Emmanuel Puybaret

les Cahiers du Programmeur

Java (1)

ISBN : 2-212-11272-6

EYROLLES

Interface utilisateur du forum

12

Java

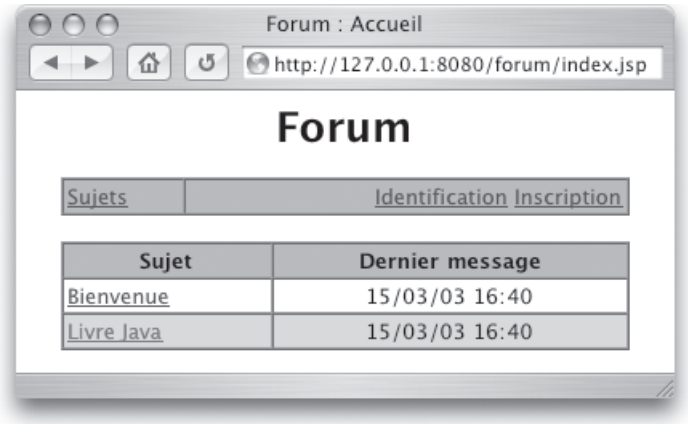
Architecture 3 tiers | jsp:usebean | scope | session | jsp:forward | jsp:include

SOMMAIRE

- Scénario d'utilisation
- Programmation des pages
- Composants JavaBeans
- Identification et inscription
- Page d'accueil
- Création de message

MOTS-CLÉS

- Architecture 3 tiers
- jsp:usebean
- scope
- session
- jsp:forward
- jsp:include



Ce chapitre montre comment intégrer les classes du forum déclarées dans les chapitres précédents pour créer l'interface utilisateur du forum grâce à des pages JSP hébergées sur un serveur Web.

Scénario d'utilisation

L'application de forum peut-être décrite à l'aide de scénarios d'utilisation, qui peuvent servir de base pour la réalisation des pages JSP (voir figure 12–1).

Le forum donne des droits d'utilisation différents selon que les utilisateurs sont identifiés ou pas :

- Un utilisateur non identifié peut consulter la liste des sujets du forum et leurs messages.
- Un utilisateur identifié peut en plus créer de nouveaux sujets, répondre à des sujets et modifier ses messages. Un modérateur a le droit de modifier tous les messages qu'il en soit l'auteur ou non.

Pour faciliter la navigation des deux types d'utilisateur, les liens hypertexte affichés dans les pages du forum deviennent différents une fois qu'un utilisateur s'est identifié (barre de navigation et liens Répondre et Modifier d'un message).

ALTERNATIVES Cas d'utilisation UML

Le lecteur pourra consulter avec profit l'ouvrage décrivant la modélisation UML d'un site de e-commerce.

📖 *Cahier du Programmeur UML – Modéliser un site de e-commerce*, Pascal Roques, Eyrolles 2002

Scénario pour un utilisateur non identifié

Quand un utilisateur arrive sur la page d'accueil du forum ❶, les sujets lui sont affichés sous forme de liste dans un ordre allant du plus récemment modifié au plus ancien. Il peut alors :

- Cliquer sur un sujet pour en consulter les messages ❷. En choisissant le lien Sujets ❸, il peut revenir à tout moment à la page d'accueil pour consulter les messages d'autres sujets.
- S'identifier s'il a déjà un pseudonyme et un mot de passe ❹.
- S'inscrire en choisissant un pseudonyme ❺. Le mot de passe qui lui est attribué par le serveur ❻ lui permet alors de s'identifier en cliquant sur le lien Identification ❼.

Scénario pour un utilisateur identifié

Une fois qu'un utilisateur a saisi son pseudonyme et son mot de passe correctement, il revient sur la page d'accueil ❽ où les liens Identification /Inscription ont été remplacés par un lien Quitter ❾. Il peut maintenant :

- Lire les messages d'un sujet ❿ et y répondre en cliquant sur le lien Répondre ⓫ qui ne s'affiche que pour les utilisateurs identifiés. Après avoir confirmé la saisie de son message ⓬, la page du sujet fait alors apparaître son nouveau texte.
- Éditer les messages ⓭ qu'il a rédigés pour modifier leur texte en cliquant sur le lien Modifier. Après confirmation de la modification ⓮, la page du sujet laisse alors apparaître son texte modifié.
- Créer de nouveaux sujets ⓯ en cliquant sur le lien Nouveau sujet dans la barre de navigation. Après avoir saisi le sujet et un message ⓰, la page du nouveau sujet est alors affichée.

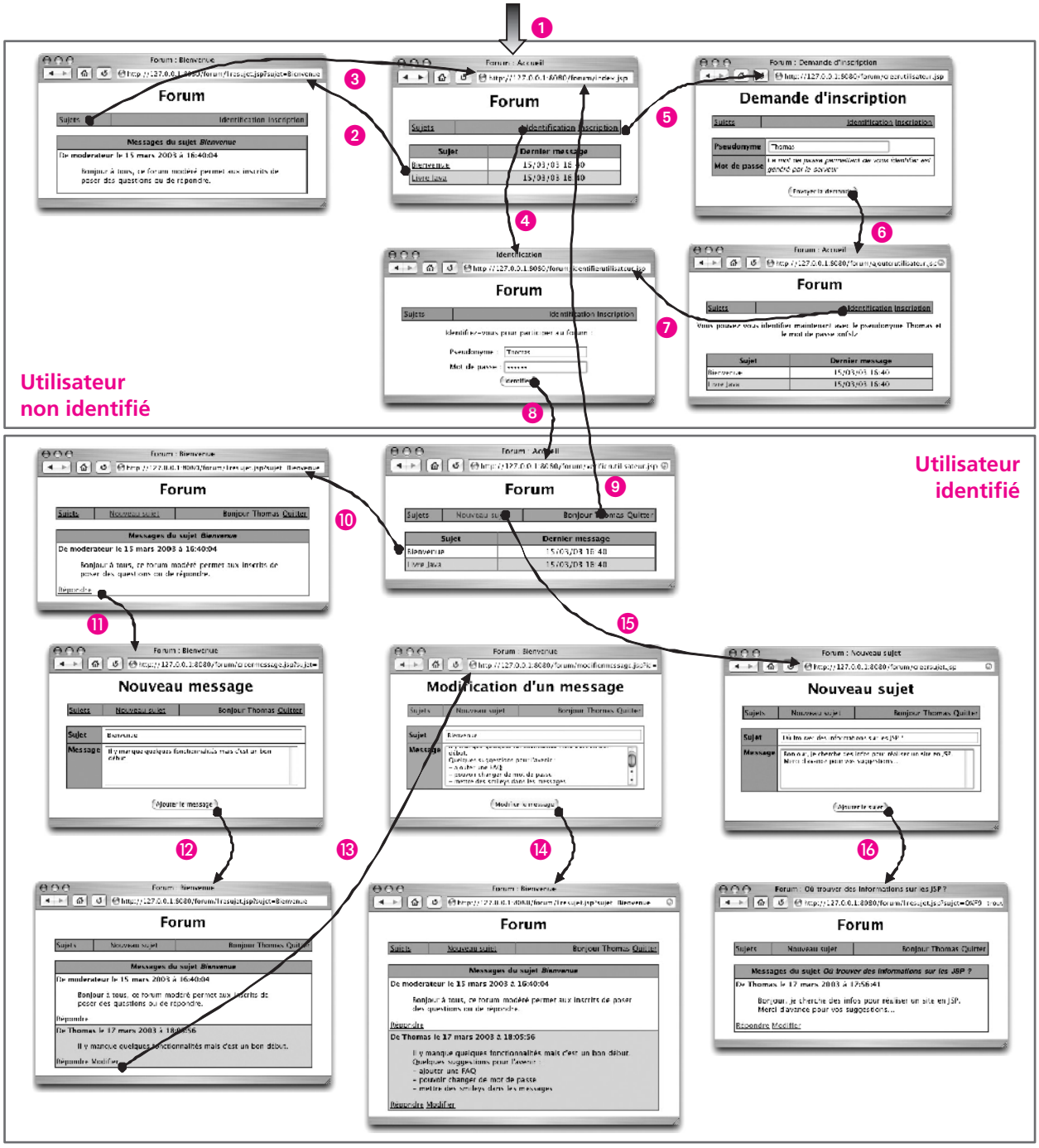


Figure 12–1 Scénario d'utilisation du forum

Programmation des pages du forum

Les utilisateurs et les messages du forum manipulés par les pages JSP du forum sont lus et sauvegardés dans une base de données grâce aux classes du paquetage `com.eteks.forum` déclarées au chapitre 10, « Connexion à la base de données avec JDBC ».

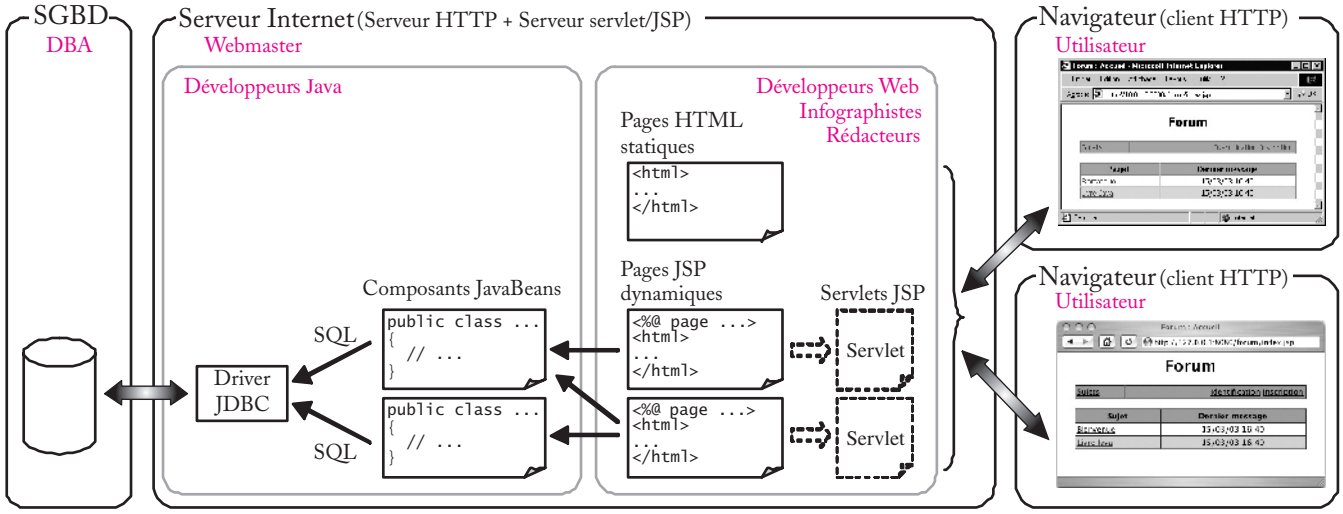


Figure 12–2 Architecture 3 tiers d'un site Web et intervenants

B.A.-BA Architecture n tiers

Une architecture n tiers répartit sur plusieurs serveurs et clients les fonctionnalités d'une application. Le forum que l'on étudie dans cet ouvrage est mis en œuvre avec une architecture 3 tiers typique :

- un SGBD chargé de la sauvegarde des données ;
- un serveur chargé d'exploiter les données du SGBD pour les mettre en page ;
- un client gérant l'affichage des données et les interactions avec l'utilisateur.

Organisation des pages du forum

Les pages intermédiaires lancées lors de la confirmation des formulaires et du lien Quitter n'affichent rien. Elles exécutent les modifications en réponse à leur formulaire puis renvoient le contrôle à une autre page pour afficher le résultat de l'action (figure 12–3).

Utilisation des classes des paquetages com.eteks.forum et com.eteks.outils

Les pages JSP du forum utilisent les classes du paquetage `com.eteks.forum` déclarées dans les chapitres précédents.

Classe com.eteks.forum.ConnecteurForum

Un objet unique de classe `com.eteks.forum.ConnecteurForum` est utilisé pour partager la connexion à la base de données entre toutes les pages JSP de l'application. Cet objet d'identificateur `connecteurForum` est créé grâce à une balise `jsp:useBean` de portée `application`, qui est déclaré dans le fichier `bean/`

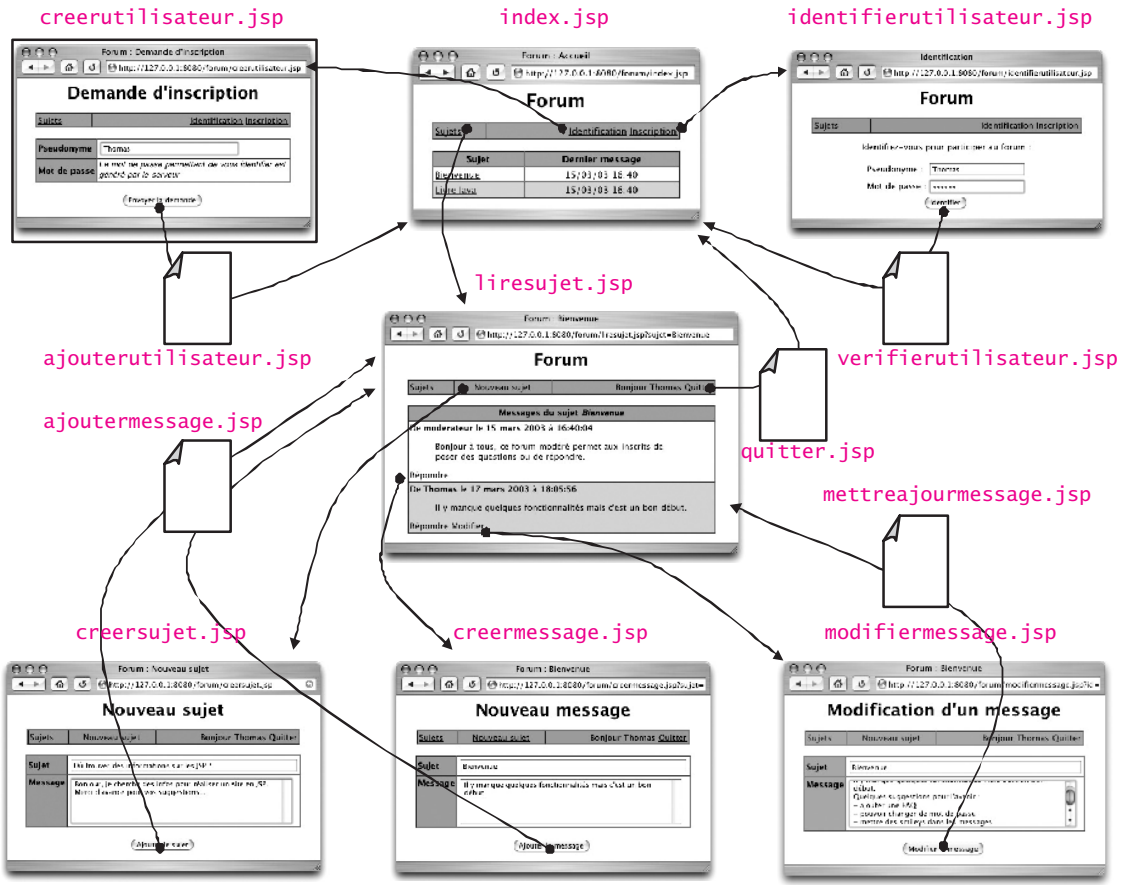


Figure 12–3 Liens entre les pages JSP du forum

`connecteurforum.jsp` ; les pages qui doivent accéder à la base de données incluent ce fichier grâce à une balise `<%@ include ... %>`.

Classe com.eteks.forum.UtilisateurForum

Un objet de classe `com.eteks.forum.UtilisateurForum` est créé pour chaque utilisateur pour la durée de sa session grâce à une balise `jsp:useBean` de portée `session`. Cet objet d'identificateur `utilisateurForum` est déclaré dans le fichier `bean/utilisateurforum.jsp` et permet à toutes les pages JSP de connaître l'utilisateur qui a lancé une requête. Cet objet étant créé, que l'utilisateur se soit identifié ou non, l'application manipule trois types d'utilisateur : les inconnus dont la propriété `autorisation` est `null`, les utilisateurs identifiés dont la propriété `autorisation` n'est pas `null` et le(s) modérateur(s) pour qui la propriété `moderateur` est vraie (cette propriété est obtenue avec la méthode `isModerateur`). Cette classe est aussi utilisée pour créer des objets de portée `page` au moment de l'inscription d'un utilisateur et au moment de l'initialisation de l'objet `connecteurForum` pour inscrire d'office un modérateur dans la base de données.

Classe com.eteks.forum.MessageForum

Cette classe est utilisée dans les pages de saisie d'un message ou d'un sujet pour créer des objets dont la portée est limitée à la page ou la requête.

Classe com.eteks.forum.EnsembleMessagesForum

Cette classe est instanciée dans la page d'accueil et la page présentant les messages d'un sujet pour effectuer des recherches dans la base de données. La liste des messages trouvés est affichée dans un tableau.

Classe com.eteks.outils.OutilsChaine

La méthode `limiterLongueur` de cette classe est appelée pour limiter la longueur du sujet pour les titres des pages. L'autre méthode `convertirEnHTML` est utilisée pour convertir les retours à la ligne et les symboles < des textes des messages.

Classe com.eteks.outils.MotDePasse

La méthode `créer` de cette classe est appelée lors de l'inscription d'un utilisateur pour générer aléatoirement son mot de passe.

Identification de l'utilisateur

La page d'identification est la plus simple du forum : c'est un simple formulaire de saisie avec deux champs, l'un pour le pseudonyme, l'autre pour le mot de passe.

FORUM identifierutilisateur.jsp

```
<html><head><title>Identification</title></head>
<body><center><h1>Forum</h1>

<jsp:include page="include/navigation.jsp" />
<p>Identifiez-vous pour participer au forum :</p>

<form action="verifierutilisateur.jsp" method="post">
  <table border="0">
    <tr>
      <td>Pseudonyme :</td>

      <td><input name="pseudonyme" type="text"
                 maxlength="30"></td>
    </tr>
    <tr>
      <td>Mot de passe :</td>

      <td><input name="motDePasse" type="password"
                 maxlength="30"></td>
    </tr>
  </table>

  <input type="submit" value="Identifier">
</form>
</center></body></html>
```

Inclut la barre de navigation et d'information commune à toutes les pages.

Formulaire d'identification du pseudonyme/mot de passe de l'utilisateur.

Champ de saisie du pseudonyme avec 30 caractères au maximum.

Champ de saisie du mot de passe avec 30 caractères au maximum.

Bouton de confirmation *Identifier*.

La barre de navigation incluse est abordée dans la section suivante, « Page d'accueil ». Le champ `action` du formulaire fait appel à la page JSP `verifierutilisateur.jsp` pour vérifier si le pseudonyme et le mot de passe correspondent aux informations d'un utilisateur existant dans la base de données.

FORUM verifierutilisateur.jsp

```
<%@ page errorPage="erreur.jsp" %>

<%@ include file="bean/connecteurforum.jsp" %>
<%@ include file="bean/utilisateurforum.jsp" %>

<%-- Authentification de l'utilisateur --%>

<jsp:setProperty name="utilisateurForum"
                 property="pseudonyme" param="pseudonyme" />


<% if (   utilisateurForum.rechercher (connecteurForum) 1
      && utilisateurForum.getMotDePasse().equals (
                    request.getParameter("motDePasse")))
   { %>
     <jsp:forward page="index.jsp" /> 2
<% }
   utilisateurForum.setAutorisation(null); %>


<jsp:forward page="identifierutilisateur.jsp"> 3
  <jsp:param name="erreur" value="Identification incorrecte" />
</jsp:forward>
```

- Balise page spécifiant la page d'erreur.
- Inclut les objets décrits dans les fichiers du sous-dossier bean.
- Remarque JSP.
- Modification de la propriété `pseudonyme` de l'objet `utilisateurForum` avec le paramètre de même nom.
- Vérification de l'existence de l'utilisateur dans la base de données et de la correspondance de son mot de passe avec le paramètre `motDePasse`.
- Transfert du contrôle à la page d'accueil.
- Annulation de l'autorisation de l'utilisateur s'il n'existe pas ou si le mot de passe est incorrect.
- Transfert du contrôle à la page `identifierutilisateur.jsp` avec le paramètre `erreur`.

Cette page passe le contrôle à la page d'accueil 2 si l'utilisateur a saisi correctement son pseudonyme et son mot de passe 1 ou sinon renvoie le contrôle à la page précédente 3. Le message d'erreur *Identification incorrecte* spécifié par le paramètre `erreur` est affiché par la barre de navigation.

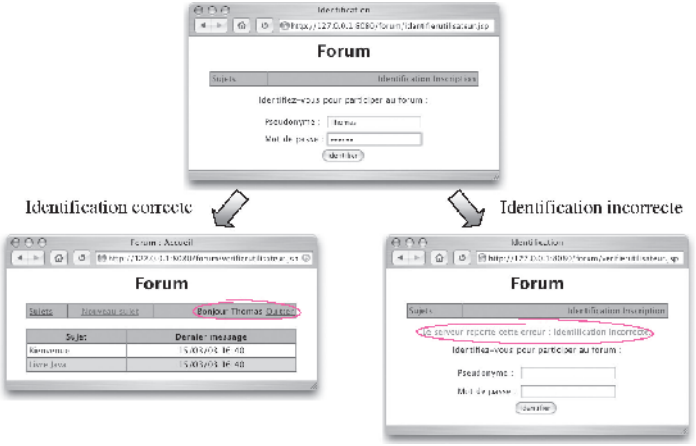


Figure 12–4 Résultat de l'identification avec la page `verifierutilisateur.jsp`

B.A.-BA Tableau HTML

Les tableaux sont très souvent utilisés pour placer des éléments dans des pages HTML. Un élément `<table>` doit inclure autant d'éléments `<tr>` (comme *table row*) que de lignes et chaque cellule est décrite dans un élément `<td>` (comme *table data*) elle-même incluse dans l'élément `<tr>` de sa ligne.

La page verifierutilisateur.jsp vérifie les coordonnées de l'utilisateur dans la base de données en utilisant l'objet connecteurForum de portée application, déclaré dans le fichier bean/connecteurforum.jsp.

FORUM bean/connecteurforum.jsp

Déclaration de l'objet connecteurForum dont la portée s'étend à toute l'application Web.

Au moment de l'initialisation de l'objet connecteurForum, modification de ses propriétés utilisées pour la connexion à la base de données.

Déclaration de l'objet moderateur de portée limitée à cette page.

Attribution d'un pseudonyme, d'un mot de passe et d'une autorisation de modérateur.

Si le modérateur n'est pas déjà dans la base de données, il y est ajouté.

Fin de l'initialisation de l'objet connecteurForum.

```
<jsp:useBean id="connecteurForum" scope="application"
             class="com.eteks.forum.ConnecteurForum">

  <jsp:setProperty name="connecteurForum" (1)
          property="driver" value="com.mysql.jdbc.Driver"/>
  <jsp:setProperty name="connecteurForum"
          property="chaineConnexion" value="jdbc:mysql:///test"/>
  <jsp:setProperty name="connecteurForum"
          property="login" value=""/>
  <jsp:setProperty name="connecteurForum"
          property="password" value=""/>

  <jsp:useBean id="moderateur"
           class="com.eteks.forum.UtilisateurForum"> (2)

    <jsp:setProperty name="moderateur"
                     property="pseudonyme" value="moderateur"/>
    <jsp:setProperty name="moderateur"
                     property="motDePasse" value="azerty"/>
    <jsp:setProperty name="moderateur" property="autorisation"
          value="<%= com.eteks.forum.Utilisateur.MODERATEUR %>"/>

    <% if (!moderateur.rechercher (connecteurForum))
         moderateur.ajouter (connecteurForum); %>
  </jsp:useBean>

</jsp:useBean>
```

Lors de l'instanciation de l'objet, les propriétés de l'objet connecteurForum relatives à la connexion avec la base de données (1) sont initialisées et un modérateur est ajouté à la base de données (2). Les valeurs des propriétés utilisées pour la connexion (1) sont les mêmes que les valeurs par défaut de la classe com.eteks.forum.ConnecteurForum ; elles sont citées pour que vous sachiez où se programme la configuration de la connexion à la base de données si vous souhaitez changer de SGBD pour le forum.

Les instructions qui permettent de créer le modérateur auraient pu aussi être programmées dans une balise <% %>, comme ceci :

```
<% com.eteks.forum.UtilisateurForum moderateur =
     new com.eteks.forum.UtilisateurForum("moderateur", "azerty",
                          com.eteks.forum.Utilisateur.MODERATEUR);
   if (!moderateur.rechercher (connecteurForum))
     moderateur.ajouter (connecteurForum); %>
```

L'objet utilisateurForum représentant l'utilisateur en cours d'identification dans la page verifierutilisateur.jsp est un objet de portée session, déclaré dans le fichier bean/utilisateurforum.jsp.

ATTENTION Initialisation d'un objet de portée application

Un objet de portée application n'est créé et initialisé qu'une seule fois pour toute la durée de vie de l'application Web, c'est-à-dire que les instructions et les balises comprises entre la balise de début <jsp:useBean ...> et la balise de fin </jsp:useBean> ne sont exécutées qu'une fois. Si vous souhaitez modifier ces instructions directement dans le fichier JSP dont se sert le serveur de servlet pour utiliser par exemple un autre SGBD ou pour changer de modérateur, vous devrez alors redémarrer l'application Web. Si vous ne maîtrisez pas les outils d'administrations de Tomcat, arrêtez et redémarrez simplement Tomcat.

FORUM bean/utilisateurforum.jsp

```
<jsp:useBean id="utilisateurForum" scope="session"
             class="com.eteks.forum.UtilisateurForum" />
```

Déclaration de l'objet `utilisateurForum` dont la portée s'étend à la session de l'utilisateur.

La page `erreur.jsp` est appelée au cas où se produiraient des exceptions pendant l'exécution des pages du forum utilisant l'attribut `errorPage="erreur.jsp"` dans leur balise `<%@ page %>`, comme la page `verifierutilisateur.jsp`.

FORUM erreur.jsp

```
<%@ page isErrorPage="true" %>
<html><head><title>Erreur</title></head>
<body><center><h1>Erreur... </h1>
<p>Votre demande n'a pu aboutir.</p>
<p>Merci de signaler les circonstances de cet incident au
webmaster<br>de ce site en lui transmettant le texte d'erreur
qui suit :</p>
<p><b><%= exception %></b></p>
</center></body></html>
```

Balise spécifiant que cette page est une page d'erreur.

Affichage du texte renvoyé par la méthode `toString` de l'exception.

Une page d'erreur reçoit l'exception déclenchée dans la variable prédéfinie `exception`, qui est ici tout simplement convertie en texte avec sa méthode `toString`.

Figure 12–5 Page d'erreur `erreur.jsp` affichée quand la base de données n'est pas démarrée

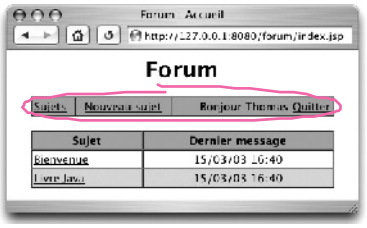
POUR ALLER PLUS LOIN Gestion des erreurs

La gestion des exceptions est simplifiée dans une page JSP car le code Java équivalent à la page est généré d'office dans une instruction `try catch`. En cas d'exception, le bloc `catch` détourne le contrôle d'erreur vers la page d'erreur spécifiée ou vers une page par défaut affichant l'exception. Mais cela ne vous interdit pas de programmer une instruction `try catch` dans la page JSP si vous voulez disposer d'un contrôle plus fin sur l'exception qui est survenue.

Page d'accueil

La page d'accueil affiche dans un tableau la liste de tous les sujets dans l'ordre rétrograde du plus récent au plus ancien.

Utilisateur non identifié



Utilisateur identifié

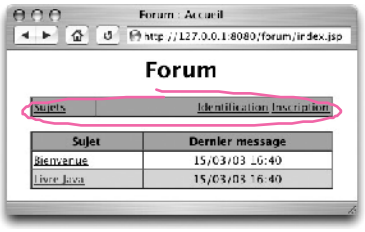


Figure 12–6 Page d'accueil du forum

FORUM index.jsp

```
<%@ page import="java.util.*,java.net.*,java.text.*,
  com.eteks.forum.*,com.eteks.outils.*" errorPage="erreur.jsp" %>
```
Balise page spécifiant les paquetages à importer et la page d'erreur.

```
<%@ include file="bean/connecteurforum.jsp" %>
<html><head><title>Forum : Accueil</title></head>
<body><center><h1>Forum</h1>
```
Inclut l'objet connecteurForum.

```
<jsp:include page="include/navigation.jsp" /><br>
```
Inclut la barre de navigation et d'information commune à toutes les pages.

```
<table border="1" cellpadding="2" cellspacing="0" width="90%">
```
Création d'un tableau HTML avec une bordure occupant 90 % de la largeur de la page.

```
  <tr bgcolor="#9999CC" align="center">
    <td><b>Sujet</b></td>
    <td><b>Dernier message</b></td>
  </tr>
```
Ligne d'en-tête avec les deux colonnes *Sujet* et *Dernier message*, affichée sur un fond bleu.

```
  <% EnsembleMessagesForum sujets = new EnsembleMessagesForum ();
     sujets.rechercherSujets (connecteurForum); 1
```
Recherche des sujets disponibles dans la base de données.

```
     DateFormat dateFormat = DateFormat.getInstance ();
```
Format de date standard.

```
     int ligne = 0;
     for (Iterator it = sujets.iterator (); it.hasNext (); ) 2
     {
       MessageForum sujet = (MessageForum)it.next (); %>
```
Énumération des sujets ligne par ligne.

```
  <tr bgcolor="<%= ligne++ % 2 == 0 ? "#FFFFFF"
                                    : "#CCCCCC" %>"> 3
```
Création d'une ligne de tableau pour chaque sujet.

```
    <td><a href="<%= "liresujet.jsp?sujet="
          + URLEncoder.encode(sujet.getSujet(),
                              response.getCharacterEncoding()) %>"
       ><%= OutilsChaine.convertirEnHTML(
                              sujet.getSujet ()) %></a></td> 4
```
La première colonne contient les sujets avec un lien vers la page liresujet.jsp.

```
    <td align="center"><%= dateFormat.format (
                           sujet.getDateCreation ()) %></td> 5
```
La seconde colonne contient la date du dernier message.

```
  </tr>
  <% } %>
</table>
</center></body></html>
```
Fin de la boucle d'énumération for.

Après avoir recherché la liste des sujets dans la base de données 1, chaque ligne du tableau est générée dans une boucle for qui énumère tous les sujets 2.

DANS LA VRAIE VIE

Surveiller la taille de la page d'accueil

Si votre forum a du succès, les sujets vont finir par être trop nombreux, ce qui rendra votre page d'accueil trop volumineuse. Il sera alors intéressant d'afficher les messages par groupe de 20 par exemple avec des liens pour chaque groupe et d'ajouter un moteur de recherche.

ATTENTION **Conversion des caractères accentués au format x-www-form-urlencoded**

Pour que les lettres accentuées d'un paramètre soient correctement codées au format x-www-form-urlencoded, il faut appeler la méthode encode de la classe java.net.URLEncoder en lui passant en second paramètre le format de codage des caractères de la réponse. La construction de l'URL des liens <A HREF="liresujet.jsp?sujet=..."> utilise ici cette méthode pour coder les sujets des messages grâce à l'expression :

```
URLEncoder.encode(sujet, response.getCharacterEncoding())
```

La méthode encode avec deux paramètres ayant été introduite dans la bibliothèque standard 1.4, le forum de discussion ne peut fonctionner en l'état qu'avec Java 1.4.

Chaque ligne du tableau a deux colonnes, l'une pour le sujet 4, l'autre pour la date du dernier message ajouté à un sujet 5. La couleur de fond des lignes alterne entre le gris et le blanc en testant la parité du numéro de ligne 3.

Le fichier `include/navigation.jsp` contient la barre de navigation qui apparaît sur chaque page.

FORUM include/navigation.jsp

```
<%@ include file="../bean/utilisateurforum.jsp" %> 1

<table border="1" cellpadding="2" cellspacing="0" width="90%">
  <tr bgcolor="#9999CC">


    <td><a href="index.jsp">Sujets</a></td>

      <% if (utilisateurForum.getAutorisation () == null)
         { %>
    <td align="right">
      <a href="identifierutilisateur.jsp">Identification</a> 2
      <a href="creerutilisateur.jsp">Inscription</a>
    </td>
      <% }
         else
         { %>

    <td align="center">
      <a href="creersujet.jsp">Nouveau sujet</a> 3
    </td>

    <td align="right">
      Bonjour <jsp:getProperty name="utilisateurForum"
                               property="pseudonyme" />
      <a href="quitter.jsp">Quitter</a>
    </td>
      <% } %>
  </tr>

</table>

<% if (request.getParameter("erreur") != null) 4
   { %>
 <p><font color="#FF0000">Le serveur reporte cette erreur :
              <%= request.getParameter("erreur") %>.</font></p>
<% }
   if (request.getParameter("information") != null) 5
   { %>
 <p><font color="##33CC00">
          <%= request.getParameter("information") %></font></p>
<% } %>
```

- Inclut l'objet `utilisateurForum`.
- Création d'un tableau HTML d'une ligne avec une bordure occupant 90 % de la largeur de la page.
- Lien vers la page d'accueil.
- Si l'utilisateur n'est pas identifié, affichage des liens *Identification* et *Inscription* alignés à droite dans une cellule.
- Si l'utilisateur est identifié…
- …affichage du lien *Nouveau sujet* centré dans une cellule.
- Affichage d'un message de bienvenue et du lien *Quitter*.
- Fin du tableau de la barre de navigation.
- Si la requête a un paramètre `erreur`, son texte est affiché en rouge, précédé du texte *Le serveur reporte cette erreur*:
- Si la requête a un paramètre `information`, son texte est affiché en vert.

Une partie de la barre de navigation étant différente selon que l'utilisateur s'est identifié ou non, l'objet `utilisateurForum` 1 de portée `session` est utilisé pour déterminer s'il faut afficher les liens Identification /Inscription 2 ou les liens Nouveau sujet /Quitter 3. Enfin, ce fichier prend en charge l'affichage des messages contenus dans les paramètres `erreur` 4 et `information` 5.

Inscription d'un utilisateur

La page d'inscription est un simple formulaire de saisie comportant un champ pour saisir un nouveau pseudonyme.

FORUM creerutilisateur.jsp

```
<html><head><title>Forum : Demande d'inscription</title></head>
<body><center><h1>Demande d'inscription</h1>
<jsp:include page="include/navigation.jsp" /><br>

<form action="ajouterutilisateur.jsp" method="post">

  <table width="90%" border="1" cellpadding="2" cellspacing="0">

    <tr>
      <td bgcolor="#9999CC"><b>Pseudonyme</b></td>
      <td><input name="pseudonyme" type="text"
                 size="30" maxlength="30" ></td>
    </tr>
    <tr>
      <td bgcolor="#9999CC"><b>Mot de passe</b></td>
      <td> <i>Le mot de passe permettant de vous identifier
        est g&eacute;n&eacute;r&eacute; par le serveur</i></td>
    </tr>
  </table>
  <br><input type="submit" value="Envoyer la demande">
</form></center></body></html>
```

Inclut la barre de navigation.

Formulaire de saisie de l'utilisateur.

Création d'un tableau HTML.

Texte *Pseudonyme* et champ de saisie du pseudonyme sur 30 colonnes avec 30 caractères au maximum.

Texte d'information sur la génération du mot de passe par le serveur.

Bouton de confirmation *Envoyer la demande*.

Le champ `action` du formulaire fait appel à la page JSP `ajouterutilisateur.jsp` pour vérifier si le pseudonyme saisi est correct, puis ajouter le nouvel utilisateur dans la base de données.

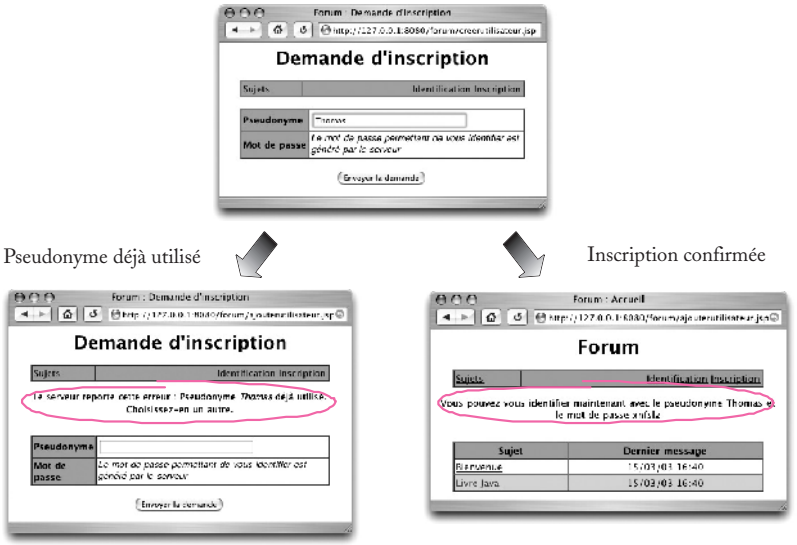


Figure 12–7 Résultat de l'inscription avec la page `ajouterutilisateur.jsp`

DANS LA VRAIE VIE

Envoi du mot de passe par e-mail

La plupart des forums d'Internet vous confirmeront votre inscription par e-mail, par sécurité, et afin que vous puissiez garder une trace de votre pseudonyme /mot de passe pour un usage futur. En ajoutant un champ pour saisir l'e-mail de l'utilisateur dans la page d'inscription, vous pourrez réaliser cette opération dans la page `ajouterutilisateur.jsp` avec la bibliothèque JavaMail™ développée par Sun Microsystems et incluse dans Tomcat

- http://java.sun.com/products/javamail

FORUM ajouterutilisateur.jsp

```
<%@ page import="com.eteks.forum.*,com.eteks.outils.*"
         errorPage="erreur.jsp" %>

<%@ include file="bean/connecteurforum.jsp" %>

<jsp:useBean id="utilisateur"
       class="com.eteks.forum.UtilisateurForum" >
  <jsp:setProperty name="utilisateur" property="*"/>
</jsp:useBean>

<% if (utilisateur.getPseudonyme () == null) 1
   { %>

  <jsp:forward page="creerutilisateur.jsp"> 2
    <jsp:param name="erreur"
               value="Vous devez choisir un pseudonyme" />
  </jsp:forward>
<% } %>

<% if (utilisateur.rechercher (connecteurForum)) 3
   { %>

  <jsp:forward page="creerutilisateur.jsp"> 4
    <jsp:param name="erreur" value="<%= "Pseudonyme <i>"
           + utilisateur.getPseudonyme ()
           + "</i> d&eacute;j&agrave; utilis&eacute;.<br>"
           + "Choisissez-en un autre" %>" />
  </jsp:forward>
<% } %>

<% utilisateur.setAutorisation (Utilisateur.UTILISATEUR);

   utilisateur.setMotDePasse (MotDePasse.creer ());

   utilisateur.ajouter (connecteurForum); %> 5

<jsp:forward page="index.jsp">
  <jsp:param name="information" value="<%= "Vous pouvez vous"
         + " identifier maintenant avec le pseudonyme "
         + utilisateur.getPseudonyme () + " et le mot de passe "
         + utilisateur.getMotDePasse () %>" /> 6
</jsp:forward>
```

- Balise page spécifiant les paquetages à importer et la page d'erreur.
- Inclut l'objet `connecteurForum`.
- Déclaration de l'objet `utilisateur` de portée limitée à cette page et initialisé avec les paramètres (ici `pseudonyme`).
- Vérification que le pseudonyme de l'utilisateur n'est pas `null` (si aucune valeur n'a été saisie).
- Transfert du contrôle à la page `creerutilisateur.jsp` avec le paramètre `erreur`.
- Vérification que le pseudonyme saisi par l'utilisateur n'existe pas déjà dans la base de données.
- Transfert du contrôle à la page `creerutilisateur.jsp` avec le paramètre `erreur` décrivant que le pseudonyme saisi existe déjà.
- Attribution de l'autorisation d'utilisateur.
- Attribution d'un mot de passe aléatoire.
- Ajout de l'utilisateur à la base de données.
- Transfert du contrôle à la page d'accueil avec le paramètre `information` contenant une information sur son pseudonyme /mot de passe.

Cette page repasse le contrôle à la page d'inscription 2 4 si l'utilisateur n'a pas saisi de pseudonyme 1 ou si le pseudonyme a déjà été choisi 3. Sinon, le nouvel utilisateur est ajouté à la base de données 5 puis le contrôle est passé à la page d'accueil avec un message communiquant au nouveau membre du forum le mot de passe qui lui a été attribué 6.

ASTUCE Utilisation du symbole *

Utilisez le plus souvent possible la balise `jsp:setProperty` avec un attribut `property` égal à * pour initialiser toutes les propriétés d'un objet avec les paramètres envoyés par un formulaire. Même si, dans un premier temps, cette balise n'initialise qu'une seule propriété comme c'est le cas ici, ce procédé se révélera pratique pour faire évoluer le site au moment où vous rajouterez de nouvelles propriétés et leurs paramètres ; vous n'aurez pas à y retoucher !

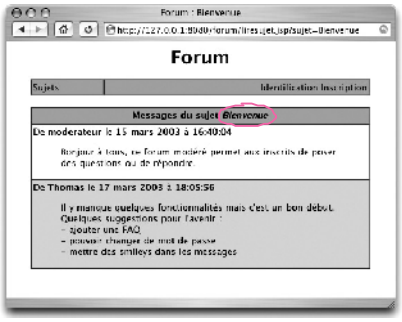
ATTENTION La balise jsp:setProperty ignore les paramètres vides

La balise `jsp:setProperty` n'a aucun effet pour les paramètres vides (par exemple pour `pseudonyme=`) et n'appelle donc pas le mutateur de la propriété avec la valeur `null` ou la chaîne vide `""`.

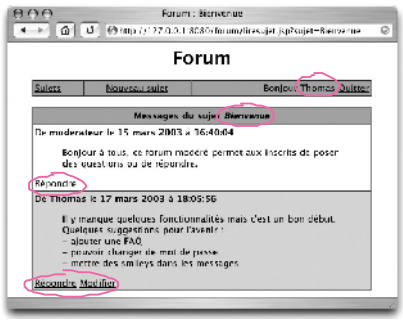
Messages d'un sujet

La page `liresujet.jsp` affiche dans un tableau la liste de tous les messages d'un sujet donné dans l'ordre chronologique.

Utilisateur non identifié



Utilisateur *Thomas*



Modérateur

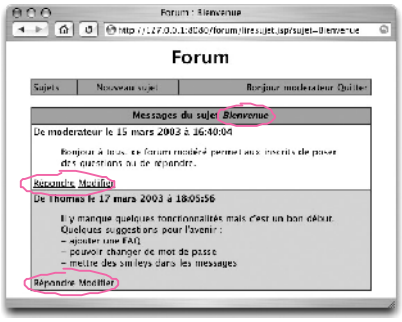


Figure 12–8 Liste des messages du sujet Bienvenue

FORUM liresujet.jsp

Balise page spécifiant les paquetages à importer et la page d'erreur.

```
<%@ page import="java.util.*,java.net.*,java.text.*,
  com.eteks.forum.*,com.eteks.outils.*" errorPage="erreur.jsp" %>
```

Inclut les objets `connecteurForum` et `utilisateurForum`.

```
<%@ include file="bean/connecteurforum.jsp" %>
<%@ include file="bean/utilisateurforum.jsp" %>
```

Récupération du paramètre `sujet`.

```
<% String sujet = request.getParameter("sujet"); %> (1)
```

Crée un titre reprenant les 50 premiers caractères du sujet.

```
<html><head><title>Forum : <%= OutilsChaine.convertirEnHTML(
     OutilsChaine.limiterLongueur (sujet, 50)) %></title></head>
<body><center><h1>Forum</h1>
```

Inclut la barre de navigation.

```
<jsp:include page="include/navigation.jsp" /><br>
```

Création d'un tableau HTML.

```
<table border="1" cellpadding="2" cellspacing="0" width="90%">
```

Ligne d'en-tête reprenant le sujet en gras et italique.

```
  <tr bgcolor="#9999CC">
    <td align="center"><b>Messages du sujet
      <i><%= OutilsChaine.convertirEnHTML(sujet) %></i></b></td>
  </tr>
```

Recherche des messages du sujet dans la base de données.

```
  <% EnsembleMessagesForum messages= new EnsembleMessagesForum();
     messages.rechercherMessagesSujet(connecteurForum, sujet); (2)
```

Format de date et d'heure standard.

```
     DateFormat formatDate  = DateFormat.getDateInstance ();
     DateFormat formatHeure = DateFormat.getTimeInstance ();
     int ligne = 0;
```

```
    for (Iterator it = messages.iterator (); it.hasNext(); ) 3
    {
      MessageForum message = (MessageForum)it.next(); %>
  <tr bgcolor="<%= ligne++ % 2 == 0 ? "#FFFFFF" : "#CCCCCC" %>">
    <td>De <b><%= message.getAuteur () %></b> 4
        le <b><%= formatDate.format (
                        message.getDateCreation()) %></b> 5
        &agrave; <b><%= formatHeure.format(
                        message.getDateCreation ()) %></b> 6
    <blockquote><%= OutilsChaine.convertirEnHTML(
                      message.getTexte()) %></blockquote> 7
    <% if (utilisateurForum.getAutorisation () != null) 8
       { %>
      <a href="<%= "creermessage.jsp?sujet=" + URLEncoder.encode(
                      sujet, response.getCharacterEncoding()) %>"
       >R&eacute;pondre</a> 9
      <% if (   utilisateurForum.isModerateur ()
             || message.estEcritPar (utilisateurForum)) 10
         { %>
        <a href="<%= "modifiermessage.jsp?id="
                     + message.getId() %>">Modifier</a> 11
      <% } %>
    <% } %></td>
  </tr>
  <% } %>
</table>
</center></body></html>
```

- Énumération des messages du sujet ligne par ligne.
- Création d'une ligne de tableau par message.
- Génération du texte.
  De *auteur* le *jj mmm aaaa* à *hh:mm.*
- Affichage du texte du message en retrait.
- Si l'utilisateur est identifié, ajout du lien *Répondre.*
- Si l'utilisateur est un modérateur ou si l'utilisateur est l'auteur du message, ajout du lien *Modifier.*
- Fin de la boucle d'énumération `for`.

Après avoir recherché dans la base de données 2 la liste des messages du sujet passé en paramètre 1, chaque ligne du tableau est générée dans une boucle `for` qui énumère tous les messages 3. Chaque ligne du tableau affiche l'auteur 4, la date 5, l'heure 6 et le texte 7 d'un message, et ajoute les liens Répondre 9 et Modifier 11 en fonction de l'autorisation de l'utilisateur 8 10.

Création de sujet, de message, et modification

Figure 12–9 Pages utilisant le formulaire de saisie d'un message

Pages de saisie

Les trois pages `creersujet.jsp`, `creermessage.jsp` et `modifiermessage.jsp` sont décrites ensemble car elles utilisent le même formulaire de saisie du fichier `include/formulairemessage.jsp`. C'est l'occasion d'aborder la mise en œuvre d'une balise `jsp:include` avec des paramètres `jsp:param`.

FORUM creersujet.jsp

```
<html><head><title>Forum : Nouveau sujet</title></head>
<body><center><h1>Nouveau sujet</h1>
<jsp:include page="include/navigation.jsp" /><br> 1

<jsp:include page="include/formulairemessage.jsp" > 2
  <jsp:param name="actionFormulaire" value="ajoutermessage.jsp"/>
  <jsp:param name="labelSubmit"      value="Ajouter le sujet" />
</jsp:include>
</center></body></html>
```

Inclut la barre de navigation.

Inclut le formulaire de saisie du message avec les paramètres `actionFormulaire` et `labelSubmit`.

La page de création d'un sujet assemble la barre de navigation 1 et le formulaire de saisie d'un message 2 auxquels elle transmet l'action de formulaire à utiliser et le label du bouton de confirmation.

FORUM creermessage.jsp

```
<%@ page import="com.eteks.outils.*" errorPage="erreur.jsp" %>


<jsp:useBean id="message" class="com.eteks.forum.MessageForum"
             scope="request"> 1
 <jsp:setProperty name="message" property="sujet" param="sujet"/>
</jsp:useBean>
<html><head><title>Forum : <%= OutilsChaine.convertirEnHTML(
      OutilsChaine.limiterLongueur (message.getSujet (), 50)) %>
</title></head><body><center><h1>Nouveau message</h1>
<jsp:include page="include/navigation.jsp" /><br>

<jsp:include page="include/formulairemessage.jsp" > 2
  <jsp:param name="actionFormulaire" value="ajoutermessage.jsp"/>
  <jsp:param name="labelSubmit" value="Ajouter le message" />
</jsp:include>
</center></body></html>
```

Balise page spécifiant le paquetage à importer et la page d'erreur.

Déclaration de l'objet `message` de portée `request` et initialisation de sa propriété `sujet` avec le paramètre de même nom.

Création d'un titre reprenant les 50 premiers caractères du sujet.

Inclut la barre de navigation.

Inclut le formulaire de saisie du message avec les paramètres `actionFormulaire` et `labelSubmit`.

La page de création d'un message crée l'objet `message` de portée `request` 1 initialisé avec le sujet en paramètre. Cet objet est utilisé au cours de la même requête pour initialiser le champ sujet du formulaire de message inclus 2. Vous noterez que les pages `creersujet.jsp` et `creermessage.jsp` appellent la même page `ajoutermessage.jsp` dans l'action du formulaire.

JAVA Utilisation des objets de portée request

Comme un objet de portée `request` existe pendant toute la durée d'une requête HTTP sur le serveur, ce type d'objet est intéressant pour créer et initialiser un objet qui est destiné à être réutiliser dans d'autres pages JSP appelées avec les balises `jsp:include` et `jsp:forward`.

FORUM modifiermessage.jsp

```
<%@ page import="com.eteks.outils.*" errorPage="erreur.jsp" %>

<%@ include file="bean/connecteurforum.jsp" %>

<jsp:useBean id="message" class="com.eteks.forum.MessageForum"
             scope="request"> (1)
  <jsp:setProperty name="message" property="id" param="id"/>

  <% message.rechercher (connecteurForum); %> (2)
</jsp:useBean>
<html><head><title>Forum : <%= OutilsChaine.convertirEnHTML(
      OutilsChaine.limiterLongueur (message.getSujet (), 50)) %>
</title></head><body><center><h1>Modification d'un message</h1>
<jsp:include page="include/navigation.jsp" /><br>

<jsp:include page="include/formulairemessage.jsp" > (3)
  <jsp:param name="actionFormulaire" value=
     "<%= "mettreajourmessage.jsp?id=" + message.getId () %>" />
  <jsp:param name="labelSubmit" value="Modifier le message" />
</jsp:include>
</center></body></html>
```

- Balise page spécifiant le paquetage à importer et la page d'erreur.
- Inclut l'objet `connecteurForum`.
- Déclaration de l'objet `message` de portée `request` et initialisation de sa propriété `id` avec le paramètre de même nom.
- Recherche du message avec son `id`.
- Création d'un titre reprenant les 50 premiers caractères du sujet.
- Inclut la barre de navigation.
- Inclut le formulaire de saisie du message avec les paramètres `actionFormulaire` et `labelSubmit`.

La page de modification d'un message crée l'objet `message` de portée `request` (1). Cet objet est initialisé avec la valeur du paramètre `id` pour rechercher les données du message en cours de modification dans la base de données (2). Cet objet est utilisé au cours de la même requête pour initialiser les champs sujet et texte du formulaire de message inclus (3).

FORUM include/formulairemessage.jsp

```
<jsp:useBean id="message" class="com.eteks.forum.MessageForum"
             scope="request"/> (1)

<form action="<%= request.getParameter ("actionFormulaire") %>"
      method="post">

  <table border="1" cellpadding="2" cellspacing="0" width="90%">
    <tr><td bgcolor="#9999CC"><b>Sujet</b></td>
        <td><input name="sujet" type="text" size="50"
             maxlength="255"

             value="<jsp:getProperty name="message"
                                property="sujet" />"></td></tr> (2)

    <tr><td valign="top" bgcolor="#9999CC"><b>Message</b></td>

        <td><textarea name="texte" cols="50" rows="5"


          ><jsp:getProperty name="message"
                        property="texte" /></textarea></td></tr> (3)
  </table>
  <br><input type="submit"
          value="<%= request.getParameter("labelSubmit") %>">
</form>
```

- Déclaration de l'objet `message` de portée `request`.
- Formulaire de saisie du message dont l'action est égale au paramètre `actionFormulaire`.
- Création d'un tableau HTML.
- Champ de saisie du sujet sur 50 colonnes avec 255 caractères au maximum.
- Initialisation du texte du champ de saisie avec la propriété `sujet` de l'objet `message`.
- Zone de saisie du texte sur 50 colonnes et 5 lignes.
- Initialisation du texte du champ de saisie avec la propriété `texte` de l'objet `message`.
- Bouton de confirmation dont le label est égal au paramètre `labelSubmit`.

ASTUCE jsp:getProperty n'a pas d'effet pour une propriété null

La balise `jsp:getProperty` affiche un texte vide si la valeur de la propriété est égale à `null`. Souvent cela évite de tester la valeur d'une propriété affichée, comme dans notre cas pour les deux balises `<jsp:getProperty name="message" property="sujet" />` et `<jsp:getProperty name="message" property="texte" />`.

Le formulaire de saisie d'un message affiche deux champs de saisie initialisés avec les propriétés `sujet` 2 et `texte` 3 de l'objet `message` 1. Si l'objet `message` de portée `request` n'existe pas déjà, il est créé (c'est le cas quand le formulaire est inclus par la page `creersujet.jsp`), sinon il est récupéré tel qu'il était avant l'appel à cette page (c'est le cas pour les pages `creermessage.jsp` et `modifiermessage.jsp`).

ATTENTION Initialisation d'un champ multiligne

Si vous désirez initialiser une balise multilignes `textarea`, veillez à bien coller au texte d'initialisation le dernier caractère > de la balise de début et le premier caractère < de la balise de fin pour ne pas insérer des espaces superflus.

Pages d'ajout et de modification de message

La page `ajoutermessage.jsp` ajoute un message à la base de données.

FORUM ajoutermessage.jsp

Balise page spécifiant le paquetage à importer et la page d'erreur.

Inclut les objets `connecteurForum` et `utilisateurForum`.

Déclaration de l'objet `message` de portée limitée à cette page et initialisé avec les paramètres (ici `sujet` et `texte`).

Modification de l'auteur.

Ajout du message dans la base de données.

Envoie au navigateur l'ordre de recharger la page `liresujet.jsp`.

```
<%@ page import="java.net.*" errorPage="erreur.jsp" %>


<%@ include file="bean/connecteurforum.jsp" %>
<%@ include file="bean/utilisateurforum.jsp" %>

<jsp:useBean id="message" class="com.eteks.forum.MessageForum" >
  <jsp:setProperty name="message" property="*"/> 1


  <% message.setAuteur (utilisateurForum); %> 2
</jsp:useBean>
<% message.ajouter (connecteurForum); %> 3

<% response.sendRedirect("liresujet.jsp?sujet="
        + URLEncoder.encode(message.getSujet(),
                            response.getCharacterEncoding())); %>
```

Un objet `message` est créé et initialisé avec le sujet et le texte en paramètres 1. L'auteur du message est ensuite déterminé grâce à l'objet de session `utilisateurForum` associé à chaque utilisateur 2. Enfin, le message est ajouté à la base de données 3.

La page `mettreajourmessage.jsp` est appelée pour modifier un message dans la base de données.

FORUM mettreajourmessage.jsp

Balise page spécifiant le paquetage à importer et la page d'erreur.

Inclut l'objet `connecteurForum`.

Déclaration de l'objet `message` de portée limitée à cette page et initialisation de sa propriété `id` avec le paramètre de même nom.

Lecture du message avec son `id`.

```
<%@ page import="java.net.*" errorPage="erreur.jsp" %>


<%@ include file="bean/connecteurforum.jsp" %>

<jsp:useBean id="message" class="com.eteks.forum.MessageForum" >
  <jsp:setProperty name="message" property="id" param="id"/> 1


  <% message.rechercher(connecteurForum); %> 2
```

```
  <jsp:setProperty name="message" property="*"/> 3
</jsp:useBean>

<% message.mettreAJour (connecteurForum); %> 4


<% response.sendRedirect("liresujet.jsp?sujet="
        + URLEncoder.encode(message.getSujet(),
                            response.getCharacterEncoding())); %>
```

- Modification des propriétés de l'objet `message` avec les paramètres (ici `sujet` et `texte`).
- Mise à jour du message dans la base de données.
- Envoie au navigateur l'ordre de recharger la page `liresujet.jsp`.

Un objet `message` est créé et initialisé avec la valeur du paramètre `id` 1 pour lire les données inchangées du message dans la base de données 2 avant de modifier son sujet et son texte 3. Enfin, le message est mis à jour dans la base de données 4.

DANS LA VRAIE VIE **Vérification des paramètres et de l'utilisateur**

Les pages `ajoutermessage.jsp` et `mettreajourmessage.jsp` ne vérifient pas les paramètres qu'elles reçoivent par souci de simplification. Dans les faits, ces pages devraient vérifier que le texte des paramètres `sujet` et `texte` n'est pas vide pour le cas échéant le signaler à l'utilisateur. De même, les pages de création et de modification de message ne vérifient pas si l'utilisateur en cours est bien identifié. En principe, l'accès à ces pages se fait par des liens visibles uniquement par un utilisateur identifié, mais n'importe qui peut en fait y accéder en saisissant directement leur URL dans le champ d'adresse d'une page d'un navigateur. Cette vérification pourrait se programmer en renvoyant le contrôle à la page d'accueil si la propriété `autorisation` de l'objet `utilisateurForum` est `null`.

Quitter l'application

Cette dernière page annule la propriété `autorisation` de l'objet `utilisateurForum` et repasse le contrôle à la page d'accueil.

FORUM `quitter.jsp`

```
<%@ include file="bean/utilisateurforum.jsp" %>

<% utilisateurForum.setAutorisation(null); %>

<jsp:forward page="index.jsp" />
```

- Inclut l'objet `utilisateurForum`.
- Annulation de l'autorisation de l'utilisateur.
- Transfert du contrôle à la page d'accueil.

ATTENTION **Pas de balise `jsp:forward` dans une page incluse**

N'utilisez pas de balise `jsp:forward` dans une page incluse avec la balise `<jsp:include page="page.jsp" />`.
Cela provoque bien l'arrêt de la page `page.jsp` mais la page JSP qui a appelé `<jsp:include page="page.jsp" />` poursuivra quant à elle son exécution !

REGARD DU DÉVELOPPEUR **jsp:forward ou sendRedirect, que choisir ?**

La dernière balise des pages JSP `ajoutermessage.jsp` et `mettreajourmessage.jsp` utilise la méthode `sendRedirect` avec l'objet `response`, et non une balise `jsp:forward` comme dans les autres pages.

Cette méthode envoie l'ordre au navigateur de recharger la page passée en paramètre ; elle est donc moins rapide qu'une balise `jsp:forward` puisqu'une nouvelle requête doit être lancée par le client.

Dans notre cas, nous préférons la méthode `sendRedirect` et ce pour deux raisons :

- Le navigateur de l'utilisateur affichera l'URL de la page `liresujet.jsp` dans le champ d'adresse du navigateur, lui permettant de bien comprendre que son message a été ajouté au sujet.
- Si l'utilisateur appelle le menu Recharger la page de son navigateur, la page `liresujet.jsp` sera relue sans ajouter à nouveau le même message.

REGARD DU DÉVELOPPEUR **Swing ou JSP, que choisir ?**

Une fois vos composants de base créés (dans notre cas, ceux du paquetage `com.eteks.forum`), les arguments suivants guideront votre choix entre les technologies JSP et Swing pour créer l'interface utilisateur de votre application (si vous avez le choix !) :

- Une application Swing est généralement plus ergonomique grâce à un plus grand choix de composants et à l'utilisation de menus et de raccourcis clavier. De plus, la mise en page des fenêtres et leur ordre d'affichage sont mieux contrôlés.
- Une application JSP est plus simple à diffuser puisqu'elle ne nécessite aucune installation sur le poste du client à part celle d'un navigateur. Une applet Swing est elle aussi simple à télécharger mais nécessite tout de même l'installation du JRE sur le poste du client.
- Une application JSP étant de fait ralentie par les accès au serveur, une application Swing est plus adéquate quand le client a besoin d'un programme réactif, comme une calculatrice ou un programme de dessin, ou tout simplement si l'utilisateur ne peut pas se connecter à un réseau.
- L'interface utilisateur d'une application JSP est plus simple à développer et permet à des non-programmeurs (infographistes, rédacteurs) d'intervenir sur la conception des pages moyennant un minimum de formation.

En résumé...

Ce chapitre vous a présenté les différentes façons d'exploiter les pages JSP sur un serveur. La présentation du forum en JSP a permis de passer en revue les principales utilisations des balises JSP en mettant particulièrement l'accent sur la mise en œuvre des composants JavaBeans avec leurs différentes portées. Vous aurez sûrement remarqué en lisant ces lignes que la programmation d'une interface utilisateur selon que l'on utilise les pages JSP ou Swing diffère du tout au tout...