Struts

Construction de
sites Web dynamiques

Qu'est-ce que c'est que Struts

Environnement logiciel *(framework)* pour aider au développement de sites Web dynamiques

- Séparer la présentation des traitements
- Séparer le code HTML du code informatique (Java)
- Simplifier l'écriture des pages grâce à des balises spécialisées
- Séparer la navigation entre les pages de leur localisation effective sur disque

Strut : support, entretoise (charpente) – se pavaner

Site Web dynamique

Un site pour lequel les pages ne peuvent pas être préparés à l'avance, mais seulement lors de leur consultation

Exemple : les pages Auger pour le suivi des transferts de fichiers entre l'Argentine et Lyon

- La page à présenter dépend de l'état du transfert
- Elle dépend aussi de la requête :
  - Transfert en cours
  - Transfert de la veille
  - Liste des incidents ...

Les pages sont construites « au vol » (dynamiquement) par consultation de la base de données.

Un site dynamique n'est pas un site interactif ce n'est pas un wiki

Pourquoi est-ce si compliqué

- Le langage HTML est destiné à décrire des pages statiques
- Le serveur HTTP ne sait pas exécuter une opération pour traiter une requête
  - Il délègue le traitement à un programme extérieur (CGI : Common Gateway Interface)
    - Celui-ci récupère les paramètres de la requête
    - Vérifie leur validité
    - Traite la requête
    - Construit la page de réponse pour présenter les résultats
    - Traite les erreurs et les présente dans des pages spécialisées

Problèmes

- Mélange de HTML, de code de traitement (Perl, Php, Python, Tcl), et de redirection vers les pages d'erreur ou d'explication
- Peu performant
  - Créer un processus est lourd
  - Le contexte doit être rétabli à chaque fois
    - Connexion à la base de données ...
- Homogénéisation difficile de la présentation
  - Le code de génération des pages est souvent très rigide
- Très difficile à faire évoluer
  - Il faut parcourir les différents scripts et décomposer ce qui est traitement de ce qui est affichage

Java et les *servlets*

Tentative pour améliorer les performances et la lisibilité

Servlet : classe Java exécutée par le serveur pour répondre à une requête

- Comme un CGI sauf qu'elle tourne dans le noyau si le serveur supporte Java (TomCat / Apache)
  - Charger une seule fois → *performances*
  - Peu conserver un contexte → *performances*

    Connexion à la base de données ...
  - Exécuter dans un *thread* Java → *performances*

Pourquoi Java

- Gestion saine de la mémoire
  - Pas de fuites...
  - Pas de corruption par débordement des tableaux ou des listes

  → *Ne met pas en péril le serveur*
- Support des *exceptions*
  - Le serveur peut récupérer les erreurs du programmeur

  → *Ne met pas en péril le serveur*
- Langage fortement typé
  - La mise au point du code est beaucoup plus rapide qu'avec un script (Perl, Php)
- Développement rapide
  - Par exemple, sous ***Eclipse***, la compilation se fait "au vol", directement dans le répertoire d'exécution du serveur

JSP

Tentative de séparer efficacement la présentation des traitements

- Pour l'essentiel, une page HTML avec « des bouts » de Java dedans
  - Permet d'exécuter des traitements (simples) directement dans la page Web
  - Exécuter côté serveur
    - **Attention :** ce n'est **pas** du *JavaScript,* qui s'exécute côté client

JSP *Java Servlet Page*

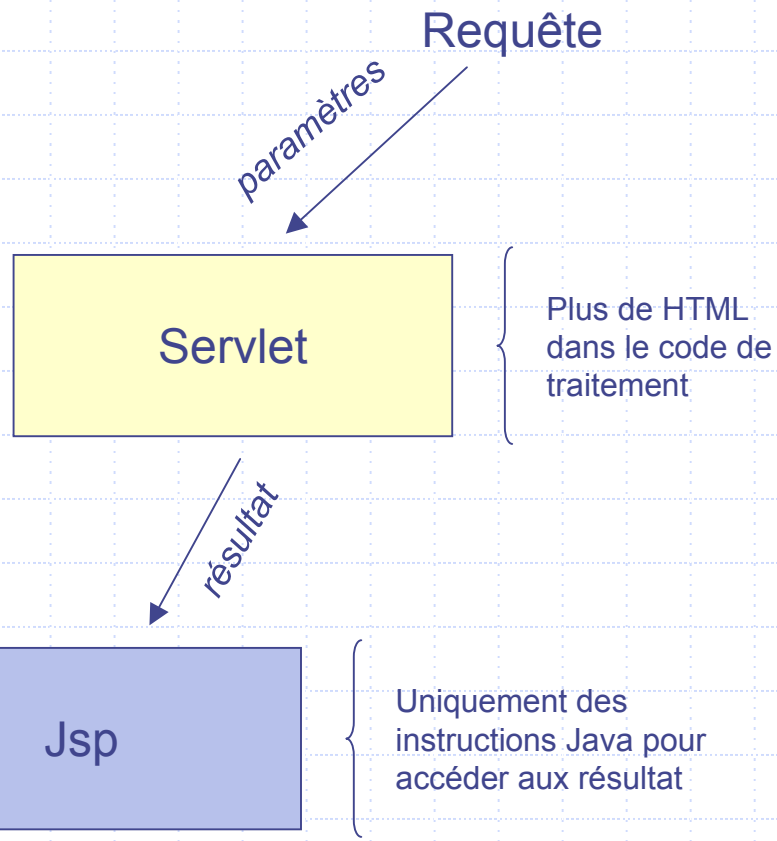
Attention :

en réalité, une JSP est une *servlet*

→ ***c'est donc du Java avec beaucoup de HTML dedans mais la génération de la servlet est automatique***

Chaînage

- La *servlet* reçoit la requête et la traite
- Elle construit un « résultat »
  un simple objet Java
- Elle passe le résultat à la JSP
- La JSP présente le résultat



Exemple

```
<%@ page language="java" %>

<html>
<head>
    <title>Démonstration</title>
</head>

<% Result result = (Result)request.getAttribute("result"); %>

<body>
<table …>
  <tr>
     <td>Nombre de fichiers</td>

     <td> <%= result.getNbFichiers() %> </td>
  </tr>
</table>

</body>
</html>
```

Struts

Extrait deux composants de la *servlet*

Action Form

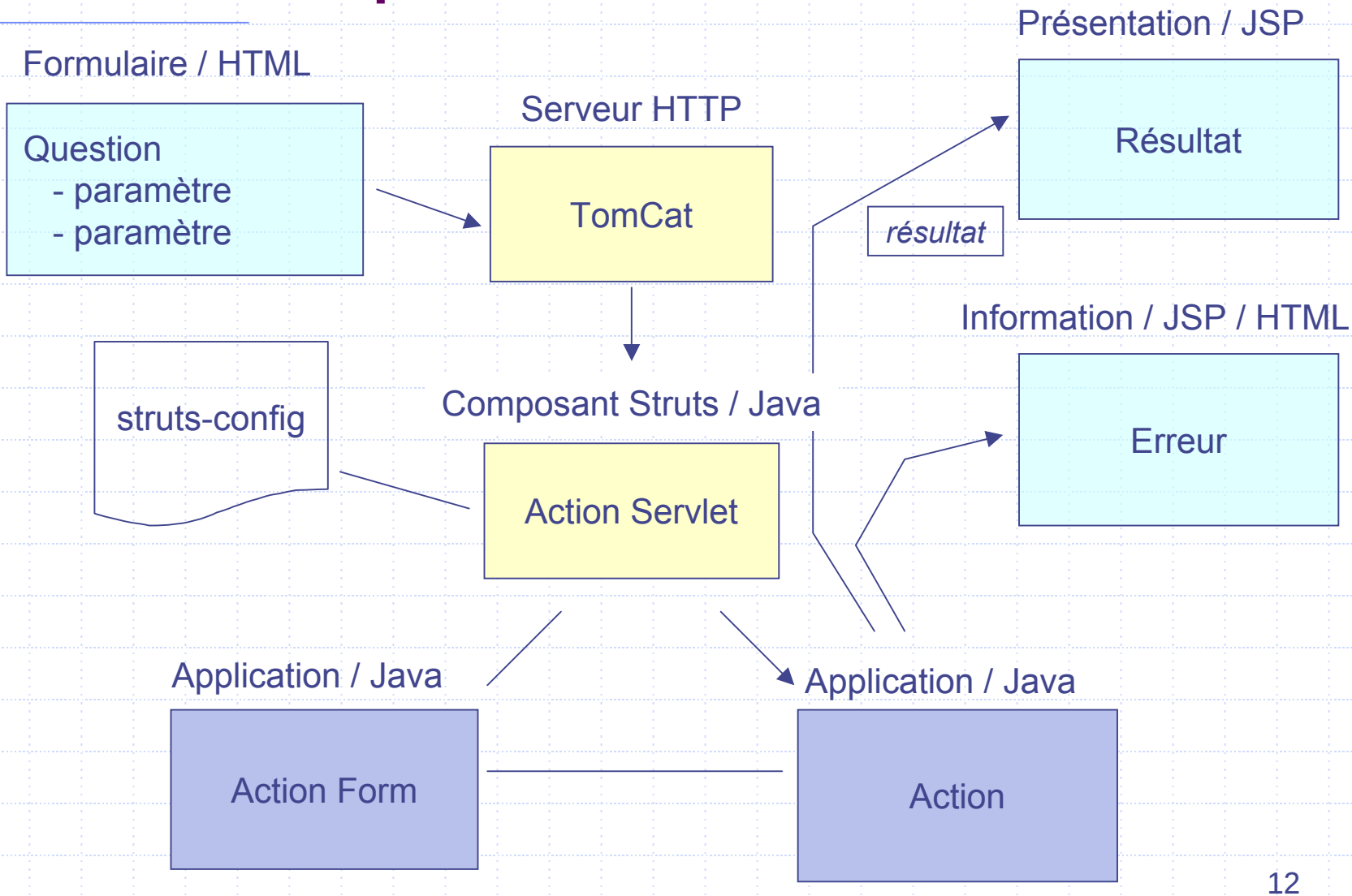
- Reçoit les paramètres associés à un formulaire - Initialisé automatiquement par ***Struts***
- Les vérifie (validation)

Action

- Traite la requête à partir des paramètres validés de l'Action Form
- Construit le résultat
- Passe le contrôle à une JSP pour la présentation *(forward)*

- Une seule *servlet* pour tous les traitements (fournie par Struts)
- Elimine une grande partie de code répétitif pour l'initialisation des *servlets* et la communication avec le serveur
- Meilleure décomposition entre traitement, présentation et navigation
- Conception plus facile et plus modulaire du site
  - Un formulaire pour la question
  - Une Action Form pour recevoir les paramètres et les valider
  - Une Action pour le traitement
  - Une JSP pour la présentation

Schématiquement



Balises spécialisées

Bibliothèques de balises pour

- Accéder aux éléments d'un composant (le *résultat)*
- Procéder à quelques opérations simples
  - Présence ou absence d'un résultat, validité de l'un des attributs
  - Itération sur une liste de valeurs
- Construire un formulaire

- Mieux isoler les traitements de la présentation
  - Supprime le code Java des pages JSP
- Simplifier la délégation de la présentation à des infographistes

Exemple

```
<%@ page language="java" %>
<%@ taglib uri="/WEB-INF/lib/struts-bean.tld" prefix="bean" %>

<html>
<head>
    <title>Démonstration</title>
</head>

<body>
<table …>
  <tr>
     <td>Nombre de fichiers</td>

     <td> <bean:write name="result" property="nbFiles"/> </td>
  </tr>
</table>

</body>
</html>
```

Navigation entre les éléments

- Utiliser des noms « symboliques » pour l'enchaînement des pages
- Ne ***pas*** utiliser des noms de fichiers

Exemple:

Plutôt que :

http://augedb/imports/manager/rapport.jsp

Utiliser :

Import_rapport.do

La correspondance entre le nom symbolique et le nom du fichier est fait dans un *unique* fichier de configuration

Simplifie les réorganisations du site

Un seul fichier à modifier lorsque les pages WEB sont déplacées

Eclipse

Environnement de développement pour Java (IDE)

- Intègre le *serveur* TomCat
- Support *Struts*
- Permet le développement rapide des *servlets* et des JSP grâce au *debugger*

Struts ...

- Décompose une grosse application monolithique (CGI Perl, Php) en composants spécialisés plus simples
  - Facilite l'évolution du site
- Sépare nettement présentation, validation et traitements
  - Facilite la collaboration d'infographistes et d'informaticiens
- Suppose « d'en avoir l'usage »
  - C'est un investissement
  - Nécessite un minimum de pratique