ADF Code Corner

014. How-to use custom JAAS Permissions in a ADF Security to implement view layer security

Abstract:

The ADF Security Framework in Oracle ADF provides JAAS based authorization for ADF bounded taskflows and JavaServer Faces pages that have associated ADF bindings. In cases where the business service layer is ADF Business Components, fine grained security can be implemented on the attributes and operation binding level through the Entity Object security settings. A not so well known feature of ADF Security is its ability to secure view layer components with custom JAAS permissions. This how-to document shows how to secure a menu bar with a custom JAAS permission created in Oracle JDeveloper 11.

twitter.com/adfcodecorner

Author: Frank Nimphius, Oracle Corporation
twitter.com/fnimphiu
15-NOV-2008

Oracle ADF Code Corner is a loose blog-style series of how-to documents that provide solutions to real world coding problems.

Disclaimer: All samples are provided as is with no guarantee for future upgrades or error correction. No support can be given through Oracle customer support.

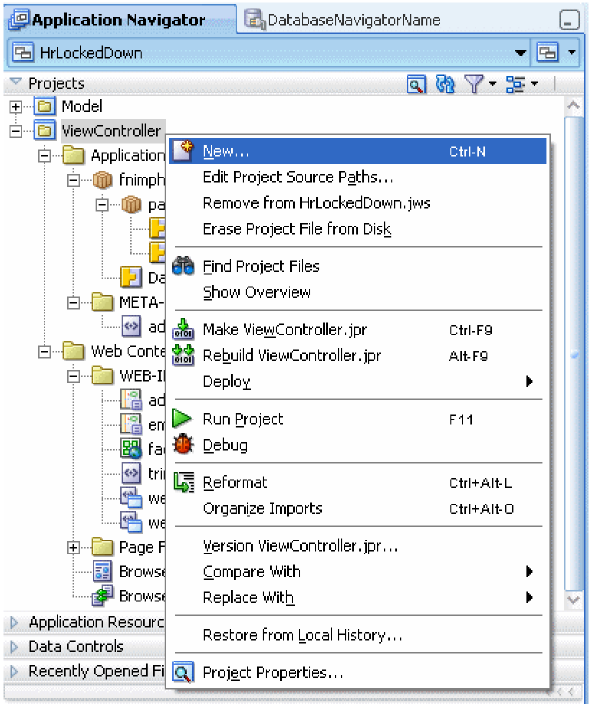
Please post questions or report problems related to the samples in this series on the OTN forum for Oracle JDeveloper: http://forums.oracle.com/forums/forum.jspa?forumID=83

Introduction

How-to documents are a snapshot to what is possible within a specific area of a product. ADF Security provides a rich set of security expressions that allow developers to hide or show UI components baased on user granted permission. Such a permission can be a custom permission that is configured and used within an ADF application to protect the UI layer of a web application, or the OPSS ResourcePermission, which is covered in various other documents on ADF Code Corner. This how-to explains how to build custom JAAS permissions and use them from EL using ADF Security.

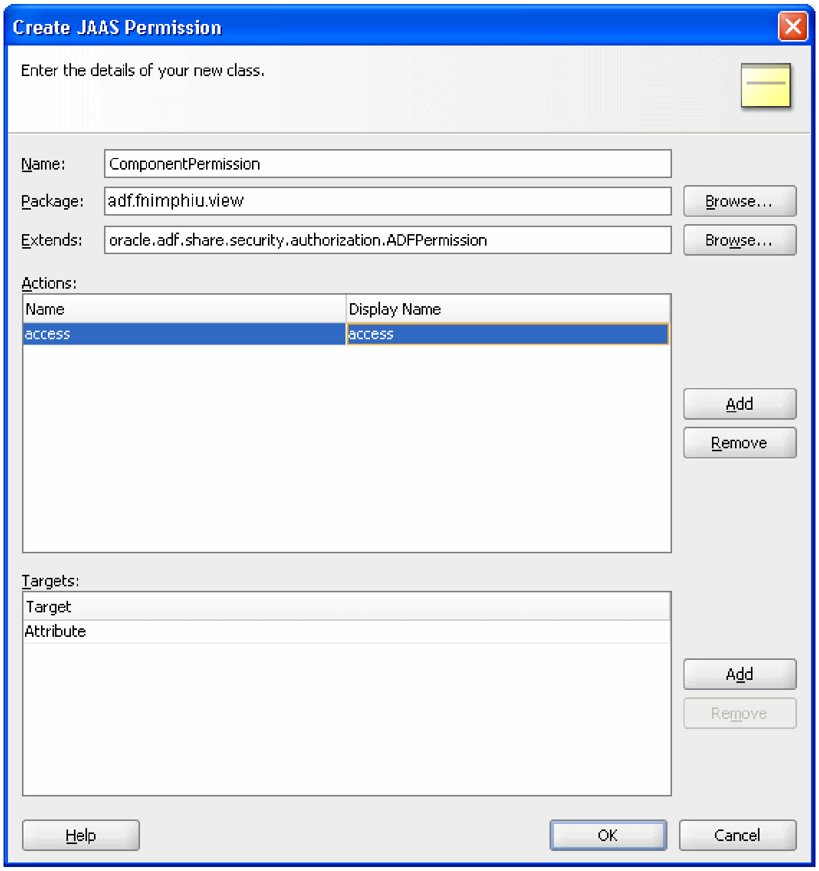
Creating a custom JAAS Permission for ADF Security

Starting from an existing ADF security enabled web application, developers can create a new JAAS permission by selecting **New | All Technologies | All Items | JAAS Permission**.



To build a custom JAAS permission for ADF Security, you need to extend `oracle.adf.share.security.authorization.ADFPermission`.

The permission can have an arbitrary name, like "ComponentPermission" in this example, and define a specific grant through its action and target property. The action for the permission below is defined as "access", but could also be "view", "execute", "run" or similar. It is also possible to define multiple actions for a permission class. For example, a manager and an employee are both allowed to view a page menu, but only the manager is allowed to execute a specific menu item. The target name is applied when adding the policy to the policy store, which is the application jazn-data.xml file in Oracle JDeveloper 11.


Create JAAS Permission

Enter the details of your new class.

Name: ComponentPermission

Package: adf.fnimphiu.view

Extends: oracle.adf.share.security.authorization.ADFPermission

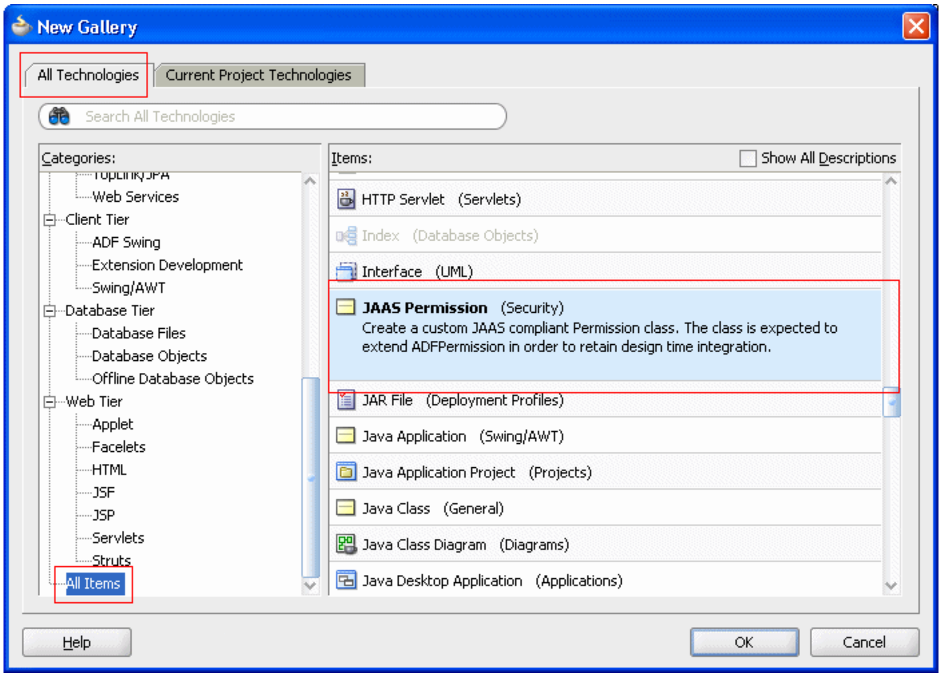
Actions:

Name	Display Name
access	access

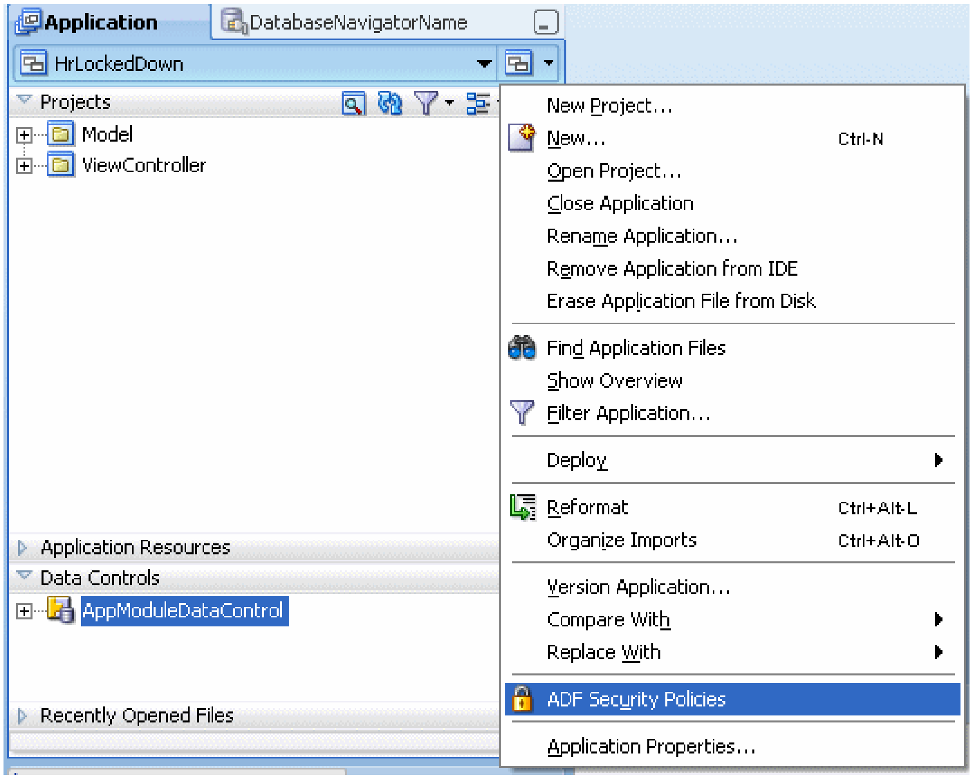
Targets:

Target
Attribute

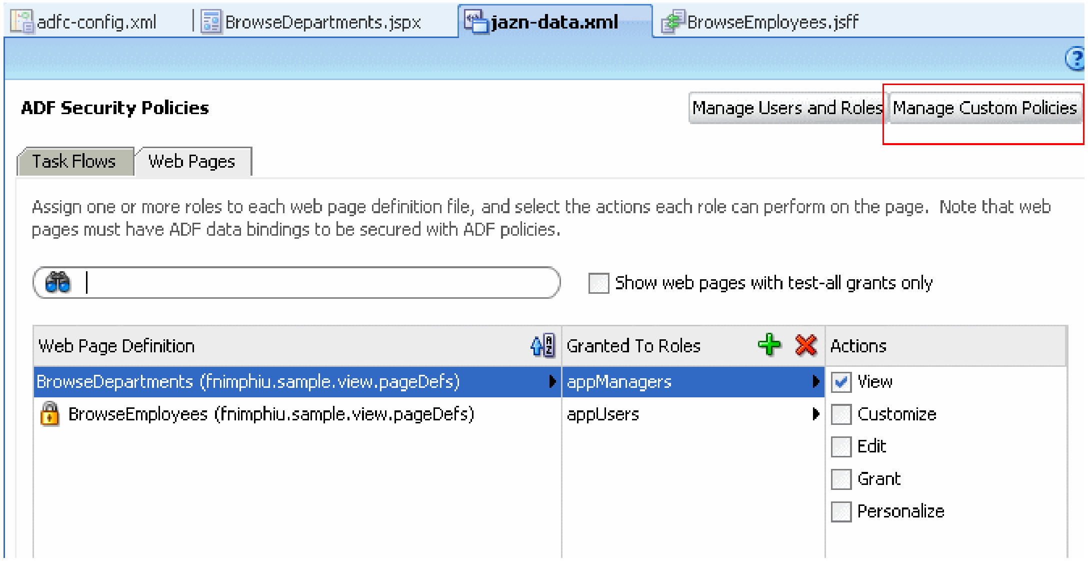
To add the custom permission class to the application jazn-data.xml policy store, open the ADF Security Policies Editor from the drop down list located next to the application name.



Create a custom JAAS compliant Permission class. The class is expected to extend ADFPermission in order to retain design time integration.



In the policies dialog, press the Manage Custom Policies button.



ADF Security Policies

Assign one or more roles to each web page definition file, and select the actions each role can perform on the page. Note that web pages must have ADF data bindings to be secured with ADF policies.

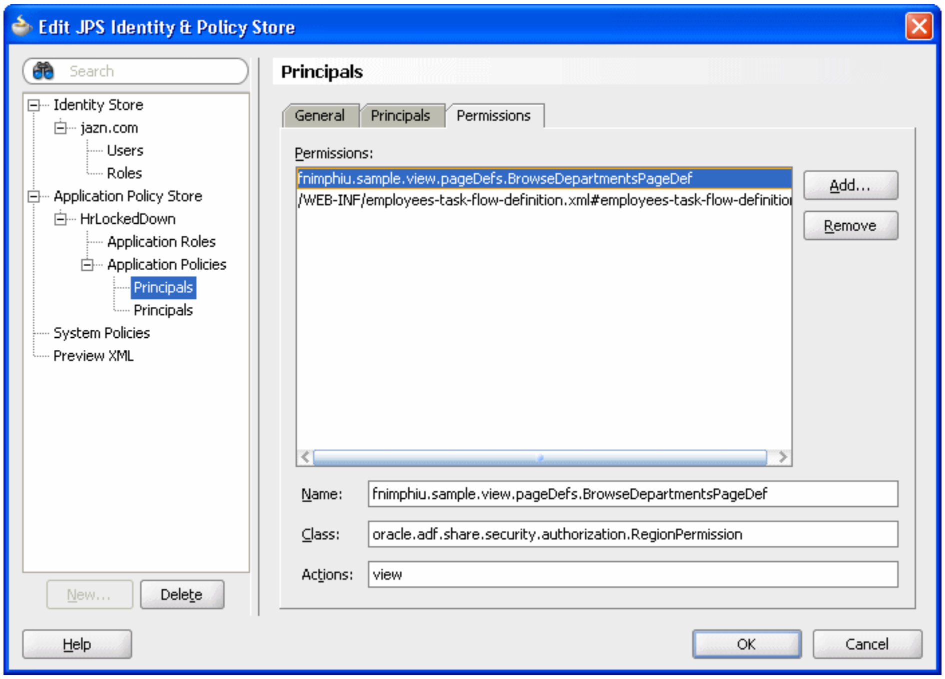
□ Show web pages with test-all grants only

Web Page Definition	Granted To Roles	Actions
BrowseDepartments (fnimphiu.sample.view.pageDefs)	appManagers	☑ View
BrowseEmployees (fnimphiu.sample.view.pageDefs)	appUsers	□ Customize
		□ Edit
		□ Grant
		□ Personalize

In the **Application Policy Store** section, create a new entry for the application roles and policies if no entry exist. The application policies are added to principals, which are application role principal.

In the sample application that was used to create the screen shots, two principal classes existed: One for "appManagers" and one for "appUsers".

To grant the new ComponentPermission to managers, the "appManagers" role principal entry is selected and the "Permissions" tab made current. Press the **Add** button to create a new grant.



Permissions:

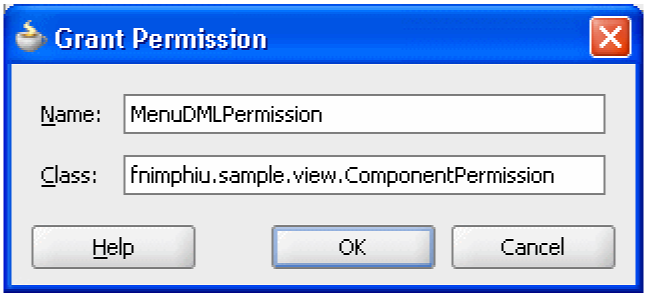
fnimphiu.sample.view.pageDefs.BrowseDepartmentsPageDef
/WEB-INF/employees-task-flow-definition.xml#employees-task-flow-definitio

Name: fnimphiu.sample.view.pageDefs.BrowseDepartmentsPageDef

Class: oracle.adf.share.security.authorization.RegionPermission

Actions: view

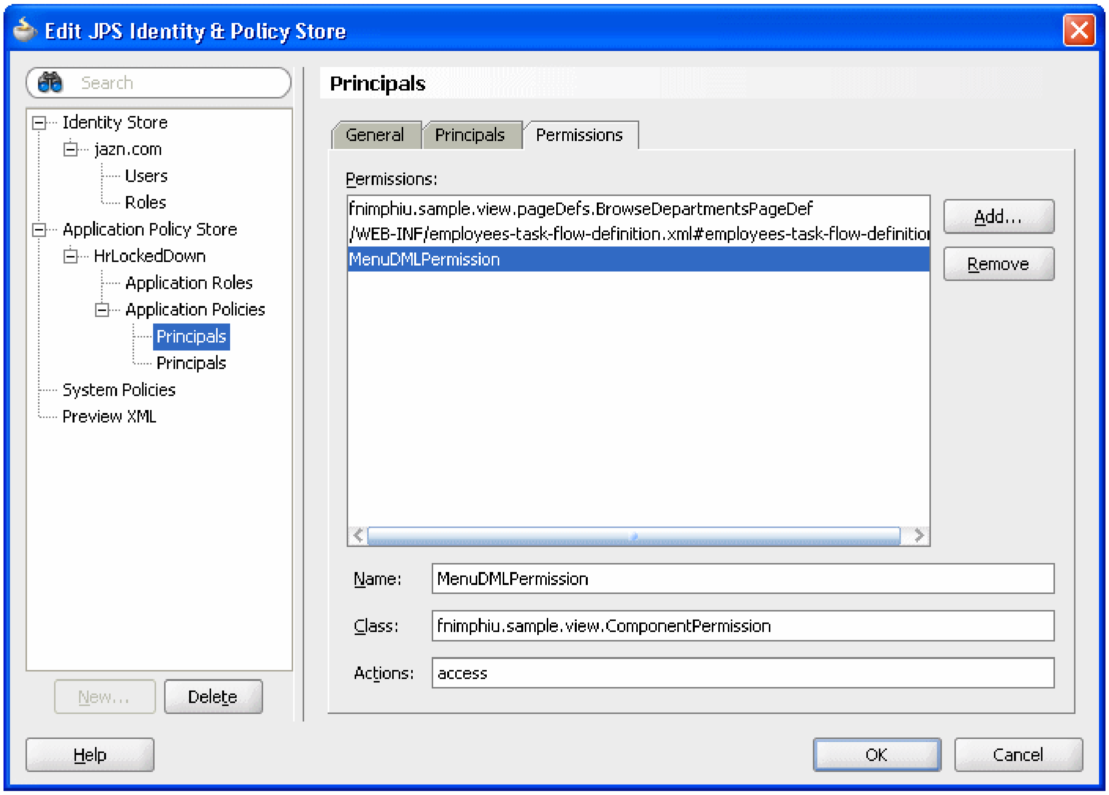
In the Grant Permission dialog, specify a target name for the grant, like "MenuDMLPermission" in this example and point it to the newly created JAAS permission class. Note that in production type of applications you may want to decide for a more formal and descriptive naming pattern. Since the JAAS permission class is only required to be in the class path, it is not mandatory to create the JAAS permission in the project where it is used.



Name: MenuDMLPermission

Class: fnimphiu.sample.view.ComponentPermission

Back in the **Permissions** dialog, select the new grant and add the allowed action into the **Actions** field. In this example, the only action possible is "access".



Permissions:

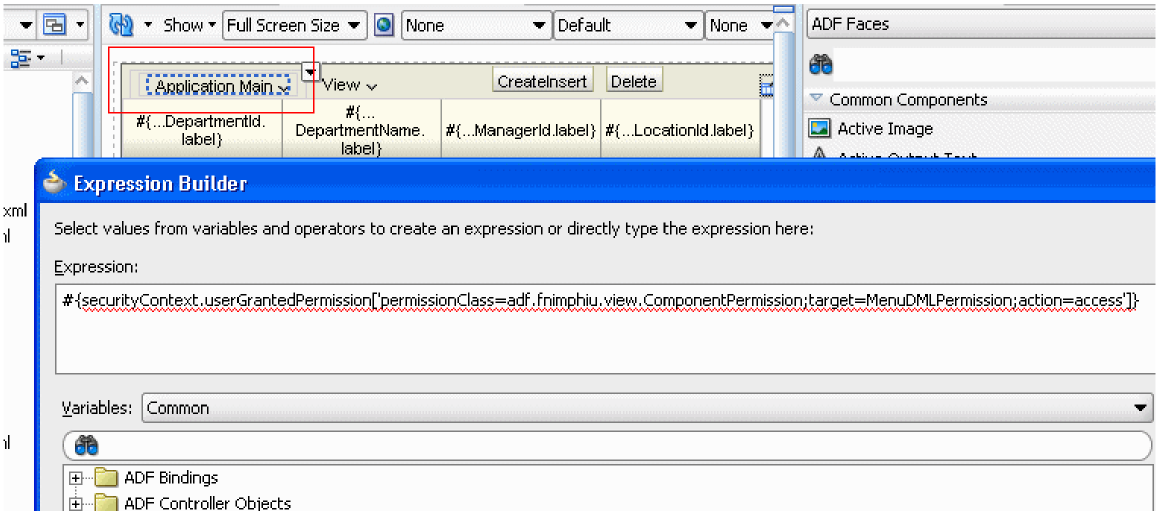
fnimphiu.sample.view.pageDefs.BrowseDepartmentsPageDef
/WEB-INF/employees-task-flow-definition.xml#employees-task-flow-definitio
MenuDMLPermission

Name: MenuDMLPermission

Class: fnimphiu.sample.view.ComponentPermission

Actions: access

To protect a UI component with ADF Security custom permission checks, select the component and open the Property Inspector [ctrl+shift+I]. In the property inspector, select the **Rendered** property and launch the Expression Builder dialog. Type in a security expression similar to the one shown in the image below to protect the menu with the JAAS permission. Note that the **target** attribute used in the expression matches the target name used in the grant. Same is true for the action, which must match the action granted in the policy.



Select values from variables and operators to create an expression or directly type the expression here:

Expression:

```
#{securityContext.userGrantedPermission['permissionClass=adf.fnimphiu.view.ComponentPermission;target=MenuDMLPermission;action=access']}
```

Running the application as an employee, in the appUsers role, the menu is not shown on the page. Running the same page authenticated as a user within the appManagers role, the menu is displayed.

Note: Be sensible when creating new permission classes for your application. Many security checks, like if a user is allowed to access a specific page, don't need an extra permission defined but can be based on an existing ADF Security permission.