



## Présentation Struts

Max Devulder

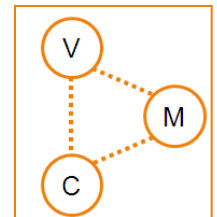
24 Slides

# *Sommaire*

- I. **Présentation MVC/Struts**
- II. Concepts clés
- III. Éléments transverses : taglib pour IHM, i18n

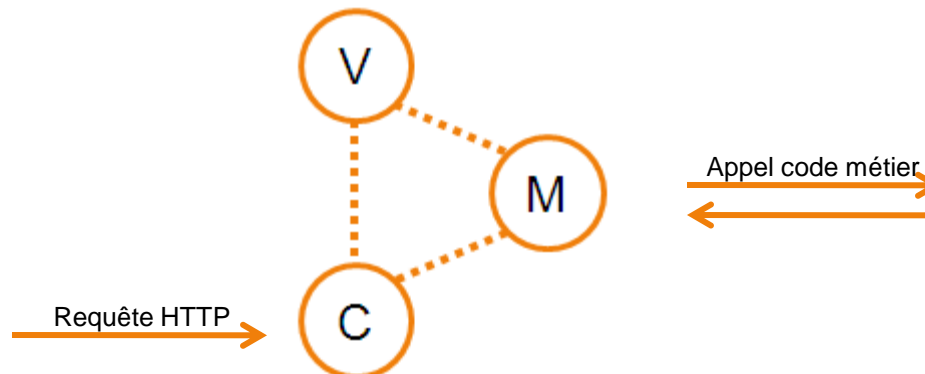


- **Framework** développé par Apache (comme Tomcat)
- Basé sur J2EE (JSP et servlets)
  - Surcouche J2EE,
  - Toute application Struts est une application web J2EE
- Ajoute de nouvelles classes (Java) /tags (Jsp)
- Basé sur l'architecture MVC (Modèle-Vue-Contrôleur)
- Populaire



Le Modèle-Vue-Contrôleur (MVC) est un **Design Pattern** pour le développement d'applications logicielles.

- **Modèle** (Musiciens) : Application des traitements de l'application
- **Vue** (Spectateurs) : Interface avec l'utilisateur.
- **Contrôleur** (Chef d'orchestre) : Gestion des événements de synchronisation entre la vue et le modèle.





- Des servlets (et autres classes) qui gèrent l'aspect métier (modèle)

M

- Des JSP qui produisent l'affichage (les vues)

V

- Un (ou plusieurs) Filter qui fait le lien entre les **M**odèles et les **V**ues, exemple simple via une MAP<Vue, Modele>


C



- Déporter le Contrôleur dans un fichier de mapping XML (**struts-config.xml**)
- Struts s'occupe de lire/parcourir le fichier (via sa servlet : ActionServlet), à définir dans le **web.xml** :

```
<web-app>
  <servlet>
    <servlet-name>action</servlet-name>
    <servlet-class>org.apache.struts.action.ActionServlet</servlet-class>
    <load-on-startup>1</load-on-startup>
  </servlet>
  ...
</web-app>
```

```
<servlet-mapping>
  <servlet-name>action</servlet-name>
  <url-pattern>*.do</url-pattern>
</servlet-mapping>
```

 Par convention, les URL des applications Struts sont mappées sur \*.do

### Avantages

- Le code Java se limite désormais aux aspects métiers.
- Souplesse : pour changer le comportement de l'application, il suffit de changer le code XML.
- Moins de compilation.

# *Sommaire*

---

- I. Présentation MVC/Struts
- II. Concepts clés**
- III. Éléments transverses : Taglib, i18n ...

## Exemple simple d'utilisation (sans formulaire)

← → ↺ 🏠

1

struts-config.xml

```
<action-mappings>
  <action path="/voirPanier"
    type="com.monsiteecommerce.actions.VoirPanierAction" >
    <forward name="success" path="/pages/monPanier.jsp"/>
    <forward name="failed" path="/pages/login.jsp"/>
  </action>
</action-mappings>
```

2

VoirPanierAction.java

```
public class VoirPanierAction extends Action{

  public ActionForward execute(final ActionMapping mapping, final ActionForm form, final
  HttpServletRequest req, final HttpServletResponse res) {

    // APPEL DU METIER ICI

    if(errors.isEmpty()){
      return mapping.findForward("success");
    }else{
      return mapping.findForward("failed"); }
  }
}
```

3

4

1. Demande d'url



2. Check struts-config.xml pour trouver l'action



3. Lance la méthode execute()



4. Retourne la jsp associée au FW



- Les éléments importants

Attribut/Élément	Description
Path	L'url de l'action, sans le suffixe. Obligatoire Doit être unique.
Type	La classe java de l'action Obligatoire Doit être unique. Doit étendre : org.apache.struts.action.Action
Scope	Scope pour le form bean « Session » ou « Request » (par défaut « Request »).
Validate	Active la validation du form bean true : lancera la méthode validate() du form bean.
Input	JSP à partir de laquelle l'action démarre.
Forward	1 ou plusieurs pages (ou action) identifiées par un mot clé

### struts-config.xml

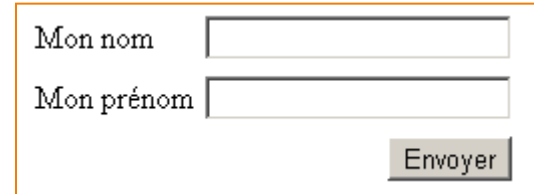
```
<action-mappings>
  <action path="/voirPanier"
    type="com.monsiteecommerce.actions.VoirPanierAction" >
    <forward name="success" path="/pages/monPanier.jsp"/>
    <forward name="failed" path="/pages/login.jsp"/>
  </action>
</action-mappings>
```

## Exemple avec formulaire

- Formulaire jsp classique (*taglib <html: fournies par struts, cf. chapitre suivant*)

### utilisateur.jsp

```
<html:form action="bonjourUtilisateur" >
  Nom <html:text property="nom" />
  Prénom <html:text property="prenom" />
  <html:submit property="submit" value="Envoyer" />
</html:form>
```



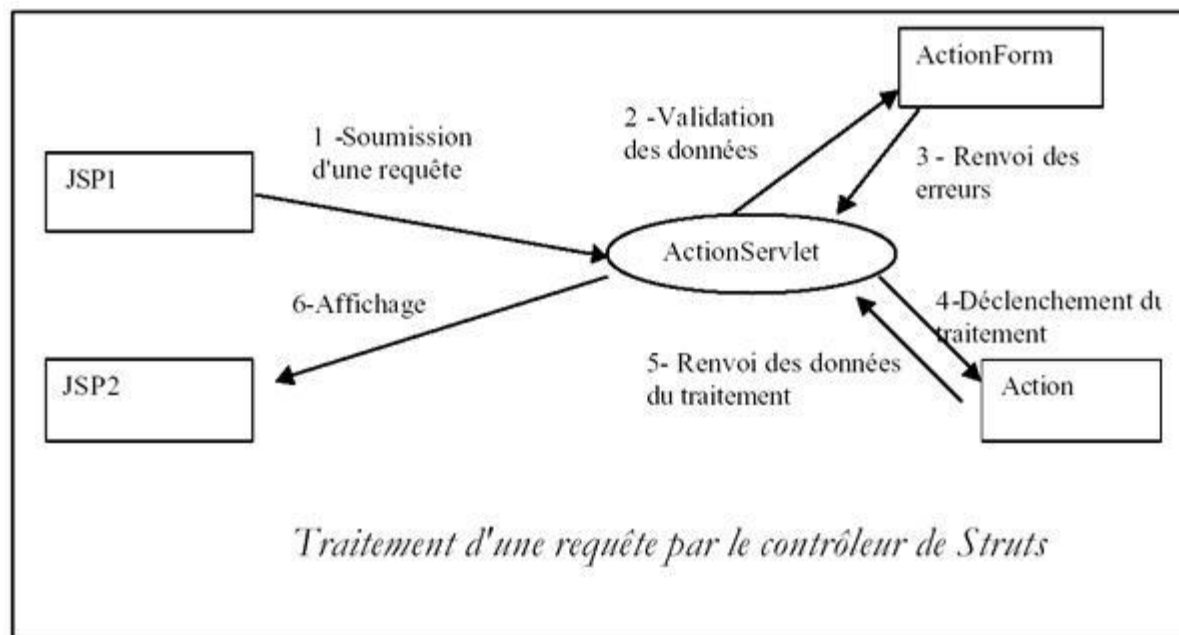
- Correspondance objet dans un bean.
  - Doit étendre `org.apache.struts.action.ActionForm`
  - Attributs de type String.
  - Respecte la convention de nommage java : `nom` ⇔ `private String nom;`  
`+ getNom()`  
`+ setNom()`

### UtilisateurForm .java

```
public class UtilisateurForm extends ActionForm {

  private String nom;
  private String prenom;

  public String getNom () { ... }
  public String setNom (final String nom) { ... }
  public String getPrenom () { ... }
  public String setPrenom (final String prenom) { ... }
}
```




## Exemple avec formulaire

### struts-config.xml

```
<struts-config>
  <!-- Définition des Formulaires -->
  <form-beans type="org.apache.struts.action.ActionFormBean">
    <form-bean name="loginForm" type="mywebapp.bean.vo.UtilisateurForm" />
  </form-beans>

  <!-- Définition des Actions -->
  <action-mappings type="org.apache.struts.action.ActionMapping">
    <action path="/bonjourUtilisateur" input="jsp/utilisateur.jsp" scope="request" name="loginForm" type="mywebapp.action.BonjourAction">
      <forward name="superForward" path="/..." redirect="false" />
      <forward name="forwardBof" path="/..." redirect="false" />
    </action>
  </action-mapping>
</struts-config>
```

*Liaison*



- Que fait struts ?

1. L'utilisateur valide le formulaire => /bonjourUtilisateur.do
2. Ouverture du struts-config pour chercher l'action correspondante => [BonjourAction](#)
3. Form-bean associé : [loginForm](#), recherche dans <form-beans > => [UtilisateurForm.java](#)
4. input = [jsp/utilisateur.jsp](#) instanciation du bean + injection des valeurs via le formulaire.
5. Lance la méthode validate() de [UtilisateurForm](#)
6. Lance la méthode execute() de [BonjourAction](#)
7. mapping.findForward()

1. Demande d'url

2. Check struts-config.xml

3+4. Instanciation du javaBean

5. Validation de surface

6. Execution du Model

7. Forward

## Objectif premier de la couche web : Contrôler les informations saisies à l'écran.

Le champ nom saisi est incorrect.

Mon nom	<input type="text" value="1234"/>
Mon prénom	<input type="text" value="Nicolas"/>
<input type="button" value="Envoyer"/>	

utilisateur.jsp

```
<html:form action="bonjourUtilisateur" >
  <html:errors />
  .....
</html:form>
```

- Ajout de la balise `<html:errors />` à la jsp.
- Nécessité d'implémenter la méthode `validate()` de `org.apache.struts.action.ActionForm`

UtilisateurForm .java

```
public class UtilisateurForm extends ActionForm {
    ....
    @Override
    public ActionErrors validate(final ActionMapping mapping, final HttpServletRequest request) {

        final ActionErrors errors = new ActionErrors();

        if (StringUtils.isEmpty(getNomUtilisateur())) {
            errors.add("nom", new ActionMessage("errors.mandatory", "nom"));
        }

        return errors;
    }
}
```

# *Sommaire*

---

- I. Présentation MVC/Struts
- II. Concepts clés
- III. Éléments transverses : Taglib, i18n ...**

## Permet l'internationalisation <sup>(i18n)</sup> des messages et erreurs.

- Fichier ApplicationResources.properties (convention)

### struts-config.xml

```
<struts-config>
  <!-- Définition des Formulaires -->
  <form-beans type="org.apache.struts.action.ActionFormBean">
  </form-beans>

  <!-- Définition des Actions -->
  <action-mappings type="org.apache.struts.action.ActionMapping">
  </action-mapping>

  <!-- Application ressource -->
  <message-resources parameter="ApplicationResources"/>

</struts-config>
```

### ApplicationResources.properties

```
# encapsulation des messages
errors.prefix=<div class="error">
errors.suffix=</div>

# erreurs générales.
errors.field.maxLength=Le champ '{0}' ne peut être supérieur à {1} caractères.
errors.field.notFound=Le champs '{0}' est obligatoire.

# Messages par UC
message.login.input.utilisateur=Mon nom
```

## Plusieurs façons d'y accéder :

A la validation de surface via l'objet **ActionMessage**

UtilisateurForm .validate

```
if (StringUtils.isEmpty(getNomUtilisateur())) {  
    errors.add("nom", new ActionMessage("errors.field.notFound", "Nom utilisateur"));  
}
```

Dans les JSP via le tag **<bean:message>**

utilisateur.jsp

```
<tr>  
    <td align="right"><bean:message key="message.login.input.utilisateur" /> :</td>  
    <td align="left"><html:text property="nomUtilisateur" size="20" maxlength="20" /></td>  
</tr>
```

Le champ nom saisi est incorrect.

Mon nom	<input type="text" value="1234"/>
Mon prénom	<input type="text" value="Nicolas"/>
<input type="button" value="Envoyer"/>	



### Plusieurs lib fournie :

- **Bean** : *Manipulation pure de beans java*
- **Html** : *Manipulation des formulaires et éléments de formulaires HTML*
- **Logic** : *Mise en place de traitements conditionnels et/ou itératifs*

### Import :

utilisateur.jsp

```
<%@ taglib uri="/WEB-INF/struts-bean-1.2.tld" prefix="bean"%>  
<%@ taglib uri="/WEB-INF/struts-html-1.2.tld" prefix="html"%>  
<%@ taglib uri="/WEB-INF/struts-logic-1.2.tld" prefix="logic"%>
```



Par convention

### > Bean

- Amélioration de la taglib `<jsp:`
- 1 seul point d'entrée pour tous les context (page, session, application)
- Permet l'accès au fichier `ApplicationResources`

Tag	Description	Exemple
<b>write</b>	Affiche une propriété d'un bean.	<code>&lt;bean:write name='item' property='prize'/&gt;</code> <code>&lt;bean:write name='items'</code> <code>property='item[2].prize'/&gt;</code>
<b>message</b>	Affiche la valeur de la clé définie dans le <code>ApplicationResources.properties</code>	<code>&lt;bean :message key='titre.index'/&gt;</code>

### > Bean

- Et bien d'autres : <http://struts.apache.org/release/1.2.x/userGuide/struts-bean.html>

Tag Name	Description
<a href="#">cookie</a>	Define a scripting variable based on the value(s) of the specified request cookie.
<a href="#">define</a>	Define a scripting variable based on the value(s) of the specified bean property.
<a href="#">header</a>	Define a scripting variable based on the value(s) of the specified request header.
<a href="#">include</a>	Load the response from a dynamic application request and make it available as a bean.
<a href="#">message</a>	Render an internationalized message string to the response.
<a href="#">page</a>	Expose a specified item from the page context as a bean.
<a href="#">parameter</a>	Define a scripting variable based on the value(s) of the specified request parameter.
<a href="#">resource</a>	Load a web application resource and make it available as a bean.
<a href="#">size</a>	Define a bean containing the number of elements in a Collection or Map.
<a href="#">struts</a>	Expose a named Struts internal configuration object as a bean.
<a href="#">write</a>	Render the value of the specified bean property to the current JspWriter.

### > Html

- Création IHM, formulaire
- Générer facilement lien & sessions

Tag	Description	Exemple
<b>text</b>	Champ input de type text	<code>&lt;html:text property='username'/&gt;</code>
<b>message</b>	Champ input de type password	<code>&lt;html:password property='password'/&gt;</code>
<b>submit</b>	Soumission du formulaire	<code>&lt;html:submit value="Envoyer"/&gt;</code>
<b>errors</b>	Affiche les erreurs contenu dans l'objet ActionErrors	<code>&lt;html:errors /&gt;</code>

### > Html

- Et bien d'autres : <http://struts.apache.org/release/1.2.x/userGuide/struts-html.html>

Tag Name	Description
<a href="#">base</a>	Render an HTML <base> Element
<a href="#">button</a>	Render A Button Input Field
<a href="#">cancel</a>	Render a Cancel Button
<a href="#">checkbox</a>	Render A Checkbox Input Field
<a href="#">errors</a>	Conditionally display a set of accumulated error messages.
<a href="#">file</a>	Render A File Select Input Field
<a href="#">form</a>	Define An Input Form
<a href="#">frame</a>	Render an HTML frame element
<a href="#">hidden</a>	Render A Hidden Field
<a href="#">html</a>	Render an HTML <html> Element
<a href="#">image</a>	Render an input tag of type "image"
<a href="#">img</a>	Render an HTML img tag
<a href="#">javascript</a>	Render JavaScript validation based on the validation rules loaded by the ValidatorPlugIn.
<a href="#">link</a>	Render an HTML anchor or hyperlink
<a href="#">messages</a>	Conditionally display a set of accumulated messages.
<a href="#">multibox</a>	Render A Checkbox Input Field
<a href="#">option</a>	Render A Select Option
<a href="#">options</a>	Render a Collection of Select Options
<a href="#">optionsCollection</a>	Render a Collection of Select Options
<a href="#">password</a>	Render A Password Input Field
<a href="#">radio</a>	Render A Radio Button Input Field
<a href="#">reset</a>	Render A Reset Button Input Field
<a href="#">rewrite</a>	Render an URI
<a href="#">select</a>	Render A Select Element
<a href="#">submit</a>	Render A Submit Button
<a href="#">text</a>	Render An Input Field of Type text
<a href="#">textarea</a>	Render A Textarea
<a href="#">xhtml</a>	Render HTML tags as XHTML

### > Logic

- Itération, test
- Evite l'écriture lourde de scriptlets java `<% %>`

Tag	Description	Exemple
<b>greaterThan</b>	<b><u>Opérateur « &gt; »</u></b> If( <code>user.getAge() &gt; 17</code> ){ <code>syso(« Vous êtes majeur »);</code> }	<code>&lt;logic:greaterThan name='user' property='age' value='17'&gt;</code> Vous êtes majeur ! <code>&lt;/logic:greaterThan&gt;</code>
<b>iterate</b>	<b><u>Boucle for :</u></b> <code>private List&lt;Enfant&gt; enfants;</code>  <code>for(Enfant monEnfant : enfants){</code> <code>syso(monEnfant.getNom());</code> }	<code>&lt;logic:iterate name='user' property='enfants' id='monEnfant'&gt;</code> <code>&lt;bean:write name=' monEnfant.nom' /&gt;&lt;br&gt;</code> <code>&lt;/logic :iterate&gt;</code>

### > Logic

- Et bien d'autres : <http://struts.apache.org/release/1.2.x/userGuide/struts-logic.html>

Tag Name	Description
<a href="#">empty</a>	Evaluate the nested body content of this tag if the requested variable is either null or an empty string.
<a href="#">equal</a>	Evaluate the nested body content of this tag if the requested variable is equal to the specified value.
<a href="#">forward</a>	Forward control to the page specified by the specified ActionForward entry.
<a href="#">greaterEqual</a>	Evaluate the nested body content of this tag if the requested variable is greater than or equal to the specified value.
<a href="#">greaterThan</a>	Evaluate the nested body content of this tag if the requested variable is greater than the specified value.
<a href="#">iterate</a>	Repeat the nested body content of this tag over a specified collection.
<a href="#">lessEqual</a>	Evaluate the nested body content of this tag if the requested variable is less than or equal to the specified value.
<a href="#">lessThan</a>	Evaluate the nested body content of this tag if the requested variable is less than the specified value.
<a href="#">match</a>	Evaluate the nested body content of this tag if the specified value is an appropriate substring of the requested variable.
<a href="#">messagesNotPresent</a>	Generate the nested body content of this tag if the specified message is not present in this request.
<a href="#">messagesPresent</a>	Generate the nested body content of this tag if the specified message is present in this request.
<a href="#">notEmpty</a>	Evaluate the nested body content of this tag if the requested variable is neither null, nor an empty string, nor an empty java.util.Collection (tested by the .isEmpty() method on the java.util.Collection interface).
<a href="#">notEqual</a>	Evaluate the nested body content of this tag if the requested variable is not equal to the specified value.
<a href="#">notMatch</a>	Evaluate the nested body content of this tag if the specified value is not an appropriate substring of the requested variable.
<a href="#">notPresent</a>	Generate the nested body content of this tag if the specified value is not present in this request.
<a href="#">present</a>	Generate the nested body content of this tag if the specified value is present in this request.
<a href="#">redirect</a>	Render an HTTP Redirect

## Conclusion

Faut-il utiliser struts ?

- Oui si l'application est conséquente.
- Des générateurs de code existent !
- Non s'il s'agit d'une application simple (exemple limite, le blog) : même le site de Struts déconseille d'utiliser le Framework dans ces cas-là.
- On peut utiliser des JSP seules, du PHP ou un framework plus léger dans les cas plus simples.