compatibilité)

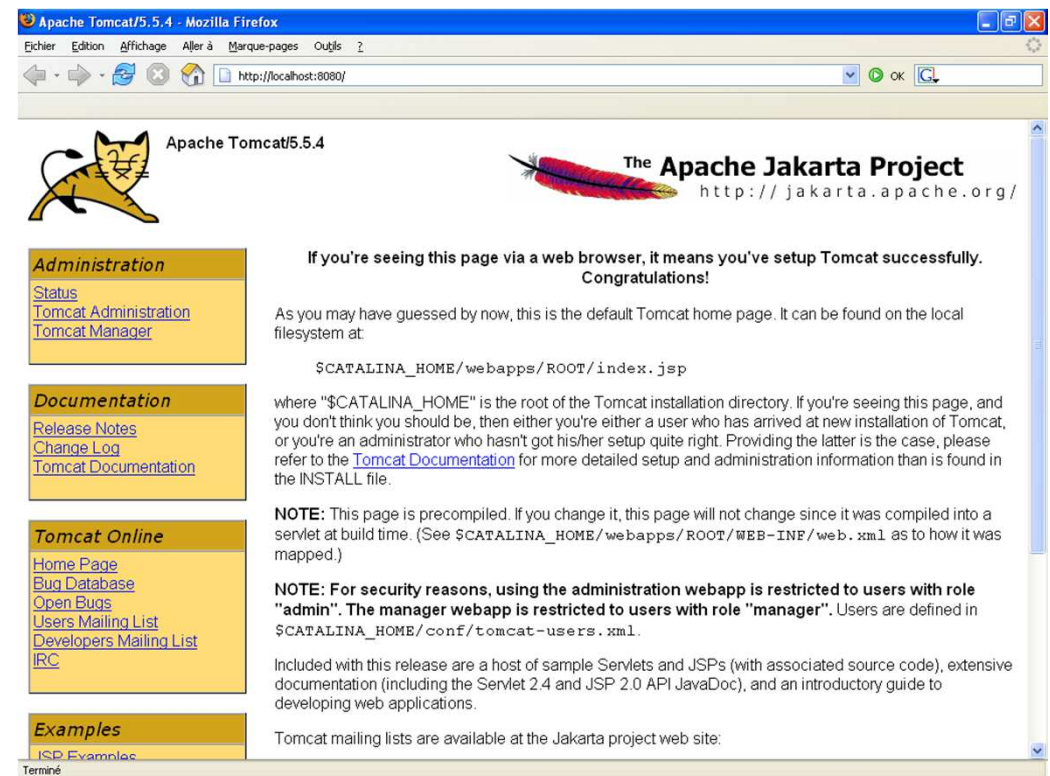


Interfaces riches (possibilité de dessiner dans un canvas, notion hiérarchique de composants, ...)

Interactions évoluées (Drag&Drop, popup menu, ...)

Différents types d'application : client léger

- Architecture dite « multi-tiers » de trois à n-niveaux
- Les technologies pour la génération et le traitement de l'IHM sont présentes dans le client et dans le serveur
- Technologies côté client
 - HTML, DHTML, JavaScript, ...
- Technologies côté serveur
 - PHP, ASP, JSP, ...
- Avantages
 - Maintenance
 - Accessibilité
- Inconvénients
 - Interfaces utilisateurs pauvres et proches du classique formulaires



Client léger : Internet et HTTP

- Les clients légers désignent essentiellement toutes les applications associées aux sites Web
- L'accès aux services se fait par l'intermédiaire d'un conteneur spécialisé qui est généralement un navigateur
 - FireFox, Internet Explorer
- Les technologies pour le transport entre le serveur et le client sont
 - Internet
 - HTTP, HTTPS
- Les sites Web actuels se caractérisent par le fameux mode page par page
 - A chaque requête de l'utilisateur (demande) le serveur génère une nouvelle présentation
 - L'affichage n'est pas continu

Client léger : l'Internet à l'origine

- A l'origine Internet a vocation de diffuser de l'information statique

- HTTP (déconnecté)
- HTML (langage de description de document)

Le premier navigateur pour Internet : Mosaic

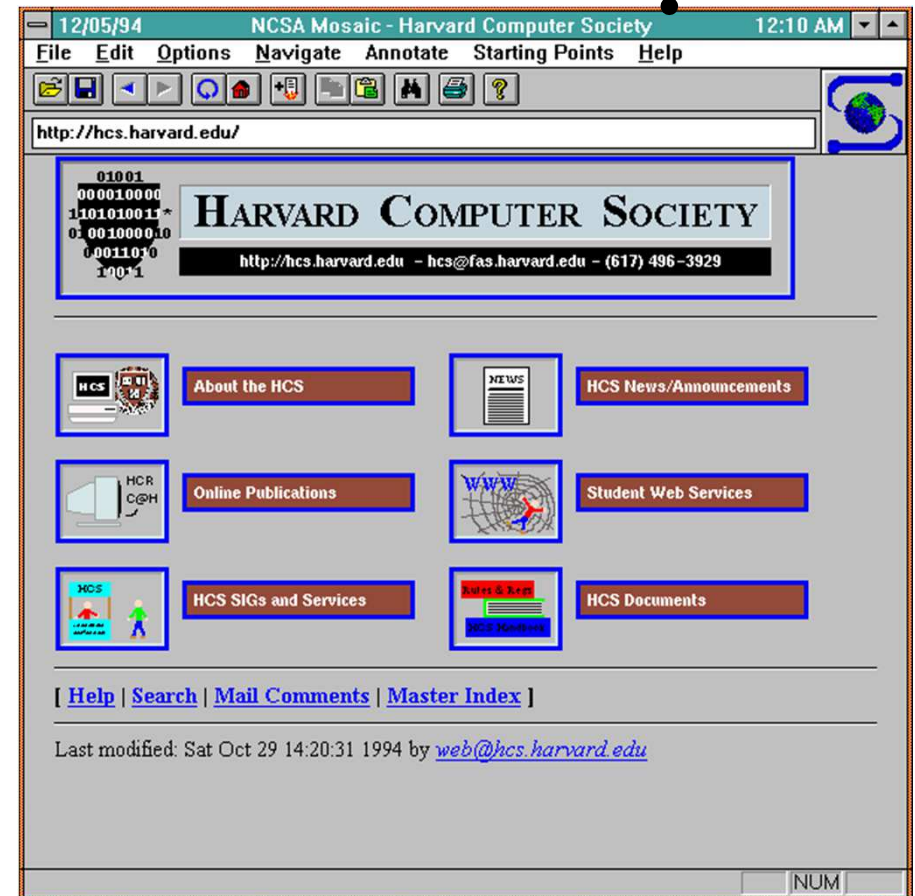
- Déploiement

- Universel (protocoles standards et réseau standard)

- Un navigateur suffit

- Pas prévu pour embarquer des applications

- Afficher des données en temps réel



Client léger : Internet et HTTP

➤ Application Web et page par page

Mobilisation des IUP:

o Patrick Girard, IUP GPhy, Poitiers. o Geneviève Jomier, IUP GMI, Dauphine.
o Jean-Noël Hallet, IUP Chim-Bio, Nantes. o France Hutzinger, IUP Manag., ...
mobilisationiup.net/CR_AG.htm - 51k - [En cache](#) - [Pages similaires](#)

Bonjour, J'ai le plaisir de vous inviter à la soutenance de mon ...

... av du Recteur Pineau 86022 POITIERS Cédex <mailto:girard@gphy.campus.univ-poitiers.fr>
<http://www-gphy.univ-poitiers.fr> Tél : (33) (0)5 49 45 36 32 - Fax ...
www.afihm.org/theses/annonces/PatrickGirard-HDR.txt - 16k - [En cache](#) - [Pages similaires](#)

Alice Recherche - Nomade.fr - Education, formation

Education - Université de Poitiers - Poitiers, France - Tous Publics Adresse :
<http://gphy.campus.univ-poitiers.fr/plaquette/> ...
nomade.aliceadsl.fr/cat/education_formation/enseignement_superie/universites/france/poitiers/ - 25k - [En cache](#) - [Pages similaires](#)

Cyclisme et Musculation : Liste des messages postés dans le forum ...

Vous pourrez répondre à ce court questionnaire à l'adresse suivante :
<http://gphy.campus.univ-poitiers.fr/omega3/> Nous vous remercions de l'attention que ...
ldran.free.fr/cyclisme/forum/read.php?id=264 - 13k - [En cache](#) - [Pages similaires](#)

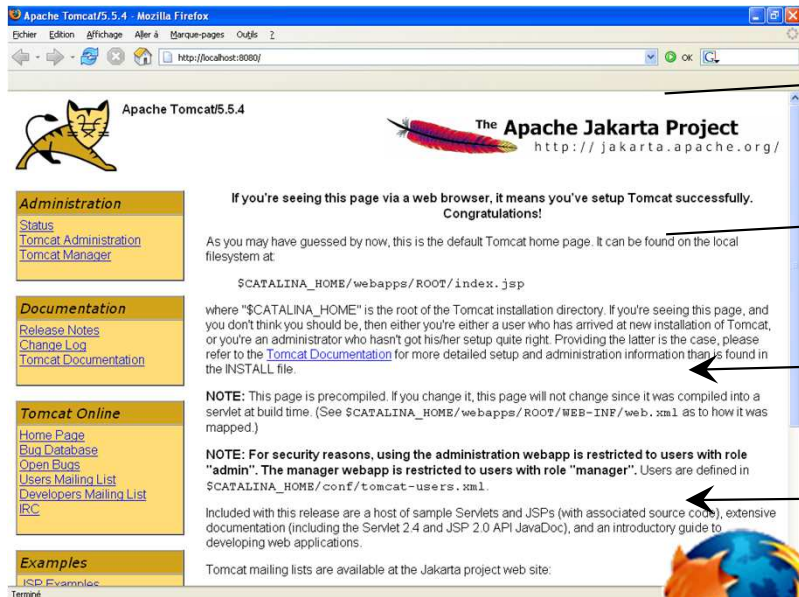


On assiste actuellement à l'émergence de nouvelles technologies permettant d'éviter le mode page par page introduit par HTTP et Internet

Mode page par page à chaque requête un réaffichage permanent de la présentation

Protocole HTTP

- Hyper Text Transfer Protocol v1.1
- Protocole Client/Serveur sans état
 - Impossibilité de conserver des informations issu du client
- La conversation HTTP est initialisée lorsque l'URL est saisie dans le navigateur



Client WEB

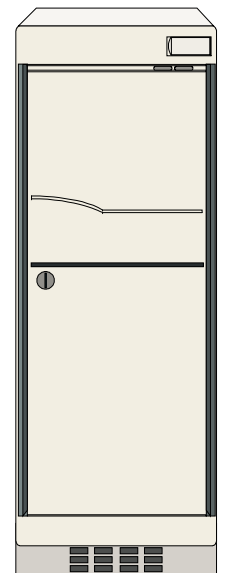


1 - le client ouvre la connexion avec le serveur

2 - le client émet une requête HTTP

3 - le serveur répond au client

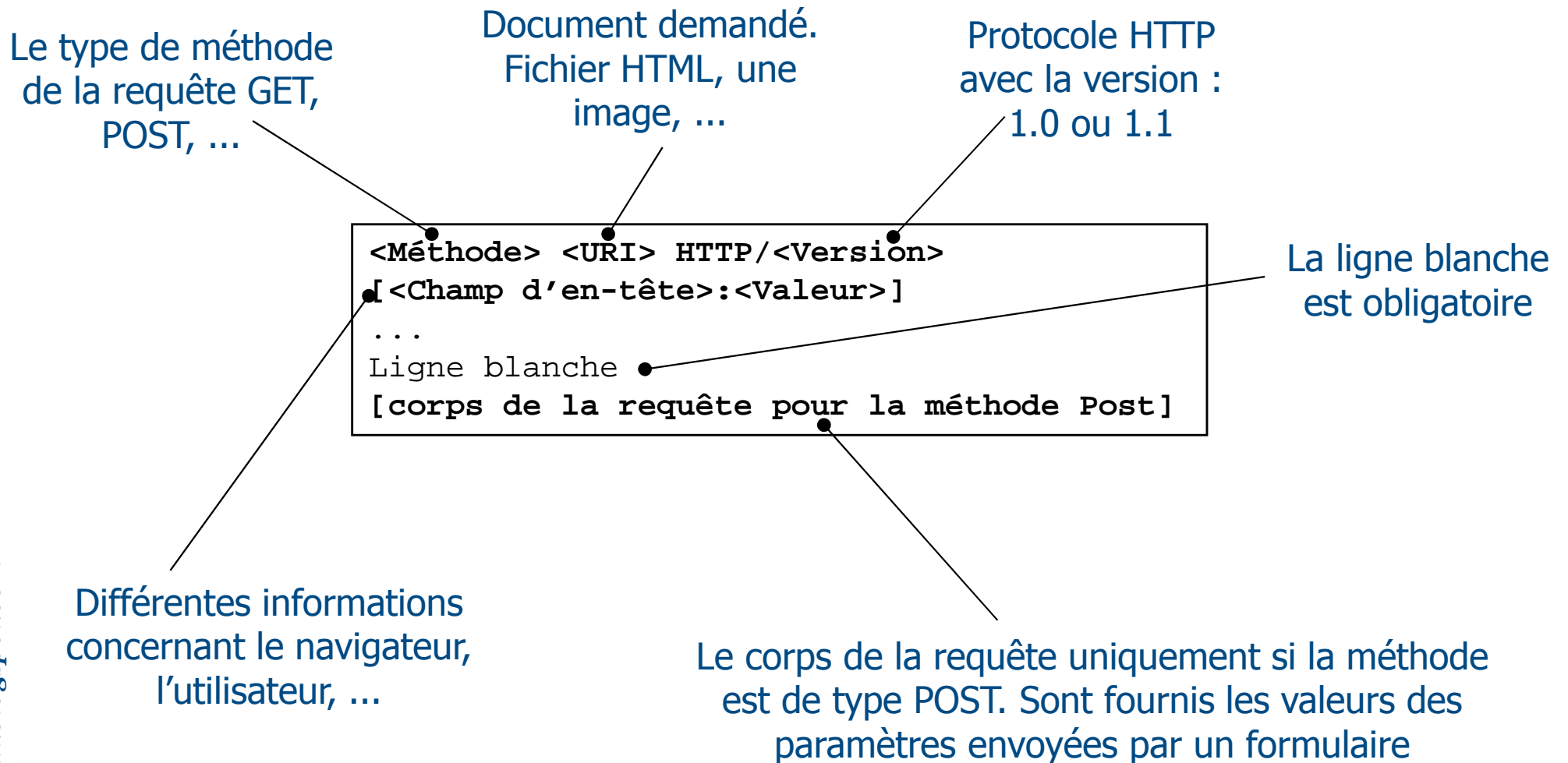
4 - la connexion est fermée



Serveur Web

Protocole HTTP : requête

➤ Requête envoyée par le client (navigateur) au serveur WWW



Protocole HTTP : en-têtes de requête

- Correspond aux formats de documents et aux paramètres pour le serveur
 - *Accept* = type MIME visualisable par le client (text/html, text/plain, ...)
 - *Accept-Encoding* = codage acceptées (compress, x-gzip, x-zip)
 - *Accept-Charset* = jeu de caractères préféré du client
 - *Accept-Language* = liste de langues (fr, en, de, ...)
 - *Authorization* = type d'autorisation
 - BASIC nom:mot de passe (en base64)
 - Transmis en clair, facile à décrypter
 - *Cookie* = cookie retourné
 - *From* = adresse email de l'utilisateur
 - ...



**Nous reviendrons sur
l'en-tête *authorization*
dans la partie suivante
au niveau de la sécurité**

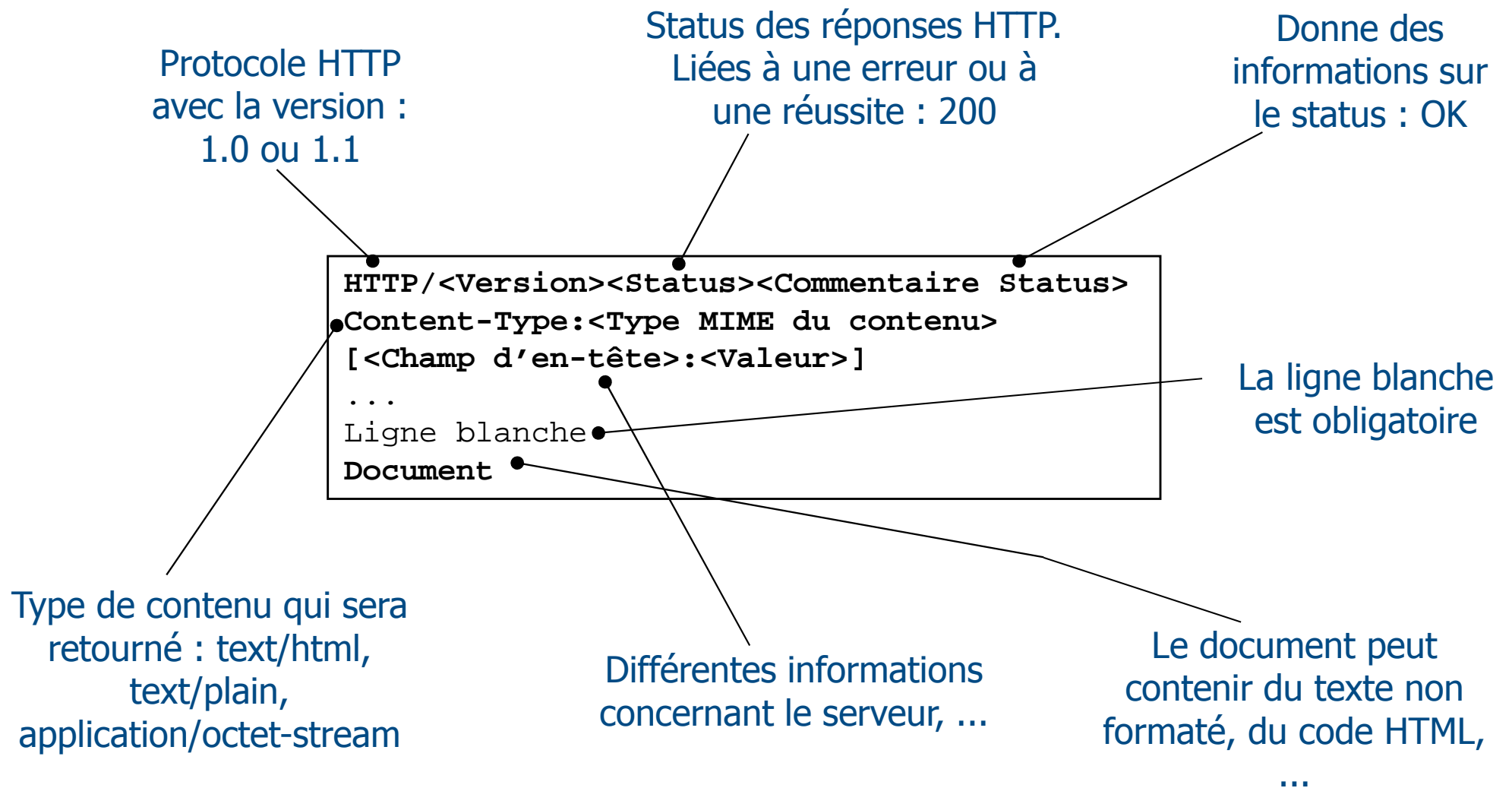
Protocole HTTP : type de méthodes

- Lorsqu'un client se connecte à un serveur et envoie une requête, cette requête peut-être de plusieurs types, appelés **méthodes**
- Requête de type GET
 - Pour extraire des informations (document, graphique, ...)
 - Intègre les données de formatage à l'URL (chaîne d'interrogation)
 - *www.exemple.com/hello?key1=titi&key2=raoul&...*
- Requête de type POST
 - Pour poster des informations secrètes, des données graphiques, ...
 - Transmis dans le corps de la requête

```
<Méthode> <URI> HTTP/<Version>
[<Champ d'en-tête>:<Valeur>]
...
Ligne blanche
[corps de la requête pour la méthode Post]
```

Protocole HTTP : réponse

➤ Réponse envoyée par le serveur WWW au client (navigateur)



Protocole HTTP : en-têtes de réponse

- Correspond aux informations concernant le serveur WWW
 - *Accept-Range* = accepte ou refus d'une requête par intervalle
 - *Age* = ancienneté du document en secondes
 - *Set-Cookie* = créé ou modifie un cookie sur le client
 - *WWW-Authenticate* = système d'authentification. Utiliser en couple avec l'en-tête requête *Authorization*

➤ ...

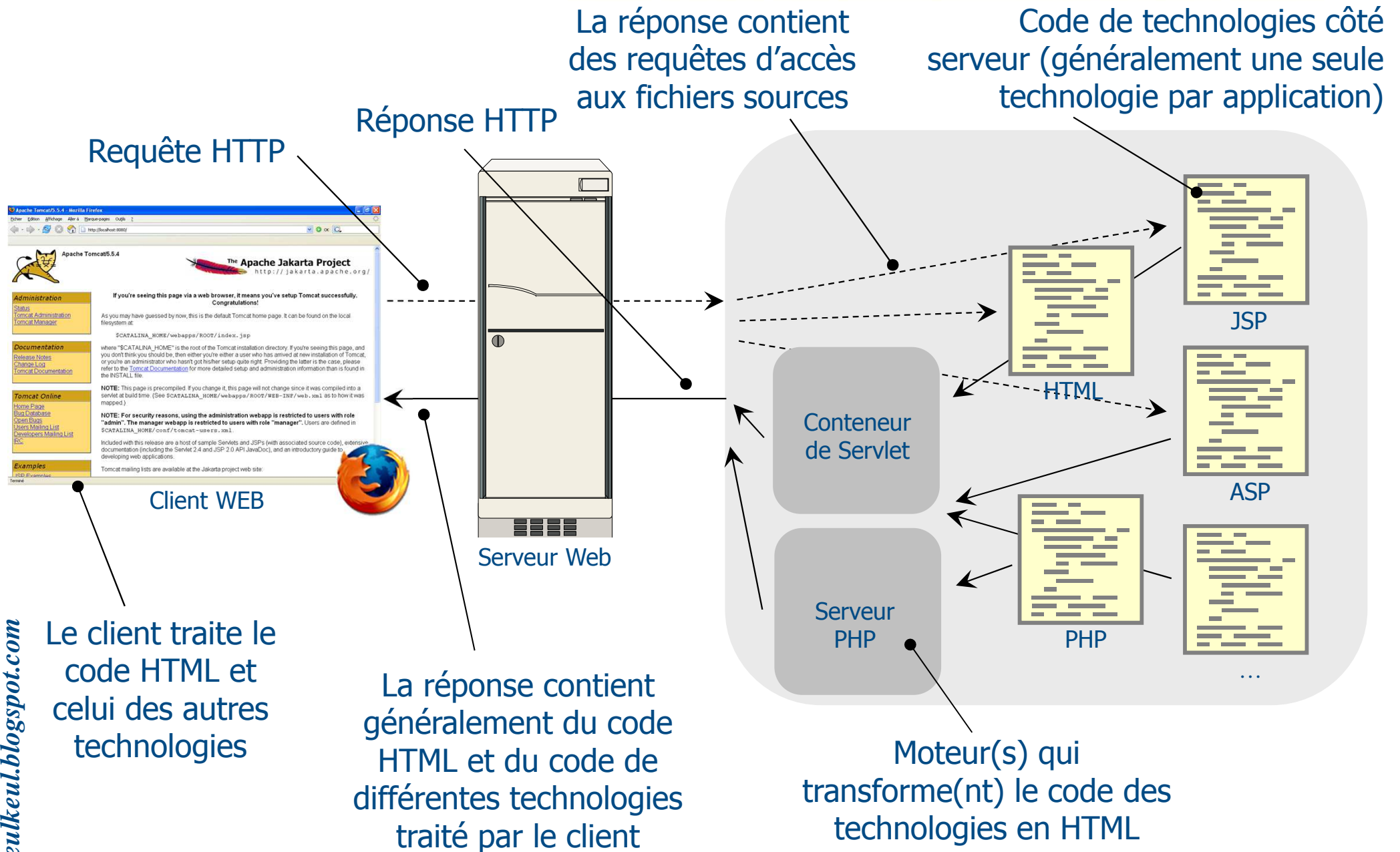


Nous reviendrons sur l'en-tête *WWW-Authenticate* dans la partie suivante au niveau de la sécurité

Protocole HTTP : status des réponses

- Réponse du serveur au client *<Status><Commentaire>*
 - *100-199 : Informationnel*
 - 100 : Continue (le client peut envoyer la suite de la requête), ...
 - *200-299 : Succès de la requête client*
 - 200 : OK, 204 : No Content (pas de nouveau corps de réponse)
 - *300-399 : Re-direction de la requête client*
 - 301 : Redirection, 302 : Moved Temporarily
 - *400-499 : Erreur client*
 - 401 : Unauthorized, 404 : Not Found (ressource non trouvée)
 - *500-599 : Erreur serveur*
 - 503 : Service Unavailable (serveur est indisponible)

Client léger : principe générale



Client léger : technologies côté serveur

- Les technologies côté serveur permettent à l'aide de langages spécialisés de générer plus ou moins du code HTML
- Nous distinguons deux types de langages
 - Langages à balises : ceux qui sont utilisés dans le code HTML
 - PHP, ASP, JSP et .NET
 - Langages de contrôle : ceux qui ne contiennent que du code propre au langage et qui généralement s'occupe du contrôle de l'application
 - CGI et Servlet
- Les langages de contrôles sont adaptés au traitement de fonctionnalités (sécurité, base de données, ...) et délèguent la partie présentation aux langages à balises
- Il n'est pas rare de trouver des applications uniquement avec des langages à balises mais la compréhension du code en devient alors difficile

Client léger : technologies côté serveur

- Deux types de langage, deux types de sémantique
 - Langages « procéduraux » : la portée des variables est limitée et l'absence de persistance oblige à bidouiller pour maintenir la valeur d'une variable pour chaque requête
 - ASP, PHP et CGI
 - Langages à objets : la persistance des objets permet de maintenir des états (valeurs d'attributs) à chaque requête
 - JSP, Servlet et .NET
- Exemple : **un compteur**
 - Dans le cas des langages procéduraux pour stocker la valeur d'un compteur on peut soit utiliser un simple fichier ou soit utiliser une base de donnée. A chaque nouvelle requête le compteur est initialisé au travers du support
 - Dans le cas des langages à objets, un objet contenant un attribut compteur est créé à la première requête et sa durée de vie est fonction de différents paramètres (serveur, scope, ...)

Client léger : technologies côté client

- Les technologies côté client permettent d'effectuer des traitements supplémentaires que ceux fournis uniquement par l'HTML
- Deux types de technologie sont à distinguer
 - Affichage : celles qui s'occupent de la partie IHM
 - HTML et DHTML
 - Dynamique : celles qui effectue des traitements dynamiques
 - JavaScript
- Quelle que soit la technologie utilisée elles devront être codée et transmises par les technologies côté serveur
- Exemple : un formulaire
 - L'HTML ou le DHTML permettent d'afficher le formulaire
 - Le JavaScript permet de vérifier la cohérence « de surface » des données (champs vides, ...)

Client léger : les solutions envisagées

- Quelle que soit la complexité du site Web les technologies côté clients sont toujours identiques
- Au contraire le choix de la technologie côté serveur dépend fortement de la complexité de l'application Web
 - Site marketing et recherche de simples informations : PHP ou ASP
 - Site commercial avec transaction: langage à objets Java EE ou .NET
- Avis personnels
 - L'utilisation de technologies à objets permettent d'imposer une architecture
 - Les technologies à objets offrent un nombre important d'API
 - La persistance liée au paradigme objets permet de gérer plus facilement les sessions utilisateurs et le stockage d'attributs (compteur)
- Exemple : Java EE

Technologie Java EE : acronymes en puissance

➤ Les API Java EE

- Le serveur Java EE va fournir à une application WEB un ensemble de services comme les connexions aux bases de données, la messagerie, les transactions, ...

➤ La spécification Java EE prévoit un ensemble d'extensions Java standard que chaque plate-forme doit prendre en charge

- **Servlet** : composant coté serveur, dont le rôle est de fournir une implémentation au traitement des requêtes/réponses
- **JSP** : JavaServer Pages est une extension au Servlet permettant de simplifier la génération de pages web dynamiques
- **JNDI** : Java Naming and Directory Interface
- **JDBC** : Java Database Connectivity est une API permettant de se connecter à une base SQL
- **JMS** : Java Messaging Service
- **JTA** : Java Transaction API
- **EJB** : Entreprise Java Bean