User Group Replacement in KM Permissions

Applies to:

SAP Netweaver Portal (2004) / SAP Enterprise Portal SR1 SP9 and higher.

Summary

This article provides information on replacing a user group in assigning KM permissions using API's.

Author : Yogalakshmi Sathyanarayanan

Company : Tata Consultancy Services Ltd.

Created on : 07 March 2008

Author Bio

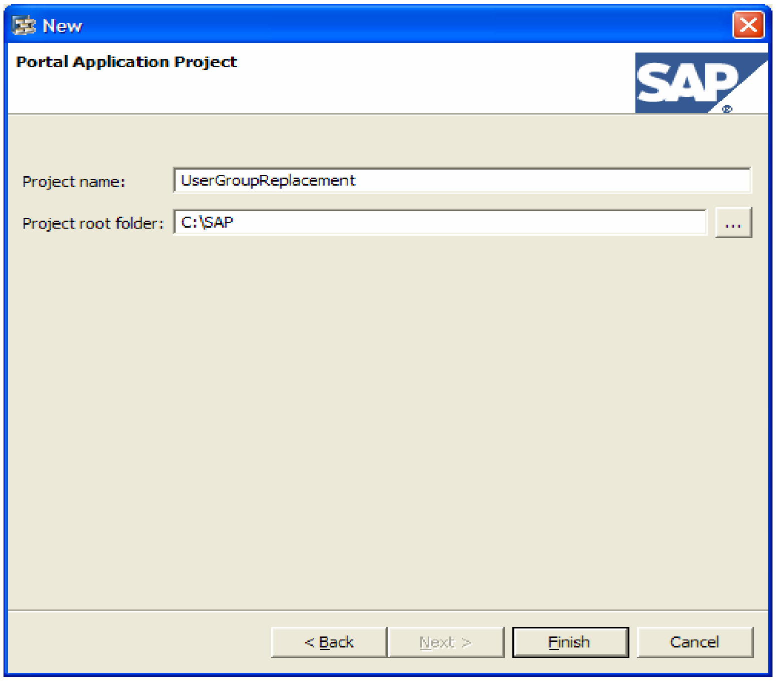
Yogalakshmi Sathyanarayanan is an Enterprise Portal Consultant working for Tata Consultancy Services Ltd and has an experience of nearly 3 years in Enterprise Portal.

Table of Contents

Getting Started

The scenario explained in this article can be used when there is a need to change the user groups in KM permissions i.e., replacing the existing user group with another group for any KM resource including all the child resources. Changing the permissions manually consumes much time and hence this solution can be used at that scenario. Below are the steps to be followed to achieve it.

1. Launch Netweaver Developer Studio and open EP perspective.
2. Create New Portal Application Project with a name, say UserGroupReplacement

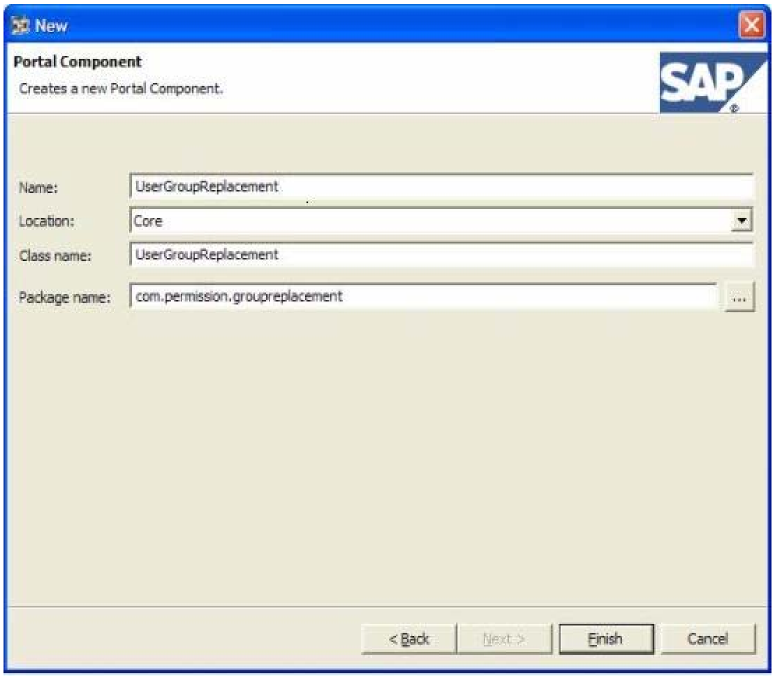


Portal Application Project

Project name: UserGroupReplacement

Project root folder: C:\SAP

3. Create a Abstract Portal Component by selecting New Portal Application Object and specify a name, say UserGroupReplacement



Portal Component

Creates a new Portal Component.

Name: UserGroupReplacement

Location: Core

Class name: UserGroupReplacement

Package name: com.permission.groupreplacement

4. Then add the code given in the section (code snippet) in the method *doContent* and then deploy the component on the portal server by selecting *File -> Export -> PAR File -> <Select the Server> -> Finish*. Make sure that you configure the portal server settings using *Configure servers settings* button. You can either execute the application directly by clicking on *Run* button in portalapp.xml or using an iView created from PAR.

Solution

KM permissions deal with the IAclSecurityManager. Steps involved in replacing user groups are as follows.

- Get the user group to be removed and the user group to be added
- Get the KM folder to which the user group replacement has to be done
- Get the list of resources in the KM folder by iterating recursively
- For each KM folder, check whether the group to be replaced is available in the inherited ACL entry
- If the user group exists, then change the ACL of the resource

Code snippet

This section provides the sample code to achieve the User Group replacement:

- **Method 1:** Getting the user group to be removed from the ACL list and to be added to the ACL list.

```
 private final static IUMPrincipal groupToBeRemoved;

 private final static IUMPrincipal groupToBeAdded;

 public void getGroup() {

   try {

   groupToBeRemoved = WPUMFactory.getGroupFactory().getGroup("<group id>");

   groupToBeAdded = WPUMFactory.getGroupFactory().getGroup("<group id>");

   }catch (Exception exception) {

    //Print the exception stack trace

   }

}
```

- **Method 2:** Getting the KM resource including its child folders to which the ACL needs to be changed.

```
 private final static IResource resource;

 public void getKMResource() {
       try {

       IUser serviceUser =
    WPUMFactory.getServiceUserFactory().getServiceUser("cmadmin_service");

    IResourceContext resourceContext = new ResourceContext(adminUser);

    resource = ResourceFactory.getInstance().getResource(RID.getRID("<KM path>"),
    resourceContext);

    }catch (Exception exception) {

    //Print the exception stack trace

    }

}
```

Note: The user group variables (groupToBeRemoved, groupToBeAdded) and the KM resource object (resource) can be declared as class variables.

- **Method 3:** Iterating through the KM resource recursively and getting all child folders and changing the ACL list for each resource.

```
private void listFolders(IResource resource) {

    try {

       //If the resource is an internal link, this resource will be skipped in
    the process

         if(resource.getLinkType().equals(LinkType.INTERNAL))

             return;

       //Checking whether the resource is a collection

       //if so, change the ACL and iterate recursively to get the child folders

       //if its not a colleciton, simply change the ACL

          if(resource.isCollection()) {

             changeACL(resource); //Method to change the ACL

             ICollection children = (ICollection)resource;

             IResourceList resourceList = children.getChildren();

             int size = resourceList.size();
```

```
            for(int counter = 0; counter < size; counter++) {

                  IResource childResource = resourceList.get(counter);

                  listFolders(childResource);

            }

         } else {

            changeACL(resource); //Method to change the ACL

         }

      }catch (Exception exception) {

            //Print the exception stack trace

      }

  }
```

- **Method 4:** Changing the ACL entries.

```
public void changeACL(IResource resource) {

    try {

    ISecurityManager sm = resource.getRepositoryManager().
    getSecurityManager(resource);

    if(sm != null && (sm instanceof IAclSecurityManager)) {

    IAclSecurityManager asm = (IAclSecurityManager)sm;

    IResourceAclManager ram = asm.getAclManager();


    // Getting the ACL list

    IResourceAcl ra = ram.getAcl(resource);

    if(ra == null)

           ra = ram.getInheritedAcl(resource);


    // Iterating through the ACL list

    IResourceAclEntryListIterator aclList = null;

    IResourceAclEntry acl = null;

    if(groupToBeAdded != null && groupToBeRemoved != null) {

           for(aclList = ra.getEntries().iterator(); aclList.hasNext();) {
              acl = aclList.next();



              //check whether the user group is present in the ACL

              //if so, remove the user group and add another user group

           if(acl.getPrincipal().getDisplayId().equals(groupToBeRemoved.getDispla
       yId())) {

              ra.removeEntry(acl);

              ra.addEntry(ram.createAclEntry(groupToBeAdded, false,
       ram.getPermission("read"), 0));

              }

           }

         }

       }

    }catch (Exception exception) {

           //Print the exception stack trace

    }

 }
```

Note: The permission is given as “READ” in the above snippet. It can be changed to write/delete as per the requirements. Also make sure that “sharing reference” for com.sap.km.application is added in the portalapp.xml file.

Related Content

- JavaDocs Reference
- SDN
- SAP Help

Disclaimer and Liability Notice

This document may discuss sample coding or other information that does not include SAP official interfaces and therefore is not supported by SAP. Changes made based on this information are not supported and can be overwritten during an upgrade.

SAP will not be held liable for any damages caused by using or misusing the information, code or methods suggested in this document, and anyone using these methods does so at his/her own risk.

SAP offers no guarantees and assumes no responsibility or liability of any type with respect to the content of this technical article or code sample, including any liability resulting from incompatibility between the content within this document and the materials and services offered by SAP. You agree that you will not hold, or seek to hold, SAP responsible or liable with respect to the content of this document.