Systèmes d'Information Avancés (et répartis)

Université Lyon 1
MIAGE

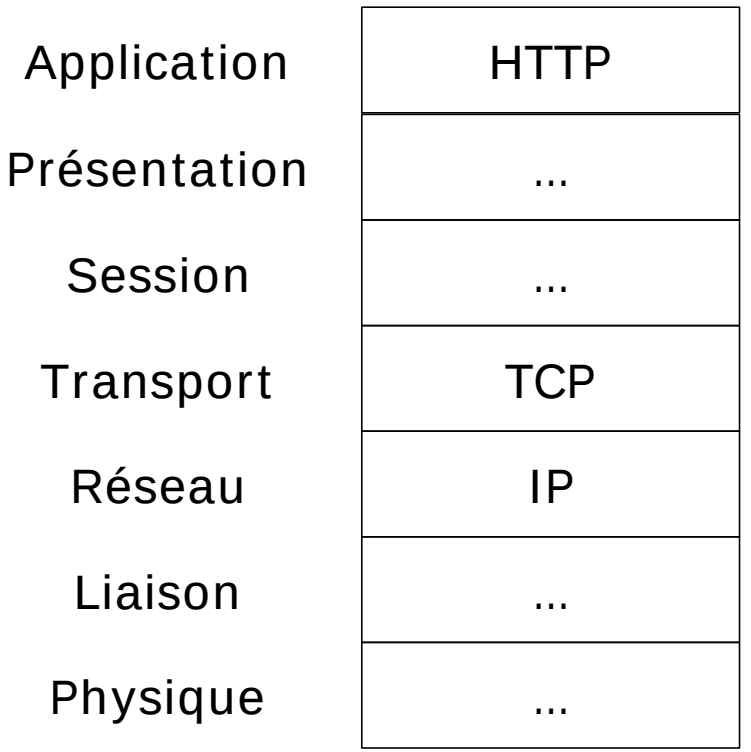
L. Médini, février 2005

Plan des cours

- Protocole HTTP et programmation serveur
  - Les différents types de requêtes
  - Le passage des arguments et des données
  - Composition de formulaires web
  - Exemples de code Java
- Architectures réparties
- Objets distribués (Javabeans, EJB)
- Web services (SOA, WSDL, SOAP, UDDI)
- Projet

Le protocole HTTP

- Place de http dans le modèle OSI



Le protocole HTTP

- Orienté connexion
- Sans état
  - chaque page WEB est transmise dans une connexion séparée (sauf pour les connections persistantes)
- RFC

  http://www.w3.org/Protocols/rfc2616/rfc2616.html
- Requêtes/réponses

Le protocole HTTP

- Format d'une requête

Méthode URL Version<crlf>
En-tête : Valeur<crlf>
.
.
En-tête : Valeur<crlf>
Ligne vide<crlf>
Corps de la requête

GET http://www.monsite.com HTTP/1.0
Accept : text/html
If-Modified-Since : Saturday, 15-January-2000 14:37:11 GMT
User-Agent : Mozilla/4.0 (compatible; MSIE 5.0; Windows 95)

Le protocole HTTP

Méthodes HTTP

GET : permet d'obtenir des informations (document, graphique, le résultat d'une requête dans une base de données, …) peut envoyer des informations dans l'URL limités en taille à 240 caractères sur certains serveurs (données d'un formulaire simple)

POST : permet de poster des informations (une données graphique, des données pour une base de données…) en quantité illimitée

HEAD : demande les en-têtes de la réponse ; seulement pour récupérer la taille des documents, l'heure de modification…

PUT et **DELETE** : agissent sur les documents du serveur (peu utilisées à cause des problèmes de sécurité)

TRACE : retourne le contenu exact de la demande (utilisée pour le débogage)

OPTIONS : pour demander au serveur les méthodes et les options qu'il supporte

Le protocole HTTP

- Champs d'en-tête

  Accept : type MIME qu'un client sait gérer (application/pdf, image/jpeg, …)

  Accept-Charset : jeu de caractères (ISO-8859-1, …)

  Accept-Encoding : type de compression qui peut être décodée (gzip compress, …)

  Accept-Langage : fr par exemple

  Authorization : accéder à des pages WEB protégés par mot de pass

  Cache-Control :

  Connection : gestion des connexions http persistantes pour récupérer un fichier HTML et plusieurs pages associées en une seule connexion de socket (option par défaut)

  Content-Length : taille des données d'une requête POST en octets (voir **getContentLength**)

  Content-type : utilisée par un client quand un document est attaché à POST

  Cookie : pour renvoyer au serveur un cookie qu'il a envoyé

Le protocole HTTP

- Champs d'en-tête

  Expect : demande d'acquittement au serveur d'un document attaché

  From : adresse e-mail de la personne responsable de la requête (envoyée par les robots de recherche mais pas par les navigateurs)

  Host : nom d'hôte en numéro de port du client

  If-Match et **If-None-Match**

  If-Modified-Since : la page sera transmise au client uniquement si elle a été modifiée après une date spécifiée ; elle pourra ainsi être mise en mémoire cache par le navigateur (voir **getLastModified**)

  If-Unmodified-Since : ...

  Pragma : un servlet servant de serveur proxy doit transmettre la requête même s'il dispose d'une copie locale

  Proxy-Authorization : authentifie un client auprès d'un serveur proxy

Le protocole HTTP

- Champs d'en-tête

  Referer : indique l'URL de la page WEB à partir de laquelle la requête a été émise (peut être facilement maquillée)

  Upgrade : spécifie un protocole autre que HTTP 1.1

  User-Agent : identifie le navigateur ou le client

  Via : défini par les passerelles et les serveurs proxy par lesquels la requête passe

  ...

Le protocole HTTP

Format d'une réponse

Version-HTTP Code Message
En-Tête : Valeur
.
En-Tête : Valeur
Ligne vide
Corps de la réponse

```
HTTP/1.0 200 OK
Date : Sat, 15 Jan 2000 14:37:12 GMT
Content-Type : text/plain
Last-Modified : Fri, 14 Jan 2000 08:25:13 GM

<html>
<head>
<title>La page qui dit bonjour</title>
</head>
<body>
<p>Coucou</p>
</body>
</html>
```

Le protocole HTTP

Codes d’état

Les codes d’état sont classés en 5 catégories

- 100 à 199 : le client doit répondre avec une autre action
- 200 à 299 : la requête à réussi
- 300 à 399 : fichiers déplacés ayant une en-tête Location précisant leurs nouvelles adresses
- 400 à 499 : erreur au niveau du client
- 500 à 599 : erreur au niveau du serveur

Le protocole HTTP

- Codes d'état
  - Exemples

    200 (OK) : code envoyé par défaut

    204 (No Content) : document non modifié ; demande au navigateur de continuer d'afficher le document précédent car aucun nouveau document n'est disponible

    302 (Found) : le document demandé se trouve ailleurs ; possibilité de rediriger automatiquement le client

    400 (Bad Request) : erreur de syntaxe dans la requête

    403 (Forbidden) : l'accès à la ressource est interdit

    404 (Not Found) : aucune ressource n'est présente à l'adresse demandée par le client

    500 (Internal Server Error) : souvent dues à des erreur de programmation côté serveur…

Le protocole HTTP

- En-têtes

  Content-Length : nombre d'octets de la réponse

  Content-Encoding : Type de codage du corps de la réponse

  Content-Language : Type de langage du corps de la réponse

  Content-Length : Longueur du corps de la réponse (uniquement pour les connexions persistantes)

  Content-Type : Type de contenu du corps de la réponse (type MIME) ; exemples :

  application/msword ; application/octet-stream (données binaires ou inconnues) ; application/pdf ; application/x-gzi

  application/x-java-serialized-object

  multipart/form-data

  text/html ; text/plain (text brut)

  image/jpeg ; video/mpeg

Le protocole HTTP

- En-têtes

  Date : Date de début de transfert des données

  Expires : Date limite de consommation des données

  Forwarded : Utilisé par les machines intermédiaires entre le browser et le serveur

  Location : Redirection vers une nouvelle URL associée au document

  Server : Caractéristiques du serveur ayant envoyé la réponse

Le protocole HTTP

- Le suivi de session
  - HTTP est un protocole sans état : déconnexion TCP à chaque requête/réponse
  - Les sessions sont simulées par le serveur en envoyant un identifiant qui est renvoyé par le client à chaque requête
  - 3 solutions de suivi
    - Champ input HIDDEN qui contient l'identifiant de la session (formulaires HTML)
    - Ré-écriture d 'URL : l'identifiant est rajouté à la fin de chaque URL (méthode GET)
    - Cookies : information positionnée par le serveur sur le client puis renvoyée par le client à chaque requête et dont la durée de vie dépasse la session
  - Objets HttpSession (Java) ou Session (ASP)

Formulaires HTML

- Principe
  - Un formulaire est envoyé par un serveur web. Il spécifie
    - Le type de méthode HTTP
    - L'URL de réponse
    - Le contenu du formulaire (éléments <*input*> à remplir)
    - Les noms/identifiants de ces éléments
  - Le client le reçoit et l'affiche ; l'utilisateur le remplit et le *soumet*
  - Le serveur reçoit une nouvelle requête contenant les noms des champs et leurs contenus et la traite

Formulaires HTML

- Exemple de syntaxe d'un formulaire HTML

  (librement adapté de www.google.fr)

```
<form name="gs" method="GET" action="/search">
  <input type="hidden" name="hl" value="fr">
  <input type="text" name="q" size="41" maxlength="2048"
    value="HTTPRequest servlet docs/api">
  <input type="submit" name="btnG" value="Rechercher">
  <input id="stw" type="radio" name="lr" value=""
    checked="checked">
  <label for="stw">Rechercher sur le Web</label>
  <input id="lrt" type="radio" name="lr" value="lang_fr" >
  <label for="lrt">Rechercher les pages en français</label>
</form>
```

Formulaires HTML

- Les différents types de champs de saisie
  - ...
- Les méthodes de récupération des données côté serveur
  - String getParameter(String name)
  - String [] getParameterValues(String name)
  - Enumeration getParameterNames()

Programmation côté serveur

- Applications web
  - Programmes fonctionnant côté serveur
  - Dépendent de l'infrastructure web choisies
  - Exemples :
    - Microsoft™ .Net : VB, C#, IIS, Visual Studio…
    - Java : servlets, JSP, Tomcat, J2EE…
    - Python : Python scripts, scriptlets, DTML, Zope, Plone…
    - CGI…
    - Php…

Programmation côté serveur

- Les outils de programmation Java serveur
  - Tomcat
    - Contenu
      - Serveur web (Apache)
      - Moteur de servlets/JSP (Catalina)
    - Avantages
      - Installation/configuration simple
      - « XML compliant »
      - Standard (fourni avec l'EDI NetBeans)
      - Extensible (sert de base à J2EE)
      - Gratuit…
    - Packages Java
      - javax.servlet
      - javax.servlet.http
      - javax.servlet.jsp
      - javax.servlet.jsp.tagext

Programmation côté serveur

- Les outils de programmation Java serveur
  - Les servlets
    - Définition (http://java.sun.com/products/servlet/whitepaper.html)
      - Servlets are protocol- and platform-independent server side components, written in Java, which dynamically extend Java enabled servers.
      - They provide a general framework for services built using the request-response paradigm.
      - Their initial use is to provide secure web-based access to data which is presented using HTML web pages, interactively viewing or modifying that data using dynamic web page generation techniques.
      - Since servlets run inside servers, they do not need a graphical user interface.

Programmation côté serveur

- Les outils de programmation Java serveur
  - Les servlets
    - Concrètement
      - Objets (classes) Java
        - Composants d'application (« petits » programmes)
        - Derrière un serveur (Web, mais pas seulement)
      - Dans un « Container »
        - Pas d'accès direct au serveur
        - Accès protégé aux autres objets métier de l'applicatior
      - Encapsulation standardisée des données
        - Requêtes
        - Réponses
        - Session
        - Application web (« contexte »)

Programmation côté serveur

- Les outils de programmation Java serveur
  - Les servlets
    - Exemple de code

```
import javax.servlet.*;
import javax.servlet.http.*;

public class NewServlet extends HttpServlet {

  public void init(ServletConfig config) throws ServletException {
       super.init(config); … }
  public void destroy() { … }

  protected void doGet(HttpServletRequest request, HttpServletResponse response
    throws ServletException, IOException {
      response.setContentType("text/html");
     PrintWriter out = response.getWriter();
     out.println("<html><head><title>Hello World!</title></head>");
     out.println("<body><h1>Hello World!</h1></body></html>");
  }

  protected void doPost(HttpServletRequest request, HttpServletResponse
    response) throws ServletException, IOException { … }
}
```

Programmation côté serveur

- Les outils de programmation Java serveur
  - Les Java Server Pages (JSP)
    - Les JSP permettent de
      - « Mélanger » du code HTML statique et du code dynamiquement généré par le serveur
      - Écrire une page HTML comme si elle était statique, et de n mettre du code que quand nécessaire (scripts)
      - Accéder aux même données/objets qu'une servlet
      - Inclure une autre JSP/servlet
      - Définir des balises personnalisées
    - Remarque : une JSP est compilée en servlet à la première utilisation

Programmation côté serveur

- Les outils de programmation Java serveur
  - Les Java Server Pages (JSP)
    - Syntaxe des balises de scripts : **< %** code **% >**
      - Trois types de scripts
        - Des expressions, qui sont évaluées et insérées

        < % = *new java.util.date()* **% >**

        <jsp:expression>

        new java.util.date()

        </jsp:expression>

Programmation côté serveur

- Les outils de programmation Java serveur
  - Les Java Server Pages (JSP)
    - Syntaxe des balises de scripts : **<%** code **%>**
      - Trois types de scripts
        - Scriptlets : morceaux de code Java dans la page

        <% *response.setContentType("text/plain");* **%>**

        <jsp:scriptlet>

        response.setContentType("text/plain");

        </jsp:scriptlet>

Programmation côté serveur

- Les outils de programmation Java serveur
  - Les Java Server Pages (JSP)
    - Syntaxe des balises de scripts : **< %** code **% >**
      - Trois types de scripts
        - Déclarations : permettent de définir des méthodes ou des champs qui seront insérés dans le corps de la servlet

        < %! *private int VariableGlobale = 0;* **% >**

        <jsp:declaration>

        private int VariableGlobale = 0;

        </jsp:declaration>

Programmation côté serveur

- Les outils de programmation Java serveur
  - Les Java Server Pages (JSP)
    - Variables prédéfinies dans les scripts
      - **request**
      - **response**
      - **out**
      - **session**
      - **application**
      - **page**
      - **...**

Programmation côté serveur

- Les outils de programmation Java serveur
  - Les Java Server Pages (JSP)
    - Syntaxe des balises de directives : **<%@** code **%>**
      - Trois types de directives
        - **page** : permet de modifier les données de la page (import de packages, spécification d'un type de contenu, gestion des sessions)

          <%@ page import="java.util." %>*
        - **include** : permet d'inclure des fichiers ou autres servlets/JSP,

          <%@ include page="/monJSP.jsp" flush="true" %>
        - **taglib** : permet d'utiliser des bibliothèques de balises personnalisées

          <%@ taglib uri="..." prefix="..." %>

Programmation côté serveur

- Les outils de programmation Java serveur
  - Les Java Server Pages (JSP)
    - Exemple de code simple : une JSP qui compte le nombre de fois où elle a été appelée

```
<html>
  <head><title>Déclarations et expressions</title></head>
  <body>
    <h1>Déclarations JSP</h1>

    <%! private int accessCount = 0; %>
    <p>Cette page a été accédée <%= ++accessCount %>
fois depuis le démarrage du serveur</p>

  </body>
</html>
```