Applications orientées données (NSY135)

5 – Vues: JSP et **tag libraries**

Auteurs: Raphaël Fournier-S'niehotta et Philippe Rigaux
(philippe.rigaux@cnam.fr,fournier@cnam.fr)

Département d'informatique
Conservatoire National des Arts & Métiers, Paris, France

Plan du cours

S1 JSP (Java Server Pages)

Introduction

- JSP : techno ancienne améliorée par
  - meilleure syntaxe
  - conventions de codage
- les langages de **vues** proposés par les **frameworks** sont proches des JSP : ce chapitre passe en revue les principes généraux

Approche

- Extension de HTML avec balises spéciales interprétées comme du Java
- Objectif : syntaxe uniforme mêlant agréablement les parties statiques et dynamiques
- Exemples (toujours valides):
  - <%-- --%> est un commentaire JSP;
  - <%! %> est une balise de déclaration d'une variable;
  - <% %> est une balise d'inclusion de code Java quelconque;
  - <%= %> est une balise d'affichage.

- **Cette approche fondée sur l'inclusion vaguement déguisée de code est maintenant dépréciée et doit être évitée.**
- Avantage : repose sur l'idée d'éviter de surcharger notre code avec des out.println devant chaque ligne HTML
- Inconvénient : laisse la porte ouverte à un code illisible et peu maintenable

Approche (suite)

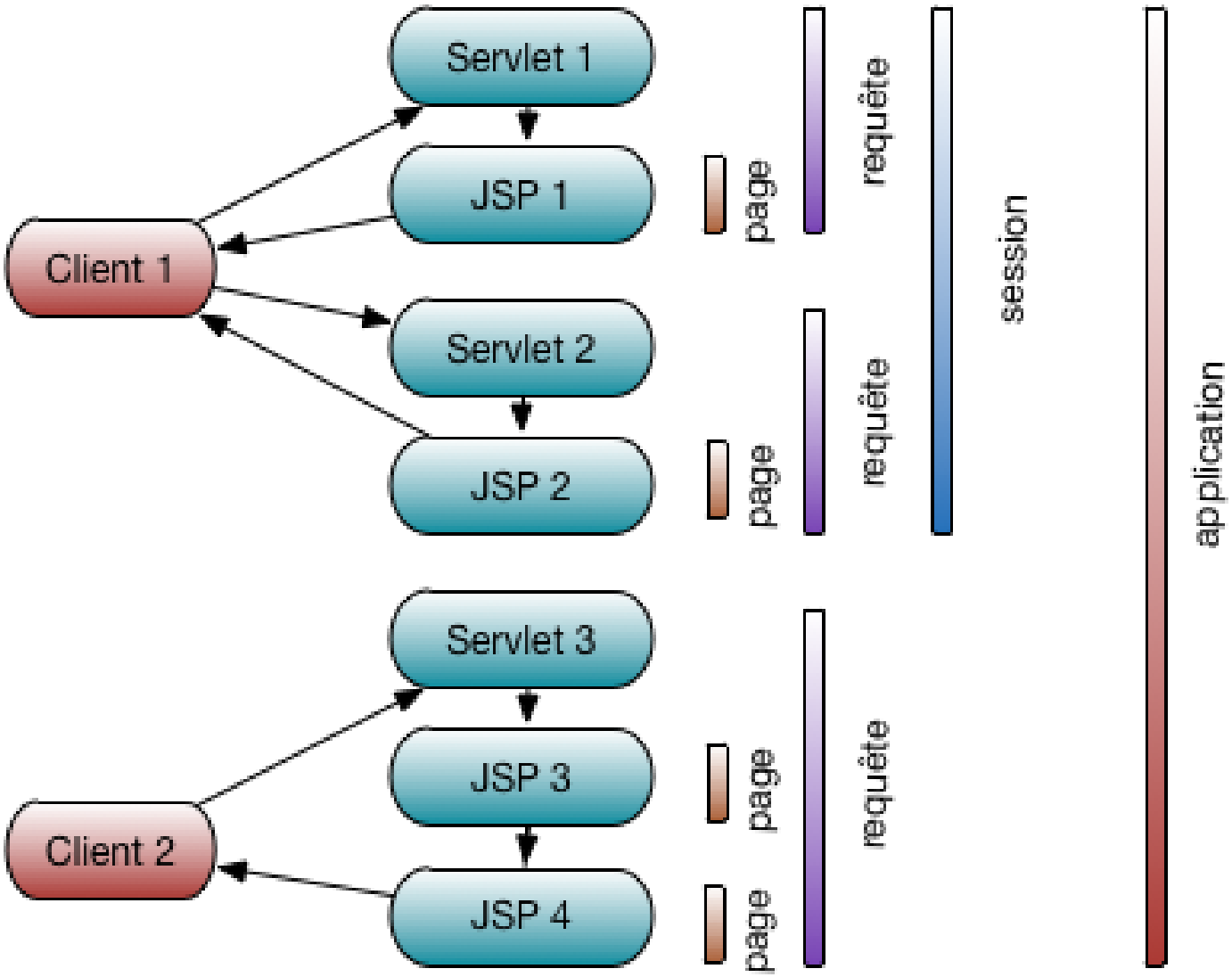
- Il y a aussi des **directives** sous forme de balises <%@ %>
- Exemple : import de packages Java :
  ```
  <%@ page import="java.util.Date"  %>
  ```
- les JSP créées précédemment contiennent une directive initiale indiquant l'encodage de la page.
- amélioration de JSP : extension via des **bibliothèques de balises** (**tag libraries**)
- exemple de déclaration
  ```
  <%@ taglib uri="NsyTagLib.tld" prefix="nsy" %>
  ```
- **prefix** désigne un espace de nommage (**namespace**), afin d'identifier la bibliothèque d'appartenance d'une balise :
  Par exemple, une balise "<nsy:conv />" sera identifiée comme appartenant à la bibliothèque "NsyTagLib.tld", et le code associé sera exécuté.
- **include** qui permet de composer des pages HTML à partir de plusieurs fragments :
  ```
  <%@ include file="header.jsp" %>
  ```

  Ces directives sont anciennes et peuvent souvent être remplacées par une approche plus moderne et plus élégante.

Portée de visibilité

- un objet instancié dans la servlet-**contrôleur** peut être mis à disposition de la vue en l'affectant à l'objet-requête `request`
- plus généralement, il existe **quatre portées de visibilité** des objets dans l'application (dont `request`)
- Un objet placé dans une portée est accessible par tous les composants de l'application (par exemple les JSP) qui ont accès à cette portée
- les 4 portées :
  - la portée **page**: tout objet instancié dans une JSP est accessible dans cette même JSP; c'est la portée la plus limitée;
  - la portée **request**: les objets placés dans cette portée restent accessibles durant toute l'exécution d'une requête HTTP par le serveur, même si l'exécution de cette requête implique plusieurs contrôleurs (par des mécanismes de "forward") ou plusieurs JSP;
  - la portée **session**, une session étant constituée par un ensemble d'interactions client/serveur, un objet placé dans la session (par exemple le fameux panier des achats dans un site de commerce) reste accessible sur la période couverte par ces interactions;
  - enfin la portée **application** est la plus générale de toute: un objet placé dans cette portée reste accessible tant que l'application est active dans le conteneur de servlets.
- Si la portée n'est pas indiquée, le conteneur va explorer les portées pour chercher un objet portant le nom indiqué. Cette pratique est source d' ambiguité et d'erreurs difficiles à détecter : indiquez les portées !

Portées de variables



Expressions

- On souhaite évidemment pouvoir utiliser les objets accessibles dans une JSP, et afficher leurs propriétés dans la vue
- On utilise pour cela un **langage d'expressions** (c'est-à-dire des instructions
- Syntaxe :

```
${expression}
```

- À l'exécution, le moteur de servlet évalue l'expression, et la valeur est insérée dans la page HTML produite, en substitution de l'expression
- Exemple : `${ 12 + 8 }` sera remplacé par `20`
- C'est une simplification syntaxique de la balise ancienne des JSP <%= %>

Propriétés des objets

- `${requestScope.monbean.maprop}`

  a pour effet d'afficher la propriété `maprop` de l'objet `monbean` placé dans la portée `request`
- Si cette propriété n'est pas publique (ce qui est toujours recommandé), le conteneur de servlet va chercher une méthode `getMaprop()` (un **"getter"** selon les conventions JavaBeans
- Plus généralement le conteneur va chercher à "deviner" quelle est l'interprétation naturelle de l'expression.
- Si votre objet `mamap` par exemple est une `HashMap` et contient une entrée indexée par `bill`, alors:

  `${requestScope.mamap.bill}`

  renverra la valeur de l'entrée "mamap['bill']"
- Les expressions sont tolérantes aux objets manquants :

  `${requestScope.mampa.bill}`

  renverra rien, (notez que "mampa" est mal écrit), alors l'objet n'existe pas dans la portée et qu'une exception "null value" devrait être levée

Objets implicites

- L'objet `requestScope` que nous utilisons dans les exemples qui précèdent est un **objet implicite** accessible par les expressions.
- Il en existe d'autres, l'ensemble constituant un contexte de communication entre le contrôleur (la servlet) et la vue (la page JSP).

Nom	Type	Description
`pageContext`		Informations sur l'environnement du serveur.
`pageScope`	`Map`	Attributs de portée **page**.
`requestScope`	`Map`	Attributs de portée `request`.
`sessionScope`	`Map`	Attributs de portée `session`.
`applicationScope`	`Map`	Attributs de portée `application`.
`param`	`Map`	Les paramètres de la requête HTTP, pour les paramètres monovalués.
`paramValues`	`Map`	Les paramètres de la requête HTTP, pour les paramètres multivalués
`header`	`Map`	Les paramètres de l'entête HTTP, pour les paramètres monovalués.
`headerValues`	`Map`	Les paramètres de l'entête HTTP, pour les paramètres multivalués
`cookie`	`Map`	Les `cookies`.
`initParam`	`Map`	Les paramètres du fichier `web.xml`.

♠ Il est évidemment tout à fait déconseillé de créer un objet portant le même nom que ceux du tableau