Struts Dialogs Mail Reader: Subscriptions

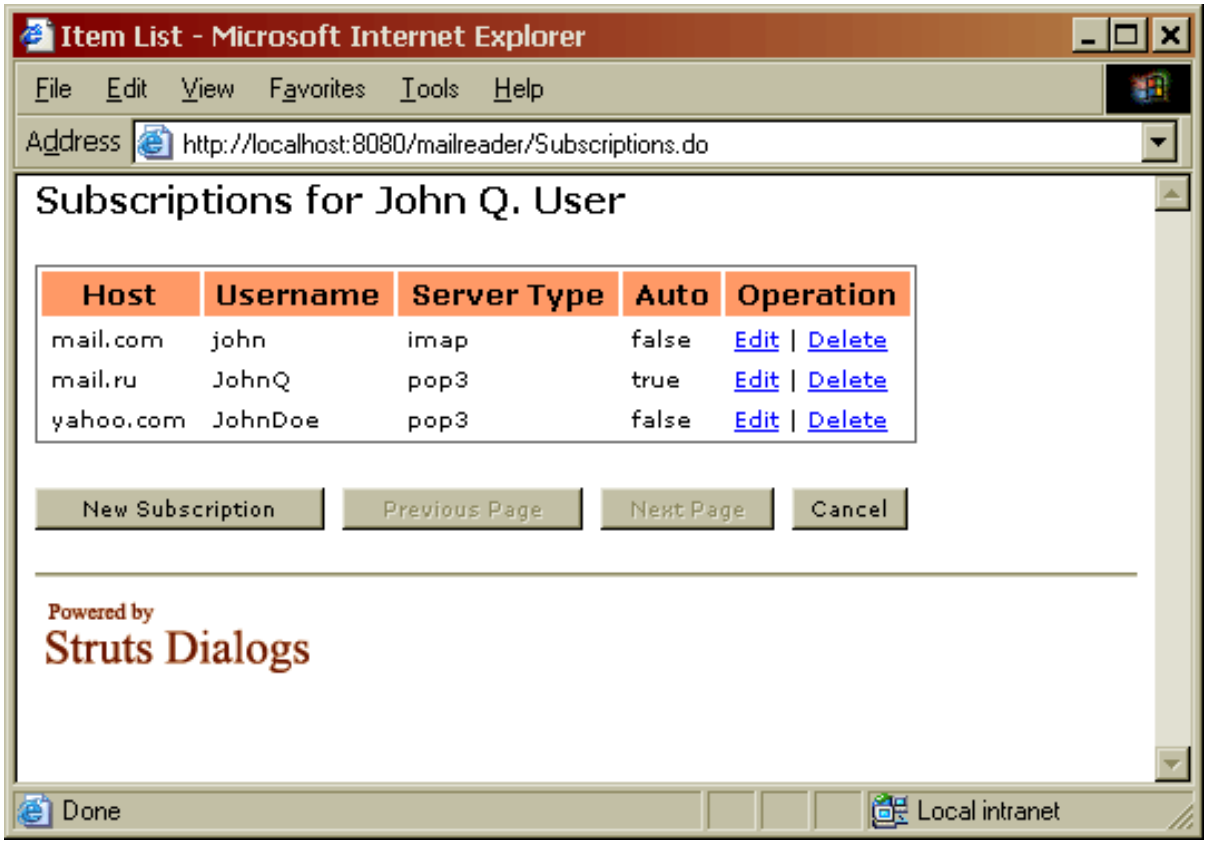
1. Overview

Subscriptions component performs the following functions:

- Displays a list of email subscriptions
- Allows to create a new sibscription
- Allows to update existing subscription

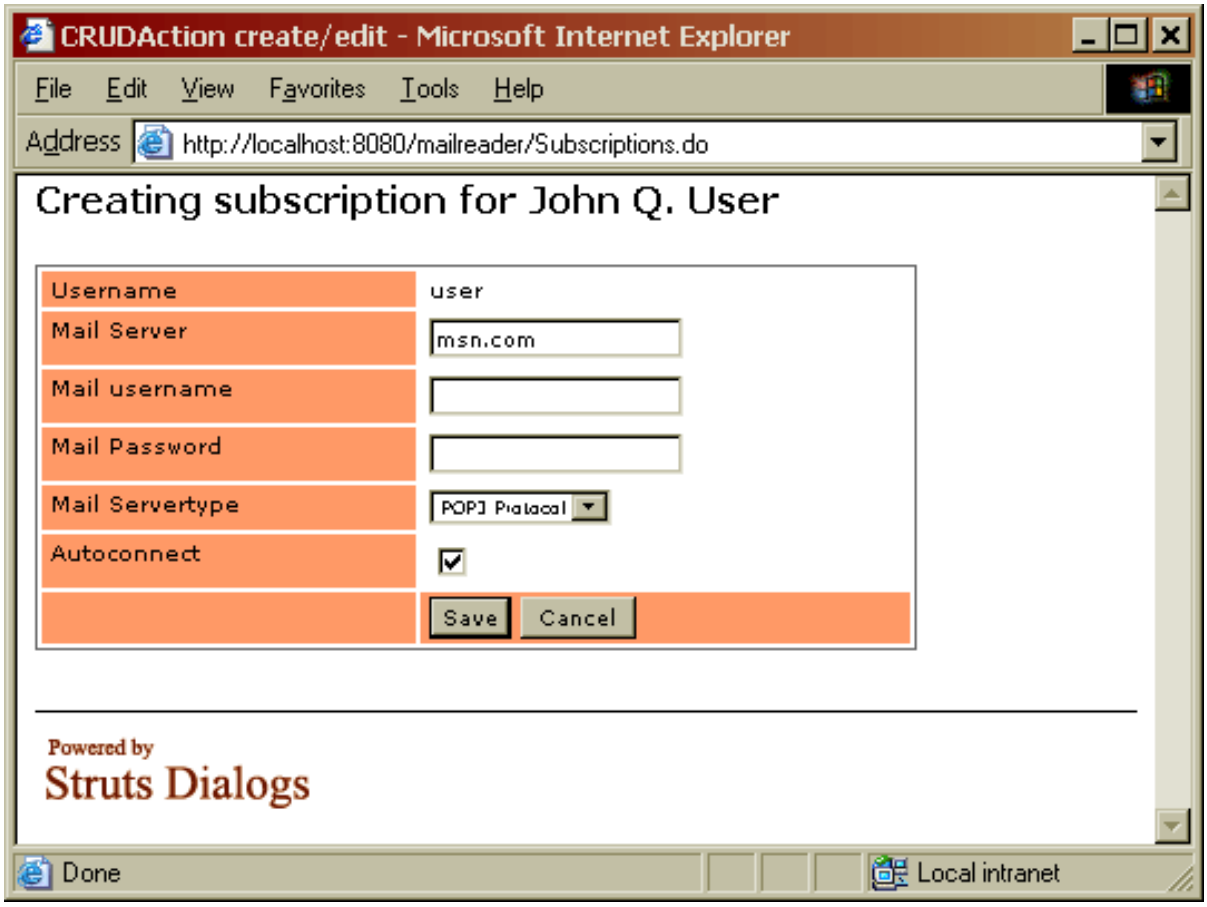
Subscriptions component is controlled by one action (SubscriptionAction.java) and has two views: `subscriptions.jsp` and `subscription.jsp`.

`subscriptions.jsp` displays a list of email subscriptions. It is shown to a logged-in user, if there is no *current* subscription.



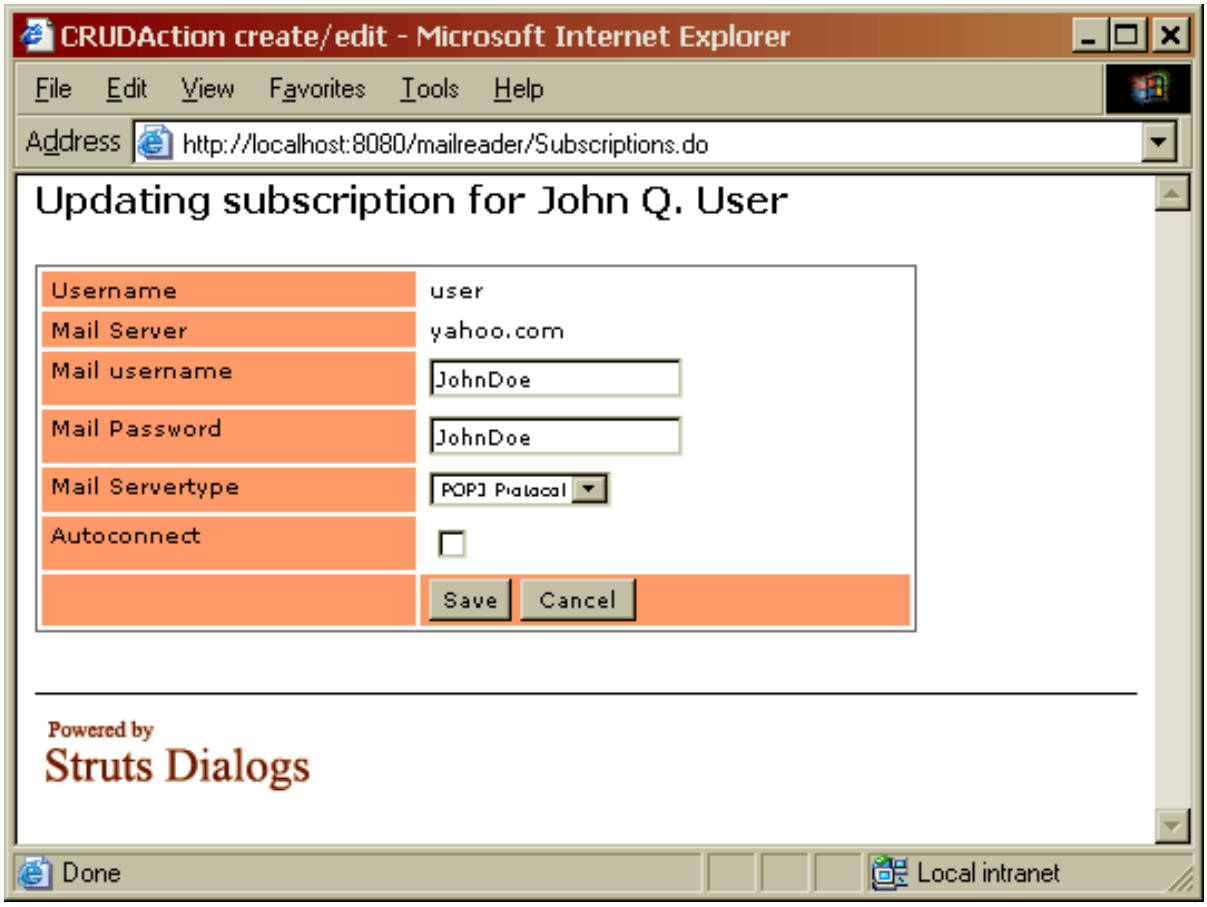
MailReader Subscriptions

If current subscription exists, `subscription.jsp` displays it to a logged-in user. It can be either a new subscription...



MailReader Create Subscription

...or an existing subscription:



MailReader Update Subscription

2. SubscriptionAction.java

`SubscriptionAction` class does not differ much from standard all-in-one CRUD Component. The major difference is that Subscription action first verifies if a user is logged in. If not, it redirects to Login action. If user is logged in, then Subscription action handles input events in standard way for DialogAction class. In real application this code should be placed in an interceptor or servlet filter. Struts Chains, introduced in Struts 1.3, should allow easy implementation of generic interceptors like login interceptor.

```
public class SubscriptionAction extends CRUDAction {
  ...
  public ActionForward execute(ActionMapping mapping,
                               ActionForm form,
                               HttpServletRequest request,
                               HttpServletResponse response)
```

```
    throws Exception {

    // If user not found, redirect to login action
    HttpSession session = request.getSession();
    if (session.getAttribute(Constants.USER_KEY) == null) {
      return mapping.findForward("logon");
    }

    // User is valid, perform standard event processing
    return super.execute(mapping, form, request, response);
  }
  ...
}
```

3. Subscriptions list

If *current* subscription does not exist, Subscrpiptions action displays subscrpiption list. This list can be split into pages. Default page size is four rows. Struts-EL is used to access actionform data.

```
<table>
<thead>
  <tr>
    <th>Host</th>
    <th>Username</th>
    <th>Server Type</th>
    <th >Auto</th>
    <th>Operation</th>
  </tr>
</thead>

<logic:present name="user" scope="session">
  <!-- Show subscription list -->
  <logic-el:iterate id="subscription"
                    collection="${SubscriptionForm.subscriptions}"
                    offset="${SubscriptionForm.offset}"
                    length="${SubscriptionForm.pagesize}"
                    type="net.jspcontrols.mailreader.business.Subscription">
    <tr>
      <td><c:out value="${subscription.host}"/></td>
      <td><c:out value="${subscription.username}"/></td>
      <td><c:out value="${subscription.type}"/></td>
      <td><c:out value="${subscription.autoConnect}"/></td>
      <td>
        <html-el:link
          href="Subscriptions.do?DIALOG-EVENT-UPDATE&host=${subscription.host}">
          Edit
        </html-el:link>
        <html-el:link
          href="Subscriptions.do?DIALOG-EVENT-DELETE&host=${subscription.host}">
          Delete
        </html-el:link>
```

```
      </td>
    </tr>
  </logic-el:iterate>
</logic:present>
</table>

<html:form action="/Subscriptions.do">
  <html:submit property="DIALOG-EVENT-CREATE" value="New Subscription"/>
  <html-el:submit property="DIALOG-EVENT-PREVPAGE" value="Previous Page"
                  disabled="${SubscriptionForm.firstpage}"/>
  <html-el:submit property="DIALOG-EVENT-NEXTPAGE" value="Next Page"
                  disabled="${SubscriptionForm.lastpage}"/>
  <html:submit property="DIALOG-EVENT-BACKHOME" value="Cancel"/>
</html:form>
```

4. Creating and updating subscriptions

The major difference of this version of MailReader application from original Ted Husted's version is that action form has session scope and current subscription is nested within the action form. Fortunately, Subscrption business object (BO) has only String and boolean fields, so no conversion is needed and it is easy to retain invalid values in the form.

When a user creates new subscription, a new `Subscription` object is created and nested in the `SubscriptionForm.java` class. If a user wants to update existing subscription, then it is cloned into a detached object, which in turn, is nested within `SubscriptionForm.java` class. Therefore, changes to nested subscription do not affect persistent data. If the user cancels the update process, then detached subscription is simply deleted from memory.

Subscription action creates new or updates existing subscription based on input event generated by button on `subscriptions.jsp` page. DIALOG-EVENT-CREATE creates new subscription, DIALOG-EVENT-UPDATE selects a subscription as current and updates it.

5. Next: Registration component

Proceed to Registration component.