Authentication of a Web Service Client with User/Password Request

Release 650

HELP.BCWEBSERVICES_TUTORIALS

Copyright

Icons in Body Text

Icon	Meaning
	Caution
	Example
	Note
	Recommendation
	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*.

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`.

Authentication of a Web Service Client with HTTP(S) User/Password Request

Task

In this tutorial, you will learn all the development steps required to configure an existing Web service in such a way that it authenticates itself to the Web service provider via a username and password. The predefined Web service *CreditCheck* is to be used.

For further information about the Web Service *CreditCheck*, refer to the following tutorial: Creating a Web Service.

The Web service *CreditCheck* offers the simple function of checking the credit-worthiness of a customer using the customer number. If a customer number between 1001 and 1005 is passed, the credit limit of the customer is returned. Otherwise a message informs you that no corresponding customer exists.

The client uses a jsp application to simulate a simple sales portal. If a customer makes an order, the system uses the corresponding customer number to check whether the sum of the order is within the credit limit of that customer. This check is carried out using the Web service *CreditCheck*.

Objectives

By the end of this tutorial, you will be able to:

- ✓ Authenticate an existing Web service client with the Web service provider using a username and password

Prerequisites

- ☐ You have worked through the tutorial *Configuring User/Password Authentication for a Java Web Service* or you have deployed the file CreditLimitCheckBasicEAR.ear which you can download here. You can call the Web service *CreditCheck*.
- ☐ All systems involved are configured for the use of SSL.

  You must install the SAP Java Cryptographic toolkit for this. With J2SE1.4, additional Sun Microsystems policy files are required; these must be copied to the directory *<java-home>\jre\lib\security*. For more information, refer to the section Deploying the SAP Java Cryptographic Toolkit

Systems, Installations, and Authorizations

- ☐ You have installed the SAP NetWeaver Developer Studio and configured it for use of the SAP J2EE Engine.
- ☐ You can access the SAP J2EE Engine using the Visual Administrator.

Knowledge

- ☐ You are familiar with the functioning of a JSP Web service client. For more information, refer to the section Creating a Client JavaServer Page.

Next Step:

Importing Projects [Page 6]

Importing Projects

The following projects are available for this tutorial:

- Deployable proxy project *CreditLimitCheckProxy* (the proxy used for the Web service CreditCheck*)*
- WEB project *CreditLimitCheckWeb* (the Web service consumer)
- EAR project *CreditLimitCheckWebEAR* (for deploying the Web project)

The projects are available in the file JAVA_CLIENT_JAVA_RAW.zip. To download the file, click here.

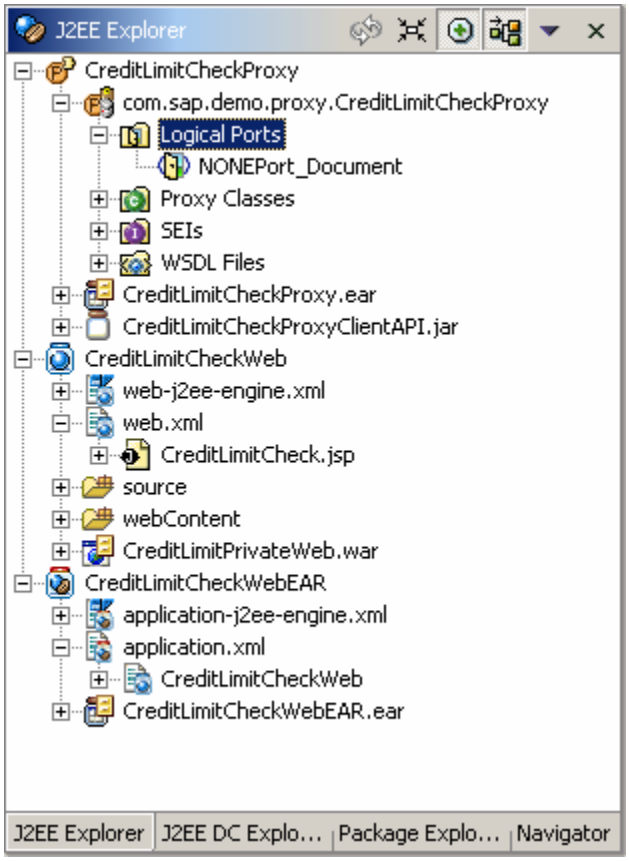
Procedure

Importing Projects into the SAP NetWeaver Developer Studio

1. Download the ZIP file JAVA_CLIENT_JAVA_RAW.ZIP, which contains the projects *CreditLimitCheckProxy*, *CreditLimitCheckWeb*, and *CreditLimitCheckWebEAR,* and save the ZIP file to any directory on your local hard disk or directly in the work area of the SAP NetWeaver Developer Studio.
2. Unpack the ZIP file.
3. Start the SAP NetWeaver Developer Studio and open the J2EE Development perspective.
4. Import the projects *CreditLimitCheckProxy, CreditLimitCheckWeb, and CreditLimitCheckWebEAR* by choosing *File →Import →Existing Project into Workspace.*

Result

Once you have imported the project into the Developer Studio, the following structure is displayed in the *J2EE Explorer*.



Next Step:

Creating and Using Logical Ports [Page 7]

Creating and Using Logical Ports

A logical port contains the configuration of the client-side SOAP runtime environment, such as the URL or the security settings of a Web service. If you want to access a Web service using a proxy, this is done via a logical port. Every logical port corresponds to a configuration of the Web service as offered by the Web service provider.

To be able to create a logical port, you first require a Web service definition. The latter can be obtained using the overview page of a Web service, for example. However, it can also be imported from the local file system or a UDDI registry.

This section shows you how, starting from an existing proxy, you can create a logical port and access it from within the application.

Prerequisites

☐ You have ensured that your Workbench proxy settings are the same as those in your Intranet. Check these settings in the Developer Studio by choosing *Window → Preferences → Workbench*

→ *Proxy Settings*.

☐ The structure of your projects *CreditLimitCheckProxy*, *CreditLimitCheckWeb,* and *CreditLimitCheckWebEar* is currently displayed in the *J2EE Explorer*.

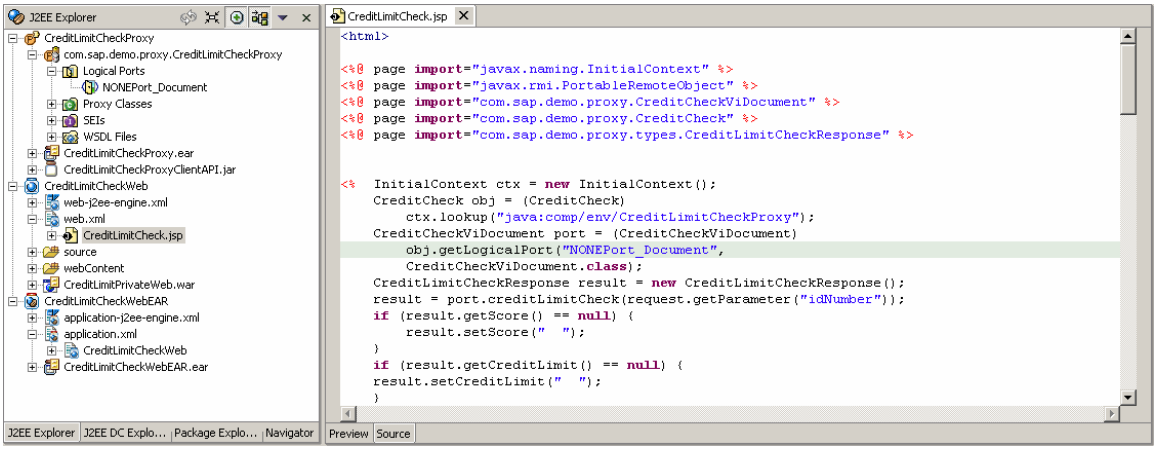
Procedure

Creating and Using Logical Ports

1. Display the overview page on the home page of the Web service *CreditCheck*. The link to the Web service definition can be found under the heading *WSDL*. Copy it to the clipboard.
2. To create the logical ports in the SAP NetWeaver Developer Studio, choose *New – Logical Ports* from the context menu for the node *CreditLimitCheckProxy – com.sap.demo.proxy.CreditLimitCheckProxy – Logical Ports*. The window *WSDL URL* appears.
3. In the field *Wsdl*, insert the link from the clipboard and confirm by choosing *OK*. The logical ports *BASICPort_Document* and *BASIC_SSLPort_Document* appear in your project hierarchy.

   If the specified logical parts should not appear, check whether you are using the Web service definition of the correct Web service. Also, note the prerequisites for this tutorial.
4. Open the file CreditLimitCheck.jsp by double-clicking the node *CreditLimitCheckWeb – web.xml – CreditLimitCheck.jsp*. Open the *Source* tab (bottom left in the preview window) to display the source code.
5. Search for the following statement in the source code: *obj.getLogicalPort("NONEPort_Document", CreditCheckViDocument.class).*



6. Replace the statement with:

```
CreditCheckViDocument port = (CreditCheckViDocument)
    obj.getLogicalPort("BASICPort_Document",
    CreditCheckViDocument.class);
```

7. Save your changes.

   The Web service consumer now uses the logical port *BASICPort_Document*, which accesses a configuration of the Web service that requires user/password authentication via HTTP.

Deploying the Changed Web Service and Proxy

1. Choose *Build EAR* from the context menu fort he project *CreditLimitCheckProxy* and deploy the file *CreditLimitCheckProxy – CreditLimitCheckProxy.ear* on the J2EE Engine.
2. Regenerate the files *CreditLimitCheckWeb.war* and *CreditLimitCheckWebEar.ear* by choosing *Build WEB Archive* or *Build Application Archive* from the context menus for the projects *CreditLimitCheckWeb* and *CreditLimitCheckWebEar* and deploy the file *CreditLimitCheckWebEAR.ear* on your J2EE Engine.

Result

Based on the consumer of the Web service CreditCheck you have created a new logical port called *BASICPort_Document* or *BASIC_SSLPort_Document* and have made them accessible by deploying the proxy on the J2EE Engine. The Web service is now configured in such a way that it uses the port *BASICPort_Document*, which accesses a configuration of the Web service that requires user/password authentication via HTTP.

Next Step:

Configuring the Web Service Client for HTTP Logon [Page 9]

Configuring the Web Service Client for HTTP Logon

Since the Web service provider requires user/password authentication, the Web service client must be configured to enable a valid logon to the provider.

Prerequisites

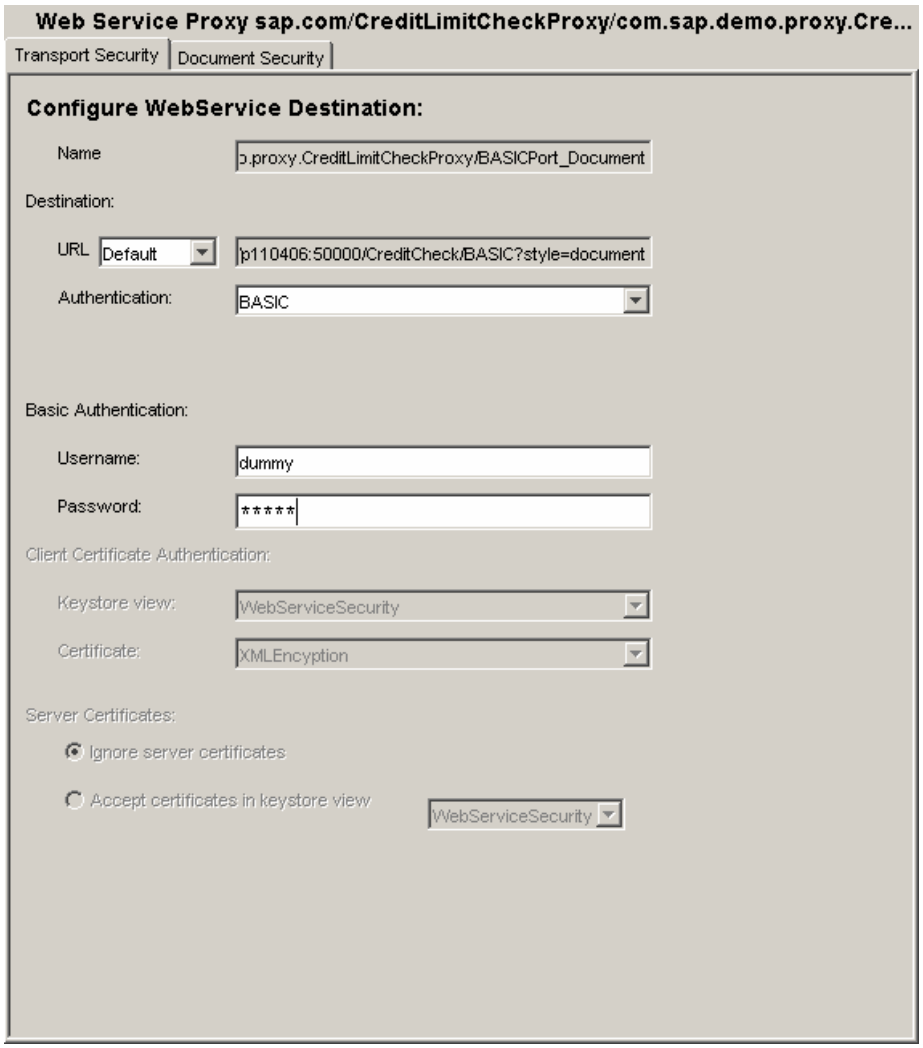
☐ The Web service proxy and the Web service client have been deployed on a SAP J2EE Engine.

Procedure

Configuring the Client

1. Start the Visual Administrator and log on to the SAP J2EE Engine of the client.
2. Select the node <*SID*> - *Server …* - *Services – Web Services Security*, where <SID> is the system ID of your client J2EE.
3. In the right window, activate the tabs *Runtime* (top) and *Security Administration* (bottom). A tree structure *Web Services* is displayed.
4. In this structure, select the node *Security Configuration – Web Service Clients – sap.com – CreditLimitCheckProxy – com.sap.demo.proxy.CreditLimitCheckProxy*BASICPort_Document*. A window containing the security measures for this Web service client is displayed on the right. In this window, activate the *Transport Security* tab.
5. Under *Authentication*, choose the option *BASIC*. This enables you to enter additional logon data.

6. In the fields *Username* and *Password*, enter valid user data for logging on to the system of the Web service provider and save your changes by choosing the *Save* button under the Web service tree structure.



Result

If the Web service client application *CreditLimitCheck* is executed, it logs on to the providing system as the specified user when the Web service *CreditCheck* is accessed; it does this by passing the user name and password via HTTP. Since this configuration will be changed again in the next steps, you should now test the client application to check that it functions correctly.

You can test the client application by calling the URL http://<host>:<port>/CreditLimitCheck in your browser. A simple screen for creating orders is displayed. After choosing the *Order* button, you are requested to enter the *Customer Identity Card Number.* If you press the *Check* button, the Web service CreditCheck is called and the value you entered in the text field is passed. The result is then evaluated and displayed. Note that only data for the customer numbers *1001* to *1005* is displayed. For all other values, a message is displayed stating that no customer was found for that number.

Next Step:

Configuring the Web Service Client for HTTPS Logon [Page 11]

Configuring the Web Service Client for HTTPS Logon

Since both username and password are clearly visible to third parties when passed via HTTP, it would be advisable to transfer this data using HTTPS. However, this also increases the configuration effort required. If the Web service client is also to be able to check the identity of the Web service provider, the certificate sent by the server must first be imported into the J2EE Engine of the client. It must then be defined as acceptable for the individual Web service clients.

Prerequisites

☐ The structure of your projects *CreditLimitCheckWeb* and *CreditLimitCheckWebEar* is currently displayed in the *J2EE Explorer*.

Procedure

Using the Corresponding Logical Port

1. Open the file CreditLimitCheck.jsp by double-clicking the node *CreditLimitCheckWeb – web.xml – CreditLimitCheck.jsp*. Open the *Source* tab (bottom left in the preview window) to display the source code.
2. Search for the following statement in the source code:

   *obj.getLogicalPort("BASICPort_Document", CreditCheckViDocument.**class**)*;

   Replace this statement with:
   *obj.getLogicalPort("BASIC_SSLPort_Document", CreditCheckViDocument.**class**);*

   ```
   CreditCheckViDocument port = (CreditCheckViDocument)
       obj.getLogicalPort("BASIC_SSLPort_Document",
       CreditCheckViDocument.class);
   ```

3. Save your changes.
4. The Web service consumer uses the logical port *BASIC_SSLPort_Document,* which requires a user/password authentication via HTTP.
5. Regenerate the files *CreditLimitCheckWeb.war* and *CreditLimitCheckWebEar.ear* by choosing *Build WEB Archive* or *Build Application Archive* from the context menus for the projects *CreditLimitCheckWeb* and *CreditLimitCheckWebEar* and deploy the file *CreditLimitCheckWebEAR.ear* on your J2EE Engine.

Configuring the Client

1. Start the Visual Administrator and log on to the client J2EE Engine.
2. Select the node <*SID*> - *Server … - Services – Web Services Security*, where <SID> is the system ID of your client J2EE.
3. In the right window, activate the tabs *Runtime* (top) and *Security Administration* (bottom). A tree structure *Web Services* is displayed.

4. In this structure, select the node *Security Configuration – Web Service Clients – sap.com – CreditLimitCheckProxy – com.sap.demo.proxy.CreditLimitCheckProxy*BASIC_SSLPort_Document*. A window containing the security measures for this Web service client is displayed on the right. In this window, activate the *Transport Security* tab.
5. Under *Authentication*, choose the option *BASIC*. This enables you to enter additional logon data.
6. In the fields *Username* and *Password*, enter valid user data for logging on to the system of the Web service provider and save your changes by choosing the *Save* button under the Web service tree structure.

Result

The username and password are now sent to the Web service provider via HTTPS and are thus no longer visible to third parties during transfer. However, the Web service client now accepts any certificate from the server without checking its identity. Such a check will be implemented in the next step.

Next Step:

Configuring the Web Service Client for Securing the Server Identity [Page 12]

Configuring the Web Service Client for Securing the Server Identity (Optional)

When a new SSL connection is opened, the server sends a certificate to the client for purposes of authentication. If the client is only to accept specific certificates, you must first import these certificates into the client and then define them as acceptable. By default, the Web service client accepts every certificate from the server.

Importing the server certificate is only necessary if your Web service client and the Web service to be consumed are not on the same J2EE. If this is not the case, you can skip the importing step and continue with the section *Configuring the Web Service Client for Checking the Server Identity* (see below).

Prerequisites

☐ You can access the certificate, which was exported as a crt file and is used by the server for authentication, via your file system.

Procedure

Importing the Server Certificate

Start the Visual Administrator and log on to the client J2EE.

1. Select the node <*SID*> - *Server …* - *Services* – *Key Storage* (where <SID> is the system ID of the client J2EE) and activate the tabs *Runtime* (top) and *Data* (bottom) in the right window.
2. Create a new keystore view by choosing the *Create View* button below the *Views* area. Give the new view a descriptive name - for example, *TrustedServers* - and confirm your entry. The new entry is displayed in the *Views* area.
3. Select your newly created keystore view.
4. In the *Entry* section (bottom right), choose *Load*. In the window that opens, select the certificate exported by the Web Service provider and choose *OK*. The certificate is now imported and is listed in the *Entries* field.

Configuring the Web Service Client for Checking the Server Identity

1. In the Visual Administrator, select the node <*SID*> - *Server …* - *Services* – *Web Services Security*, where <SID> is the system ID of your client J2EE.
2. In the right window, activate the tabs *Runtime* (top) and *Security Administration* (bottom). A tree structure *Web Services* is displayed.
3. In this structure, select the node *Security Configuration* – *Web Service Clients* – *sap.com* – *CreditLimitCheckProxy* – *com.sap.demo.proxy.CreditLimitCheckProxy*BASIC_SSLPort_Document*. A window containing the security measures for this Web service client is displayed on the right. In this window, activate the *Transport Security* tab.
4. In the *Server Certificates* area, activate the option *Accept certificates in keystore view* and then select the keystore view created in the last section from the list (for example, *TrustedServers*). If you skipped that section, because your client J2EE and server J2EE are identical, choose *service_ssl* here.
5. Save your changes by choosing *Save* below the Web service tree structure.

Result

The certificate sent by the server is now checked by the Web service client and only accepted if it is valid and is in the specified *keystore view*. Otherwise, the SSL connection is rejected and the Web service is not consumed.