Tempo

Details of Alfresco SSO integration in liferay

Added by Ark Xu, last edited by Ark Xu on Sep 16, 2008

Integrate Alfresco with Liferay, along with tempo

The basic idea of Alfresco portlet user authentication is to check whether the attribute:

- AuthenticationHelper.AUTHENTICATION_USER exists in the PortletSession.

If it does exist, it will get the user in it and validate the user in Alfresco user management service.

Tempo has a new portlet class to replace the original one. The code for AlfrescoFacesPortlet.java is in tempo svn.

First, we retrieve the user from the Liferay session, if there is user logged in from CAS, we can find it by:

```
Method getHttpServletRequest = request.getClass().getMethod("getHttpServletRequest");
HttpServletRequest hsr = (HttpServletRequest) getHttpServletRequest.invoke(request);
Long userID = (Long) hsr.getSession().getAttribute("USER_ID");
com.liferay.portal.model.User liferayUser = UserServiceUtil.getUserById(userID);
String userName = liferayUser.getScreenName();
setAuthenticatedUser(request, userName);
```

We get the liferay screenName as the userName just like CAS does. We then set it to the authenticated user for Alfresco and let Alfresco to do the validation later on.

The new method which used to do get the authentication bean, get the user and put it in the session is:

```
private void setAuthenticatedUser(PortletRequest req, String userName) {

        WebApplicationContext ctx = (WebApplicationContext) getPortletContext().getAttribute(WebApplicationContext.ROOT_WEB_APPLICATION_CONTEXT_ATTRIBUTE);
        ServiceRegistry serviceRegistry = (ServiceRegistry) ctx.getBean(ServiceRegistry.SERVICE_REGISTRY);
        TransactionService transactionService = serviceRegistry.getTransactionService();
        NodeService nodeService = serviceRegistry.getNodeService();

        AuthenticationComponent authComponent = (AuthenticationComponent) ctx.getBean("authenticationComponent");
        AuthenticationService authService = (AuthenticationService) ctx.getBean("authenticationService");
        PersonService personService = (PersonService) ctx.getBean("personService");

        // Get a list of the available locales
        ConfigService configServiceService = (ConfigService) ctx.getBean("webClientConfigService");
        LanguagesConfigElement configElement = (LanguagesConfigElement) configServiceService.getConfig("Languages").getConfigElement(
                        LanguagesConfigElement.CONFIG_ELEMENT_ID);

        m_languages = configElement.getLanguages();

        // Set up the user information
        UserTransaction tx = transactionService.getUserTransaction();
        NodeRef homeSpaceRef = null;
        User user;
        try {
            tx.begin();
            // Set the authentication
            authComponent.setCurrentUser(userName);
            user = new User(userName, authService.getCurrentTicket(), personService.getPerson(userName));
            homeSpaceRef = (NodeRef) nodeService.getProperty(personService.getPerson(userName), ContentModel.PROP_HOMEFOLDER);
            if (homeSpaceRef == null) {
                logger.warn("Home Folder is null for user '" + userName + "', using company_home.");
                homeSpaceRef = (NodeRef) nodeService.getRootNode(Repository.getStoreRef());
            }
            user.setHomeSpaceId(homeSpaceRef.getId());
            tx.commit();
        } catch (Throwable ex) {
            ...

            try {
                tx.rollback();
            } catch (Exception ex2) {
                ...
            }

            throw ...
        }

        // Store the user
        req.getPortletSession().setAttribute(AuthenticationHelper.AUTHENTICATION_USER, user);
        req.getPortletSession().setAttribute(LoginBean.LOGIN_EXTERNAL_AUTH, Boolean.TRUE);
    }
```

Then, get the user out from portlet session and validate the user, it is the original portlet class does:

```
if (user != null) {
    // setup the authentication context
    auth.validate(user.getTicket());
  }

  // do the normal JSF processing
  String loggedin = (String) getPortletContext().getAttribute("loggedin");
  if (loggedin != null && loggedin.equalsIgnoreCase("true") && viewId != null) {
    super.facesRender(request, response);
  } else {
    getPortletContext().setAttribute("loggedin", "true");
    response.setContentType("text/html");
    request.getPortletSession().setAttribute(PortletUtil.PORTLET_REQUEST_FLAG, "true");
    nonFacesRequest(request, response, "/jsp/browse/browse.jsp");
  }
```

Comments