SAP NetWeaver ’04 Security Tutorials

Tutorials for Using Roles and Permissions in Applications

Document Version 1.00 – March 2, 2005

THE BEST-RUN BUSINESSES RUN SAP

THE BEST-RUN BUSINESSES RUN SAP

SAP AG
Neurottstraße 16
69190 Walldorf
Germany
T +49/18 05/34 34 24
F +49/18 05/34 34 20
www.sap.com

Disclaimer
SAP code or applications samples and tutorials are NOT FOR PRODUCTION USE unless specifically noted. You may not demonstrate, test, examine, evaluate or otherwise use them in a live operating environment or with data that has not been sufficiently backed up.

You may not rent, lease, lend, or resell SAP code or application samples and tutorials.

Documentation in the SAP Developer Network
You can find this documentation at the following Internet address:
`sdn.sap.com`

Typographic Conventions

Type Style	Description
Example Text	Words or characters quoted from the screen. These include field names, screen titles, pushbuttons labels, menu names, menu paths, and menu options.
	Cross-references to other documentation
Example text	Emphasized words or phrases in body text, graphic titles, and table titles
EXAMPLE TEXT	Technical names of system objects. These include report names, program names, transaction codes, table names, and key concepts of a programming language when they are surrounded by body text, for example, SELECT and INCLUDE.
`Example text`	Output on the screen. This includes file and directory names and their paths, messages, names of variables and parameters, source text, and names of installation, upgrade and database tools.
`Example text`	Exact user entry. These are words or characters that you enter in the system exactly as they appear in the documentation.
`<Example text>`	Variable user entry. Angle brackets indicate that you replace these words and characters with appropriate entries to make entries in the system.
`EXAMPLE TEXT`	Keys on the keyboard, for example, *F2* or *ENTER*.

Icons

Icon	Meaning
⚠	Caution
💬	Example
💡	Note
🧭	Recommendation
SYN	Syntax

Additional icons are used in SAP Library documentation to help you identify different types of information at a glance. For more information, see *Help on Help* → *General Information Classes and Information Classes for Business Information Warehouse* on the first page of any version of *SAP Library*.

Tutorials for Using Roles and Permissions in Applications

Purpose

There are numerous possibilities for protecting your applications using security roles and permissions. Therefore, the first step is to decide which approach works best for the type of application you have. For example, you can protect your Web application using authentication and security roles for authorization, or you can protect the application at the backend to protect access to individual EJB methods. If the application is provided as a Web service, then you can also apply protection to the Web service.

Approaches

The primary approaches for including access protection in your applications are:

- Declarative

  With the declarative approach, you specify that your application should check permissions and the name of the role that is to be checked. The J2EE Engine performs the authorization check based on this declaration.

- Programmatic

  With the programmatic approach, you insert the authorization check into the application coding. In this case, the responsibility for the authorization check lies in the application and not in the Web container.

  Another approach is the use of access control lists (ACLs) to protect access to specific resources (for example, files in the file system) as objects. However, because this set of tutorials focuses on protecting access at the business application level, and not at the object level, we do not show the use of ACLs here. These tutorials only show the use of roles and permissions using either the declarative or programmatic approaches.

Types of Permissions

There are also two different types of permissions that you can use for protecting access to your applications:

- J2EE security roles

  J2EE security roles are security roles that are constructed according to the J2EE specification. They support both the declarative and programmatic approaches. When using the declarative approach, the information is stored in deployment descriptors for the application. Alternatively, you can use the available methods to perform checks in the application.

- UME permissions

  UME permissions are provided by the User Management Engine (UME) and extend the possibilities provided by the J2EE security roles. However, they are only supported when using a programmatic approach.

  One advantage for using UME permissions are that the administration is easier because you can consolidate permissions into actions. The administrator works with the actions and not with all of the individual permissions. In addition, you can perform more complex checks than those supported when using J2EE security roles, for example, you can use wildcards in the authorization check.

Determining Which Approach to Use

To determine which approach you should use, see the following table.

Approach	When to Use	Type of Permission to Use
Declarative	Use the declarative approach to protect access to static information or applications that are explicitly distinguishable from another, for example, to protect access to explicit EJB methods or to subsets of applications that are accessible using different URL paths.	J2EE security roles
Programmatic	Use the programmatic approach if you need more flexibility or more complex rules for allowing access. For example, use this approach to adjust screen items depending on the authorizations.	UME permissions

Although the J2EE security roles are also supported by the programmatic approach, in this case we recommend using UME permissions due to the increased flexibility.

Also, do not mix the use of J2EE security roles and UME permissions. If you decide to use UME permissions for a certain aspect of your application, then use UME permissions for the complete application. This makes the administration of the user and role assignments much easier.

For more information, see Using Security Roles and Security Role References [SAP Library] in the Development Manual for the J2EE Engine.

Overview of the Tutorials

Because of these numerous possibilities for using roles and permissions, in the following tutorials we concentrate only on the most frequently used scenarios. See the sections below:

Tutorial 1: Protecting Access to the J2EE-Based Application Using J2EE Security Roles

In this tutorial, you will protect access to the J2EE-based car rental application using authentication mechanisms and J2EE security roles. The Web client for this application is a JSP and servlet application; the business logic is implemented using entity beans. The table below shows the protection used for each component.

Protection Used for Tutorial 1

	Component	Protection
Presentation Layer	JSP and servlet	Authentication J2EE security roles
Business Logic	EJB	J2EE security roles

Tutorial 2: Protecting Access to the J2EE-Based Application Using UME Permissions and Actions

In this tutorial, you will protect access to the J2EE-based car rental application using authentication mechanisms and UME permissions and actions. The Web client for this application is the same JSP and servlet used in the first tutorial; the business logic is also implemented using entity beans. The table below shows the protection used for each component.

Protection Used for Tutorial 2

	Component	Protection
Presentation Layer	JSP and servlet	Authentication UME permissions
Business Logic	EJB	UME permissions

Tutorial 3: Protecting Access to a Web Dynpro and EJB Application When it is Available as a Web Service

In this tutorial, you will use a Web Dynpro client to access the EJB as a Web service.

To obtain the user ID, you will activate authentication on the Web Dynpro client as well as on the Web service.

Because the EJB methods can also be accessed directly, it is best to provide the authorization protection at the backend. Therefore you will include permission checks in the EJB methods.

Although it is possible, it is not necessary to provide authorization protection for the Web service for this tutorial.

We recommend applying authorization protection for Web services that access components that do not directly support access protection, for example, java classes that are available as Web services.

As an optional step, you can also check UME permissions in the Web Dynpro client and adjust the Web Dynpro screen based on the user's permissions.

The table below shows the protection to use at each level.

Protection Used for Tutorial 3

	Component	**Protection**
Presentation Layer	Web Dynpro	Authentication UME permissions (optional)
Middleware	Web service	Authentication
Business Logic	EJB	UME permissions

Applications to Use

The applications used in this set of tutorials are the J2EE quick car rental application and the Web Dynpro car rental application. Both of these applications are provided with the SAP NetWeaver Developer Studio example applications. In a default installation, you can find these applications in the directory *C:\Program Files\SAP\JDT\eclipse\examples*. They are provided with the archive files **`J2EE_QuickCarRental.zip`** and **`WebDynpro_CarRental.zip`** respectively.

The finished tutorials are also available on SDN in the Security download area.

Tutorials for Using Roles and Permissions in Applications

To continue with the tutorials for using roles and permissions, see:

- Protecting Access to the J2EE-Based Car Rental Application
- Protecting Access to the J2EE-Based Application Using UME Permissions
- Protecting Access to the Web Dynpro Application Using UME Permissions